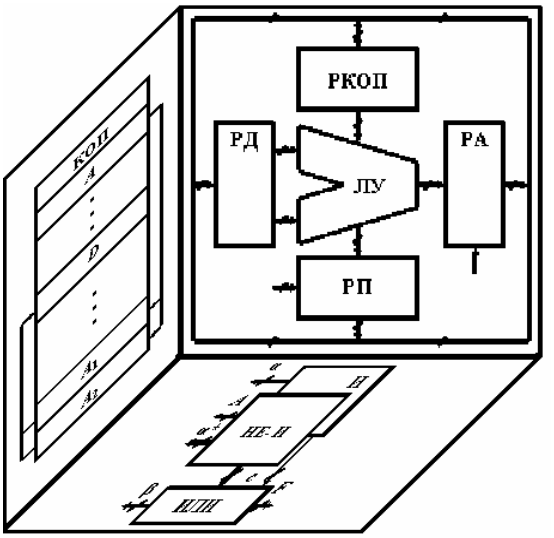
Е.И. ГЛИНКИН, М.Е. ГЛИНКИН

СХЕМОТЕХНИКА МИКРОПРОЦЕССОРНЫХ СРЕДСТВ



♦ ИЗДАТЕЛЬСТВО ТГТУ ♦

Министерство образования и науки Российской Федерации

Государственное образовательное учреждение
высшего профессионального образования
«Тамбовский государственный технический университет»

Е.И. ГЛИНКИН, М.Е. ГЛИНКИН

СХЕМОТЕХНИКА МИКРОПРОЦЕССОРНЫХ СРЕДСТВ

Утверждено к изданию секцией по техническим наукам
Научно-технического совета ТГТУ

Тамбов
Издательство ТГТУ
2005

УДК 681.335(07)
ББК 844.15я73-5
Г54

Рецензенты:

доктор технических наук, профессор
Б.И. Герасимов

доктор технических наук, профессор
Д.А. Дмитриев

Глинкин Е.И., Глинкин М.Е.

Г54 Схемотехника микропроцессорных средств. Тамбов:
Изд-во Тамб. гос. техн. ун-та, 2005. 148 с.

Проведен информационный анализ цифровых интегральных микросхем (ИС, СИС и БИС) для систематизации последовательностных преобразователей и методов анализа и синтеза в информационную технологию проектирования микропроцессорных средств. Триггеры, регистры и микропроцессор рассмотрены в комбинаторной, релейной и матричной логике на уровне аппаратных и метрологических средств, программного и математического обеспечения.

Монография предназначена для аспирантов и инженеров-исследователей, занимающихся вопросами автоматизации электрооборудования и технологических процессов, а также для студентов 2 – 4 курсов дневного и заочного отделений специальностей 100400 и 311400, 210200 и 220300, 2008 и 0720.

УДК 681.335(07)
ББК 844.15я73-5

ISBN 5-8265-0390-4

Научное издание

ГЛИНКИН Евгений Иванович,
ГЛИНКИН Михаил Евгеньевич

СХЕМОТЕХНИКА МИКРОПРОЦЕССОРНЫХ СРЕДСТВ

Монография

Редактор Т.М. Глинкина
Инженер по компьютерному макетированию Т.А. Сынкова

Подписано к печати 3.10.2005.
Формат 60 × 84 / 16. Бумага офсетная. Печать офсетная.
Гарнитура Times New Roman. Объем: 8,6 усл. печ. л.; 8,5 уч.-изд. л.
Тираж 400 экз. С. 665^{M}

Издательско-полиграфический центр
Тамбовского государственного технического университета
392000, Тамбов, Советская, 106, к. 14

ВВЕДЕНИЕ

Ускорение темпов информатизации требует развития информационных технологий проектирования микропроцессорных средств и обучения квалифицированных специалистов. Эффективность проектирования ограничивают итерационные методы программирования гибких последовательностных преобразователей, универсальность архитектуры которых регламентируют жесткие алгоритмы вычислений. Информационная технология проектирования комбинационных преобразователей не предназначена для создания интерфейсов памяти и микропроцессоров и требует развития для синтеза и анализа гибкой архитектуры последовательностных интегральных схем с согласованными компонентами микропроцессорных средств [1 – 14].

Повышение качества обучения при систематическом уменьшении количества почасовой нагрузки невозможно традиционными методами подготовки пользователей устаревшей техники комбинаторного типа без учета информатизации учебного процесса. Для разрешения качественно-количественного конфликта необходимо внедрение перспективных методов целенаправленного творчества с информационно-методическим обеспечением лекционно-лабораторного комплекса информационной технологии обучения микропроцессорным средствам.

Монография посвящена развитию информационной технологии проектирования микропроцессорных средств и обучения квалифицированных специалистов и служит информационно-методическим обеспечением анализа и синтеза ассоциативных структур логических элементов памяти с избыточными связями, программируемыми в адресном пространстве архитектуры матрицы. Технология проектирования цифровых комбинационных схем модифицирована для создания последовательностных преобразователей на различных уровнях иерархии от триггеров и регистров до микропроцессоров и компьютеров. При этом информационная технология обучения поднимает уровень эффективности учебного процесса за счет систематизации знаний по объективным закономерностям созидания интеллектуального продукта. Монография, как учебно-методическое обеспечение, повышает оперативность и информативность лекционного курса, практических занятий и лабораторного практикума за счет представления теоретического материала по законам гносеологии и дидактики от простого к сложному с учетом преемственности и последовательности интеграции творческих навыков и информационных процессов.

Результатом информатизации научно-технической революции является внедрение персональных компьютеров в автоматизацию электрооборудования и технологических процессов, приборостроения и аналитического контроля за счет развития информационной технологии проектирования микропроцессорных средств. Информационная технология интегрирует перспективные методы анализа и синтеза компонент и базисных структур микроэлектроники, основанных на информационных принципах аналогии и эквивалентности, инверсии и симметрии, отражающих объективные закономерности программного управления цифрового преобразования в адресном пространстве микросхемотехники.

Методы проектирования дифференцированы в координатах адресного пространства информационной модели по компонентам информационного обеспечения на аппаратные и метрологические средства топологии схем и оценки эффективности, программное и математическое обеспечение мнемоники алгоритмов и логических операторов. Согласованным компонентам микропроцессорных средств информационная модель согласно информационным принципам ставит в соответствие оптимальные формы представления логических процессов базисных структур микроэлектроники различных уровней иерархии от полупроводниковых приборов (ПП) и интегральных схем малой (ИС), средней (СИС), большой (БИС) степени интеграции до персональных компьютеров (ПК).

Диалектика развития базисных структур и компонент микропроцессорных средств систематизирована в информационной концепции интеграции логических функций от обмена энергией в ПП, преобразования сигнала в ИС и управления структурой в СИС при становлении аппаратных средств (АС) до хранения информации в программно управляемом адресном пространстве БИС при появлении программного обеспечения (ПО), интегрированного с АС в архитектуру. Архитектура отражает гибкое программирование избыточных связей, организованных в многомерное матричное пространство ассоциации элементарных функций дизъюнкции, конъюнкции и инверторов по множеству алгоритмов универсальной математической модели.

Гибкие алгоритмы и универсальные модели – неделимые грани математического обеспечения (МО) персональных компьютеров (ПК), интегрирующих функцию программного управления цифровым преобразованием (программирования) в вычислительный процесс обработки информации. Развитие вычислений в измерение за счет анализа по эквивалентным мерам приводит к созданию метрологических средств (МС) для оценки эффективности компонент информационного обеспечения микропроцессорных измерительных средств (МИС). Информационная концепция диалектического развития информационных процессов при интеграции базисных структур организует микроэлектронику и измерительную технику в микросхемотехнику. Принципы микросхемотехники преобразуют оптимальные формы представления логических функций схемо- и мнемотехники, математики и физики в согласованные компоненты с гибкой архитектурой, информативным математическим обеспечением и эффективными метрологическими средствами.

С позиций информационной концепции в монографии приводится азбука микросхемотехники, включающая информационную технологию проектирования микропроцессорных средств на уровне неделимого комплекса компонент информационного обеспечения и форм представления логических функций на различных уровнях иерархии. Азы и аксиомы логики комбинаторного, релейного и матричного базисов систематизированы по рациональным методам итерационного анализа, алгебры Буля и математики образов информационной технологии проектирования. Элементы последовательностных цифровых ИС от простых статических до сложных динамических триггеров спроектированы в комбинаторной, релейной и матричной логике методами делителей напряжения и токов, структурных формул и единиц и нулей, аналогии и эквивалентов в основных формах схемо- и мнемотехники, математики и физики. Структурные схемы и формулы, векторные таблицы состояния и семейства временных диаграмм иллюстрируют перспективные методы на примере базисов ИЛИ-НЕ и И-НЕ, нормальных форм дизъюнкции (НДФ) и конъюнкции (НКФ). Анализируется становление жесткой структуры СИС в гибкую архитектуру программируемых логических матриц (ПЛМ) методами программирования по эквивалентам универсальных триггеров и бинарных счетчиков, сдвиговых регистров и программируемых знакогенераторов. Показано развитие метода эквивалентных программ при тиражировании элементарных модулей в открытую архитектуру адресного пространства при тождественности исследуемого решения физическим эквивалентам. Технология проектирования микропроцессорных средств проиллюстрирована при сопоставительном анализе развития вычислителей с жесткой структурой в гибкую архитектуру персональных компьютеров за счет организации релейной логики процессора в адресное пространство микропропрограммного управления матричной логики микропроцессора. Синтезирована обобщенная архитектура микропроцессора и его регистров в процессе анализа техники адресации при создании ствола программы и условных признаков ветвления подпрограмм.

Данная работа развивает информационную концепцию цифровых комбинационных ИС и СИС в микросхемотехнику БИС и ПК с последовательностной структурой и является логическим продолжением монографий [12, 13, 22] и учебных пособий [16 – 21] по электронике и информационно-измерительной технике. Теория проектирования запоминающих устройств интерфейсов памяти и микропроцессоров положена в основу цикла «Информационно-измерительные системы ВЭЛ». Теоретические материалы систематизируют тридцатилетний опыт учебно-методической и научно-исследовательской работы авторов по цифровой, импульсной и микропроцессорной технике для автоматизации аналитического контроля и технологических процессов, электрооборудования и электроснабжения, конструирования радиоэлектронных и микропроцессорных средств. Монография предназначена для инженерного синтеза и анализа интерфейсов и микропроцессорных средств на практике в научных исследованиях аспирантов и магистров, а также учебном процессе студентов дневной и заочной формы обучения.

Авторы благодарят преподавателей и аспирантов кафедры «Электрооборудование и автоматизация» за обсуждения и замечания, послужившие повышению качества изложения материала, рецензентов д-ра техн. наук, профессора Б.И. Герасимова и д-ра техн. наук, профессора Д.А. Дмитриева за ценные советы учебно-методического характера, а также сотрудников ИПЦ университета за эффективную техническую помощь при подготовке и публикации работы.

1 АЗЫ И АКСИОМЫ ЛОГИКИ

Азы и аксиомы логики – основа информационной концепции микросхемотехники, классифицирующей цифровые схемы и методы их проектирования по упорядоченности адресации логических функций в основных формах представления науки и техники для систематизации принципов анализа и синтеза базисных структур микроэлектроники в информационную технологию проектирования микропроцессорных средств.

Информационная концепция [12 – 24] отражает диалектическое развитие информационных процессов при интеграции базисных структур в компоненты информационного обеспечения на уровне аппаратных и метрологических средств, программного и математического обеспечения. Компоненты интегрированы в информационную модель микропроцессорных средств, дифференцированную по координатам управления в основных формах представления схемо- и мнемотехники, математики и физики для их согласования в адресном пространстве логических функций. Анализ и синтез согласованных компонент и базисных структур на различных уровнях иерархии организованы по объективным закономерностям, сформулированным в виде принципов аналогии и эквивалентности, инверсии и симметрии, методы которых объединены в информационную технологию проектирования микропроцессорных средств.

Азы микросхемотехники [1, 2, 15 – 22] в основных формах логических функций иллюстрируют аксиомы операторов дизъюнкции, конъюнкции и инверсии, как результат параллельного, последовательного и смешанного соединения структур и связей в адресном пространстве элементарных схем и таблиц, правил и диаграмм. В образах науки и техники поясняются аксиомы булевой алгебры на примере алгоритмов и формул, элементарных структур и связей. Аксиомы математической логики одновременно служат дидактическим примером представления элементарных функций логического сложения, умножения и инверсии в виде многогранного неделимого комплекса топологических и мнемонических, алгоритмических и метрологических образов.

Азы логики [29, 30, 35 – 38, 57 – 67] отражают информационный анализ базисных структур цифровых полупроводниковых интегральных схем и известные способы их анализа и синтеза для выбора оптимальных методов проектирования ПП и ИС, СИС и БИС в комбинаторной, релейной и матричной логике. Аксиомы логики поясняют основы логических операторов элементов дизъюнкции, конъюнкции, инверсии для их сопоставительного анализа и синтеза в процессе организации согласованных базисных структур и компонент микропроцессорных схем.

1.1 МЕТОДЫ ПРОЕКТИРОВАНИЯ СХЕМ

Электротехника и электроника обеспечены широким арсеналом методов анализа аналоговых, импульсных и цифровых схем, так как исторически, основные разделы математики вызваны необходимостью рассчитывать линейные, нелинейные и квазилинейные функции операторами арифметического, алгебраического и интегродифференциального исчисления. Представление информации в цифровой форме привело к бурному развитию методов счисления, формирующих коды в нормальной дизъюнктивной (НДФ) и конъюнктивной (НКФ) форме алгебры Буля – основы релейно-контактных схем. Появление цифровых схем в диодной (ДЛ), транзисторной (ТЛ), диодно-транзисторной (ДТЛ) логике и на интегральных схемах (ИС) потребовало совершенствования итерационных методов анализа, систематизирующих произвольную адресацию и учитывающих вентильный эффект нелинейных элементов. Методы узловых потенциалов и сигнальных графов, делителей напряжения и тока позволили применить закон Ома и правила Кирхгофа при расчете ДТЛ и ИС. Однако, при всей изящности известных методов анализа они сложны и трудоемки для синтеза цифровых схем из-за проектирования желаемой функции многошаговым последовательным приближением методами итерационного анализа.

Методы булевой алгебры [1, 2, 12, 13, 29, 30, 37, 63, 67] рациональны для синтеза структурных формул в НКФ и НДФ при анализе цифровых схем и таблиц истинности ограниченного формата, не превышающего размерность 4×4 по числу входов и выходов ИС. Анализ средних (СИС) интегральных схем комбинационного и последовательностного типа организуют минимизационными булевыми преобразованиями в адресном пространстве морфологических диаграмм с использованием карт Карно и Вейча. Для проектирования цифровых больших (БИС) интегральных схем предложено векторное исчисление [38], не получившее применения на практике из-за сложности операторов транспонирования матриц.

Многообразие методов проектирования логических функций в различных формах представления [20 – 38, 41 – 53, 62 – 71] физиков и математиков, программистов и электриков приводит к несогласованности аппаратных и метрологических средств, программного и математического обеспечения на различных уровнях интеграции (ДТЛ и ИС, СИС и БИС) и элементных базисах (диоды и транзисторы, реле и тиристоры) цифровых схем. Информационная технология проектирования микропроцессорных средств требует классификации известных и создания универсальных методов анализа и синтеза логи-

ческих функций в схемо- и мнемотехнике, в образах математики и физики, эффективных для рационально систематизированных цифровых схем.

1.1.1 Классификация схем по упорядоченности

Лидирующее развитие микропроцессорной техники обусловлено современной фазой научно-технической революции – информатизацией [16], результата интеграции микроэлектроники и приборостроения при дифференциации топологии схемотехники в упорядоченное адресное пространство мнемотехники. Матричная структура позволила систематизировать комбинаторную структуру полупроводниковых приборов, малых (ИС) и средних (СИС) интегральных схем в высокоорганизованные алгоритмом программного управления большие интегральные (БИС) и персональные компьютеры (ПК). Анализ базисных структур от ПП до БИС и от ПК до микропроцессорных средств с позиции информационной концепции интеграции информационных процессов, т.е. по вектору организации порядка, показывает, что целесообразно классифицировать цифровые схемы по упорядоченности информации от хаотических структур комбинаторики ПП, ИС, СИС до организованных архитектур программируемой логики БИС, ПК, МИС на комбинаторные, релейные и матричные (рис. 1.1).

Хаотичность комбинаторных схем [2, 25, 29 – 30, 34 – 36, 67] обусловлена созданием логической функции F из множества элементных структур с различными функциями f_k, беспорядочно расположенных в декартовых координатах топологии. Это обусловлено появлением электронных схем из элементов электротехники, при этом пассивные делители на резисторах, индуктивностях и конденсаторах дополнялись вентилями на диодах и усилителями на транзисторах. По аналогии с линейными преобразователями на резисторах и квазилинейными схемами на реактивных элементах появились нелинейные цепи на электромеханических реле. В процессе развития микроэлектроники механические реле уступили место полупроводниковым на диодах, тиристорах и транзисторах. Комбинаторные цифровые схемы проходят становление от диодно-транзисторной логики к интегральным схемам, при этом из множества различных элементарных функций f_k, где $k = \overline{\{0, l\}}$, создается логическая функция $F = \{f_k\}_0^l$ на структурах с безадресной топологией. Например, электронный ключ (см. рис. 1.1) собран на диоде f_1 и транзисторе f_2, электромагнитной катушке f_3 и нормально открытыми контактами f реле P_f, термостабилизирующей цепочке на резисторе f_4 и конденсаторе f_5. Из-за реализации комбинаторной логической схемы на линейных f_4, квазилинейных f_3, f_5 и нелинейных f_1, f_2 элементах анализ функции F возможен операторными и интегродифференциальными методами, методами узловых потенциалов и комплексного переменного, делителя напряжения и токов. Однако синтез функции в комбинаторной логике вышеперечисленными методами трудоемок и нетехнологичен из-за поиска решений последовательным приближением методами итерационного анализа. Следовательно, мощный математический аппарат для расчета линейных, нелинейных и квазилинейных электрических цепей не эффективен при синтезе электронных функций в комбинаторной логике из-за итерационного анализа различных структур с безадресной топологией (рис. 1.1, 1.2).

Наиболее упорядочены БИС с архитектурой матричной логики [12, 38, 41 – 49], организованной в программно управляемом адресном пространстве с избыточной ассоциативной структурой матрицы. Структуру матрицы организуют из пересечения проводников в виде m строк и n столбцов, в узлах соединений которых размещены однотипные элементы с фиксированной функцией f_c (см. рис. 1.1). Состояние функции в ij-м узле многомерной матрицы $m \times n$ программно управляется кодом адресации по j-й строке ($j = \overline{0, m}$) и i-му столбцу ($i = \overline{0, n}$). Из-за избыточности ассоциативных связей и структур матрица БИС перестраивается алгоритмом программы на заданную оператором функцию $F = \{f_{c\,ij}\}_{0,0}^{n,m}$. При этом ассоциативная матричная структура организована по правилам адресного пространства в гибкую архитектуру для реализации заданного алгоритма по универсальной математической модели. Для проектирования программируемых матриц не подходят традиционные методы итерационного анализа комбинаторной логики с регламентированными алгоритмами логических функций, реализуемых жесткими структурами с неуправляемыми связями. Методы алгебры Буля не эффективны для проектирования многомерных матриц с адресным пространством $m \times n$ более 4×4 [15, 16].

Для анализа и синтеза функций в матричной логике [12 – 24, 38] с гибким алгоритмом программирования и универсальной математической моделью необходимы коммуникабельные операторы, систематизирующие закономерности известных методов в информационную технологию проектирования микропроцессорных средств. Коммуникабельные операторы систематизированы в математике образов, идентифицирующей компоненты и функции представления микропроцессорных средств и матричных архитектур БИС по нормированным эквивалентам аппаратных и метрологических средств, программ-

ного и математического обеспечения. Математика образов, как логическое развитие векторного анализа, оперирует закономерностями идентификации, систематизированными в принципы аналогии и эквивалентности, дуальности и симметрии. Математический аппарат идентификации образов компонент и функций представления в основных формах науки и техники интегрирует априорную информацию в технологию проектирования перспективных микропроцессорных средств [15, 16, 22].

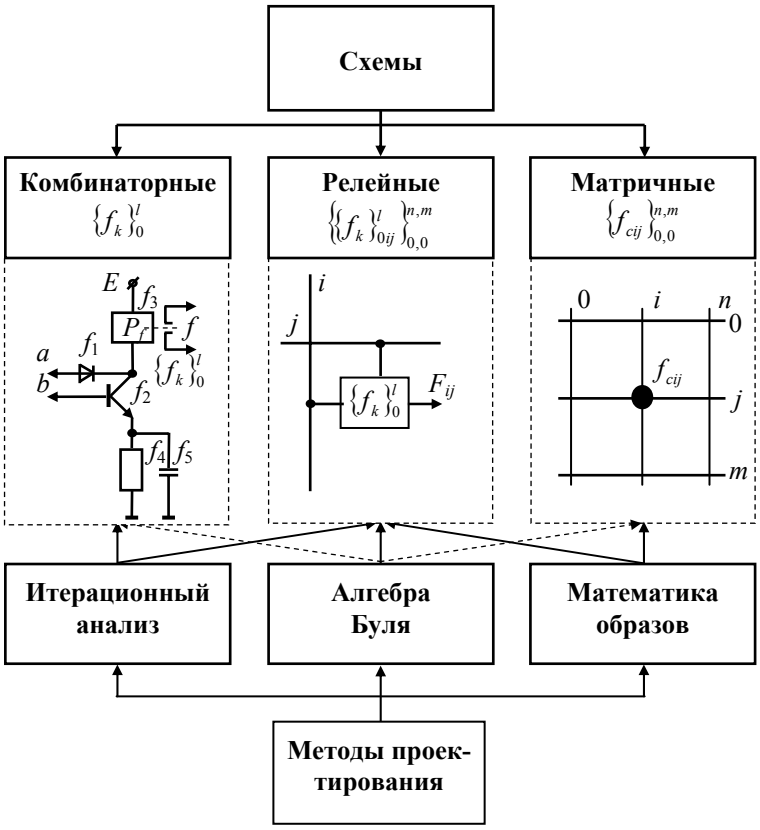


Рис. 1.1 Классификация логических схем по упорядоченности и методам проектирования

Итерационный анализ		Алгебра Буля	
Делитель напряжения (МДН)	Метод токов (МТ)	Единиц и нулей (М10)	Структурные формулы (МСФ)
Аналогия (МА)	Эквивалентность (МЭ)	Инверсия (МИ)	Симметрия (МС)
МАТЕМАТИКА ОБРАЗОВ			

Рис. 1.2 Классификация методов проектирования

Следует отметить, что программируемая матричная архитектура может быть организована на любых однотипных элементах с фиксированной функцией накопления электрической и магнитной энергии, положенных в основу приборов с зарядовой связью (ПЗС) и ферромагнитных носителей (ФМН); вентильного и усилительного эффекта, реализуемого в интерфейсах ввода-вывода, памяти и микропроцессорах; тензо-, термо- и фотоэффектов, необходимых при проектировании программируемых первичных преобразователей (ПИП) и органов управления автоматизацией электрооборудования и объектами автоматического контроля, радиоконструировании систем и технологических процессов. Следовательно, для анализа и синтеза гибкой матричной логики с универсальной математической моделью целесообразны универсальные операторы математики образов, идентифицирующие по нормируемым мерам компоненты микропроцессорных средств и основные формы представления функций БИС в науке и технике.

Между комбинаторными и матричными схемами по упорядоченности находится релейная логика [12, 13], организованная по структуре матриц с систематизированными в ограниченном адресном пространстве строками и столбцами, в *ij*-х узлах которых включены комбинаторные схемы реле (см. рис. 1.1). Электронное реле реализовано по комбинаторному принципу из функционально законченных элементов f_k, безадресно расположенных в *ij*-м сегменте *ij*-го узла релейной матрицы. При этом в узлах проектируют однотипные релейные функции $F_{ij} = \{f_k\}_0^l$, а функция матрицы является ассоциацией $F = \{F_{ij}\}_{0,0}^{n,m}$ комбинаторных функций F_{ij}, организованных в архитектуру ограниченного адресного пространства. Как и в матричной логике, программное управление узлами релейной матрицы формируют по кодам операций, систематизированных в пошаговую последовательность программ. Для проектирования контактно-релейной логики предложен логический аппарат алгебры Буля, включающий методы «единиц и нулей (М10)» и «структурных формул (МСФ)» для синтеза соответственно таблиц истинности и математических моделей в процессе анализа релейных схем. Методы булевой алгебры более эффективны не в комбинаторной, а матричной логике. Учитывая комбинаторно-матричный характер релейной логики, следует признать, что оптимальными методами ее анализа и синтеза являются булевы преобразования, а также информационные методы [15 – 22] математики образов (см. рис. 1.1).

Из анализа схем по упорядоченности следует их классификация по вектору информативности на комбинаторные, релейные и матричные, для синтеза которых рациональны соответственно методы итерационного анализа, алгебры Буля и математики образов. Из-за комбинаторно-матричной формы релейной логики возможен синтез комбинаторных реле методами итерационного анализа, а их матричной архитектуры – методами аналогии для создания информационной технологии проектирования микропроцессорных средств.

1.1.2 Методы итерации по эквивалентам

Сложность анализа комбинаторных схем в диодной, транзисторной и диодно-транзисторной логике [12, 29, 63, 66] обусловлена наличием входных и выходных связей, а также нелинейных элементов с релейной вольт-амперной характеристикой. Сложность задачи возрастает пропорционально числу выходных связей и логических состояний нелинейных элементов, которые по двоичному коду зависят от количества входов. Синтез схем в ДТЛ еще более трудоемок из-за неопределенности структуры для заданной логической функции, так как число решений определяется факториалом элементарных структур $\{f_k\}_0^l$ и зависит от электронного типа полупроводников и полярности их включения, вида элементарных функций f_k и топологии их электрических соединений. Неопределенность решения нелинейных электрических цепей приводит синтез заданной функции к эвристическим методам проб и ошибок, последовательному перебору необозримого банка вариантов, что нивелирует эффективность классических методов анализа до примитивных итераций, аналогичных поиску иголки в стогу сена.

Из мощного арсенала методов итерационного анализа для рационального проектирования комбинаторных схем целесообразны методы делителя напряжения (МДН) и токов (МТ) при синтеза ДЛ, ТЛ и ДТЛ, а для анализа и синтеза ИС – предпочтительны методы алгебры Буля и аналогии (см. рис. 1.2), разработанные для релейной логики (см. рис. 1.1). МДН и МТ систематизируют закономерности итерационного анализа методов сигнальных графов и узловых потенциалов, основанных на правилах Кирх-

гофа и законе Ома. В отличие от известных методов последовательного приближения, исследуемой функции F_i ставится в соответствие заданное решение F_0, принимаемое за нормированный эквивалент. При этом решение задачи инверсно итерационному анализу и сводится к поиску условий тождественности исследуемой и эквивалентной функции $F_i = F_0$.

В МДН (рис. 1.2) эквивалентному делителю напряжения с известной амплитудой U_{0j} на j-м эталонном сопротивлении приравнивается исследуемый делитель с напряжением U_i, а условия тождественности определяют по эквивалентному решению выбранной априори схемы последовательным упрощением схем замещения на каждом шаге итерации. При этом сложную логическую задачу дифференцируют на последовательность аналогичных решений, систематизированных в таблицу истинности по числу возможных состояний, определяемых числом входных переменных адресации нелинейных структур для соответствующего выхода схемы.

Метод токов (рис. 1.2) синтезирует таблицу истинности при адресном анализе логических состояний комбинаторной схемы за счет выявленных закономерностей МДН. При этом нелинейность вентильных элементов отождествляется с ключевыми логическими состояниями: прохождение тока через линейный элемент при замещении его проводником и отсутствие тока через разомкнутый логический контакт с высокоомным сопротивлением. Если в исследуемом состоянии входной ток появляется на выходе, то функция по анализируемому адресу тождественна логической единице, в противном случае при отсутствии тока ей присваивают инверсное значение – логического нуля.

Следовательно, рациональными для анализа схем комбинаторной логики являются интегрирующие закономерности классических расчетов методы итерации по эквивалентам: делителей напряжения и токов, основанные на системном поадресном синтезе условий тождественности эквиваленту исследуемой функции. Условия тождественности исследуемого состояния организуют итерационным анализом схем замещения, адекватных выбранной априори функции. При анализе схем диодной, транзисторной и диодно-транзисторной логики методы делителя напряжения и токов одновременно синтезируют таблицы истинности.

1.1.3 Методы алгебры Буля

Методы булевой алгебры [12, 29 – 34, 63 – 67], основанные на аналогии математических операторов алгебры и логики, целесообразно обобщить на аналогию структур и связей логической функции в образах науки и техники. При этом используют аналогию элементарных операторов дизъюнкции, конъюнкции и инверсии схемам логических элементов дизъюнктора (сложения, ИЛИ), конъюнктора (умножения, И) и инвертора (отрицания, НЕ) и их эквивалентам – таблицам истинности и временным диаграммам. Методы булевой алгебры предназначены для проектирования релейных логических функций в основных формах представления науки и техники, а также для анализа схем в комбинаторной и матричной логике в адресном пространстве, не превышающем размерность 4×4. Алгебра Буля объединяет неделимый комплекс операторов анализа и синтеза функций в релейной логике (см. рис. 1.1), разделенных с методической точки зрения на методы единиц и нулей (М10) и структурных формул (МСФ) в основных формах счисления НДФ и НКФ, ИЛИ-НЕ и И-НЕ (см. рис. 1.2).

Методы единиц и нулей (см. рис. 1.2) анализируют структурные схемы и формулы [12, 29, 30, 63, 67] в процессе синтеза таблиц истинности и временных диаграмм за счет систематизации произвольных состояний в организованное адресное пространство циклической программы и интервалы времени с последовательной адресацией за период. В М10 анализируют по произвольным адресам сложные структурные схемы и формулы за счет их дифференциации на элементарные структуры и связи для целенаправленной последовательности преобразований по аналогии с входов на выход функции. При анализе структурных формул веса $\xi_{ij} = \{\overline{0, 1}\}$ i-го терма j-го адреса распределяют по k-м позициям элементарных связей a_k. Выполняют с ними действия по логическим структурам элементарных операторов (умножение, сложение, отрицание). Численный результат адресованной функции $f_j = \{\overline{0, 1}\}$ размещают по j-му адресу выходного столбца таблицы истинности или на j-м интервале выходной временной диаграммы иллюстрируют соответствующий потенциал высокого (единичного) или низкого (нулевого) уровня. Анализ структурных схем М10 организуют аналогично алгоритму счисления структурных формул, но предварительно преобразуют структуры элементов (И, ИЛИ, НЕ) в аналогичные элементарные структуры логических операторов (умножения, сложения, отрицания).

По МСФ (см. рис. 1.2) синтезируют структурные формулы [12 – 17, 29, 30, 45, 67] при последовательном анализе структурных схем с выхода на их входы за счет замены структур и связей элементов аналогичными элементарными операторами и переменными. Это соответствует инверсному алгоритму

счисления М10 при анализе схем со входов на выходы за счет аналогичных преобразований логических элементов булевыми элементарными операторами, но без подстановки единиц и нулей для вычисления значения функции. Анализируют таблицы истинности МСФ при синтезе структурных формул единичной $F(1)$ или нулевой $F(0)$ функции по правилам НДФ или НКФ, соответствующих сумме j-х единичных функций $f_j(1)$ (произведению j-х нулевых функций $f_j(0)$), составленных из i произведений минтермов (или i сумм макстермов). Преобразование нормальных форм в $F(\bar{1})$ и $F(\bar{\&})$ выполняют по аксиомам и теоремам булевой логики. Для минимизации структурных формул используют также морфологические диаграммы, называемые картами Карно и Вейча.

Для организации информационной технологии проектирования микропроцессорных средств методы единиц и нулей, структурных формул дополняют эквивалентами схемо- и мнемотехники, математики и физики для исключения ошибок при синтезе таблиц истинности и временных диаграмм М10, структурных формул и схем МСФ. При этом анализируют одну из форм функции и одновременно синтезируют другой образ, который сопоставляют с эквивалентом функции в той же форме. Если синтезированный образ тождественен эквиваленту, то заключают, что анализируемая форма функции спроектирована верно, в противном случае необходимо перепроверить синтез исследуемой формы.

Следовательно, проектирование релейной логики в основных формах представления функций целесообразно проводить булевой алгеброй методами структурных формул и единиц и нулей в процессе дифференциации исследуемой формы на элементы и их замещением аналогами в заданной форме. Методы единиц и нулей служат для анализа структурных схем и формул в процессе синтеза таблиц истинности и временных диаграмм, а методы структурных формул необходимы при анализе структурных схем, таблиц истинности и временных диаграмм для синтеза структурных формул. Методы булевой алгебры развивают в информационную технологию оценкой исследуемой формы по тождественности эквиваленту синтезируемого образа функции в заданной форме представления.

1.1.4 Информационные методы

Информационная технология проектирования микропроцессорных средств представляет целенаправленную последовательность закономерных преобразований для организации согласованных компонент и форм представления функции. Логическим аппаратом анализа и синтеза компонент информационного обеспечения и основных форм представления функций является математика образов [15, 22], систематизирующая закономерности итерационного анализа по эквивалентам и булевой алгебры по аналогии в информационные принципы: аналогии и эквивалентности, инверсии и симметрии. Информационные принципы положены в основу соответствующих им методов: аналогии (МА) и эквивалентности (МЭ), инверсии (МИ) и симметрии (МС) (см. рис. 1.2).

Принцип аналогии систематизирует подобные закономерности различных отраслей знаний [15]. Подобие образов в различных областях науки и техники: топологии схем $F(R)$ и мнемонике программ $F(T)$, булевых операторах $F(\Phi)$ и метрологических оценках $F(\varepsilon)$, можно представить как

$$F(R) \approx F(T) \approx F(\Phi) \approx F(\varepsilon).$$

Методы аналогии идентифицируют компоненты микропроцессорных средств: аппаратные и метрологические средства, программное и математическое обеспечение – и формы представления функции: структурные схемы и формулы, таблицы истинности и временные диаграммы. Методы анализа и синтеза по аналогии ставят в соответствие метрику структурных схем адресации таблиц истинности, логику структурных формул метрологии временных диаграмм [24].

Принцип эквивалентности (см. рис. 1.2) отождествляет равноценные образы из различных разделов науки и техники для их систематизации в единую отрасль знаний [22, 24]. Если аналогия предполагает решение с известной степенью неопределенности, то эквивалентность исключает неопределенность и регламентирует тождественные преобразования:

$$F(R) = F(T) = F(\Phi) = F(\varepsilon).$$

Функция не зависит от эквивалентных преобразований. Образы функции в различных формах представления физики $F(\varepsilon)$ и математики $F(\Phi)$, мнемо- $F(T)$ и схемотехники $F(R)$ эквивалентны. Принцип эквивалентности регламентирует тождественность структур и связей, алгоритмов и моделей преобра-

зуемой функции в континууме пространство R – время T – функция Ф, так как образы из инвариантных признаков эквивалентны.

Принцип инверсии (двойственности) отражает дуальный характер информационных процессов и является следствием фундаментального закона единства и борьбы противоположностей [15 – 17, 22 – 24]. Информационное обеспечение, реализующее эти процессы, без дополнительных ресурсов осуществляет инверсные преобразования функции. Принцип инверсии (рис. 1.2) постулирует нормируемую меру $\{\varepsilon_R, \varepsilon_T, \varepsilon_\Phi\}$, тождественную векторному произведению функции F на ее инверсное F^{-1} отображение

$$F\,F^{-1} = E.$$

В общем случае мера E представляет собой матрицу из нормированных значений, а в частном – константу, принимаемую за эквивалент. Булевые формы НКФ и НДФ связаны инверсными преобразованиями теоремы Деморгана, основанной на аксиомах логики, что позволяет не только минимизировать оригинал, но и синтезировать функцию в базисах И-НЕ и ИЛИ-НЕ. Дуальность и неделимость анализа и синтеза – основа схемо- и мнемотехники, систематизирующая принципы аналогии и эквивалентности по вектору упорядочивания информации к симметричным решениям.

Принципы симметрии (см. рис. 1.2) нормируют структуру и параметры исследуемой функции в пространственно-временном континууме пропорционально мерам и эквивалентно критерию эффективности [17, 22, 24]. С позиций метрологии симметричные решения соразмерны в инерционной системе отсчета. Симметрия регламентирует инвариантность структуры функции относительно ее преобразований с априори заданной погрешностью $\varepsilon_0 \geq \varepsilon(\varepsilon_R, \varepsilon_T, \varepsilon_\Phi)$, обусловленной неточностью преобразований в математике ε_Φ, мнемо- ε_T и схемотехнике ε_R. Согласованность форм представления функции и компонент микропроцессорных средств достигается методами симметрии по критерию метрологической эффективности:

$$\varepsilon_R = \varepsilon_T = \varepsilon_\Phi = \varepsilon \leq \varepsilon_0.$$

Для оценки эффективности рационально использование приведенной погрешности, усредненной по известным критериям оценки [16, 22].

Следовательно, информационные принципы развиваются по вектору метрологической эффективности по аналогии в науке и технике до эквивалентности в систематизированной отрасли знаний, от инверсии с нормируемыми мерами до симметрии с точностью меры по критерию оценки эффективности. Информационные методы проектируют формы представления функций и компоненты микропроцессорных средств, согласованные между собой по критерию эффективности с погрешностью нормируемых мер. Информационные методы систематизируют анализ и синтез матричных БИС микропроцессорных средств в информационную технологию проектирования согласованных решений с гибкой архитектурой, информативным математическим обеспечением и эффективными метрологическими средствами.

1.1.5 Формы логических функций

Неделимый комплекс основных форм представления логических функций позволяет всесторонне исследовать структурные схемы и формулы, таблицы и диаграммы (табл. 1.1) на различных уровнях иерархии комбинаторной, релейной и матричной логики. Иерархия структурных схем соответствует базисам развития микроэлектроники и включает полупроводниковые приборы (ПП) и интегральные схемы, которые делят на малые (ИС), средние (СИС) и большие (БИС). Во второй строке (табл. 1.1) мнемоформы на уровне ПП и СИС систематизируют на таблицы истинности (ТИ) и состояния (ТС), а для СИС и БИС функцию отражают в векторных таблицах мультиплексора (ТМ) и кодов операций (ТК) соответственно для комбинационных и последовательностных структур. По аксиомам и алгоритмам булевой алгебры моделируют структурные формулы (и схемы по аналогии) в НДФ $F(1)$ и НКФ $F(0)$, а также базисах ИЛИ-НЕ $F(\bar{1})$ и И-НЕ $F(\overline{\&})$. Иллюстрируют функцию в адресном пространстве временной синхронизации векторные (в статике) и временные (в динамике) диаграммы [13, 25, 29, 66], карты Карно и Венна [1, 30, 37], которые служат также для минимизации структурных схем и формул.

Иерархия цифровых схем по упорядоченности логики и базисам микроэлектроники представлена в табл. 1.2. Комбинаторные схемы на ПП усложняются от диодной через транзисторную к диодно-транзисторной логике и выпускаются промышленностью в виде сборок ДЛ, ТЛ

1.1 Формы функций

Схемы	Полупроводниковые приборы (ПП)	Интегральные схемы (ИС)	Средние интегральные схемы (СИС)	Большие интегральные схемы (БИС)
Таблицы	Таблица истинности (ТИ)	Таблица состояния (ТС)	Векторная таблица мультиплексора (ТМ)	Таблица кодов операций (ТК)
Формулы	Нормальная дизъюнктивная форма $F(1)$ (НДФ)	Нормальная конъюнктивная форма $F(0)$ (НКФ)	Базис ИЛИ-НЕ $F(\bar{1})$ (ИЛИ/НЕ)	Базис И-НЕ $F(\bar{\&})$ (И/НЕ)
Диаграм-	Векторные	Карно	Венна	Временные

1.2 Цифровые схемы

Комбинатор-	Диодная логика (ДЛ)	Транзисторная логика (ТЛ)	Диодно-транзисторная логика (ДТЛ)	Интегральные схемы (ИС)
Релейные	ЭЛЕКТРОСТАТИЧЕСКИЕ реле (ЭСР)	Электромагнитные реле (ЭМР)	Тиристорные реле (ТР)	Транзисторные реле (ЭР)
Матричные	Диодные логические матрицы (ДЛМ)	Программируемые дешифраторы (ПД)	Программируемые мультиплексоры (ПМ)	Программируемые логические матрицы (ПЛМ)
Базисы	Логические элементы (И, ИЛИ, НЕ)	Триггеры: статические *T*, динамические *TT*	Дешифраторы *DC*, мультиплексоры *MUX*	Счетчики *CT*, регистры *RG*

и ДТЛ. Развитием ПП в планарной топологии ДТЛ являются малые интегральные схемы комбинационного и последовательностного типа. В базисе ИС первые отражены логическими элементами (И, ИЛИ, НЕ), а последние представлены элементами памяти, в виде статических *T* и динамических *TT*-триггеров. На уровне базиса СИС комбинационные логические элементы формируются в дешифраторы *DC* и мультиплексоры *MUX*, а элементы памяти интегрируются в счетчики *CT* и регистры *RG* запоминающих устройств (см. табл. 1.2, строка 4).

Релейные схемы в безадресной топологии расширяют номенклатуру комбинаторной логики, а систематизированные в матричные структуры формируют релейную логику. Различают электрические и

электронные реле, поэтому релейные матрицы выполняют на электростатических (ЭСР) и электромагнитных (ЭМР), тиристорных (ТР) и транзисторных (ЭР) реле (см. табл. 1.2, строка 2). По энергетической мощности релейную логику делят на информационную для обработки малоточного (нано-, микроуровня) сигнала и энергетическую для преобразования высоковольтных (кило-, мегауровня) напряжений. Основой микропроцессорных средств является матричная логика БИС, которые по жесткости структуры и гибкости программы делят на диодные логические (ДЛМ) и программируемые (ПЛМ) матрицы, программируемые дешифраторы (ПД) и мультиплексоры (ПМ). ДЛМ выпускают в виде заказных БИС, запрограммированных на заводе-изготовителе, а ПЛМ предназначены для программирования пользователем всего адресного пространства. В ПД и ПМ завод-изготовитель прожигает соответственно матрицы И/НЕ-И и ИЛИ, а пользователю для программирования доступны адресные пространства матриц ИЛИ и И/НЕ-И.

Следовательно, формы представления логических функций отражают компоненты микропроцессорных средств в виде неделимого комплекса структурных схем и формул, таблиц и диаграмм комбинаторной, релейной и матричной логики энергетических и информационных преобразователей на реле и полупроводниковых приборах, ИС, СИС и БИС.

1.2 ОСНОВЫ ЛОГИЧЕСКИХ ОПЕРАТОРОВ

Представление цифровых схем в форме функции в функциональных $F(\Phi)$ координатах организуют логическим исчислением булевой алгебры, основанной на трех элементарных операторах: дизъюнкции (логического сложения), конъюнкции (логического умножения) и инверсии (логического отрицания). Первый раздел математической логики предложен в середине XIX века ирландским математиком Джорджем Булем и назван его именем. В основу своей алгебры Буль предложил аналогию между алгеброй и логикой [40, С. 26]. Идея применения алгебры Буля в технике впервые была высказана в России известным физиком П. Эренфестом (1910 г.), а математически доказана русским ученым В.И. Шестаковым (1936 г.) и американским инженером К.Э. Шенноном (1938 г.) в теории и практике контактных (релейных) схем.

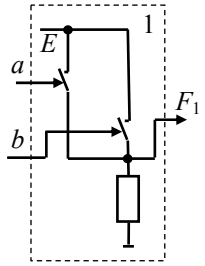
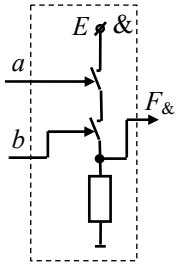
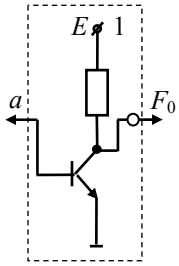
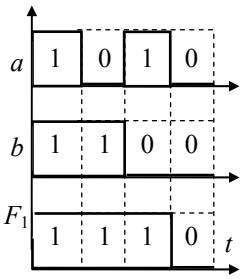
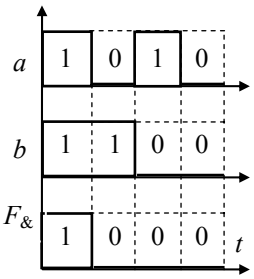
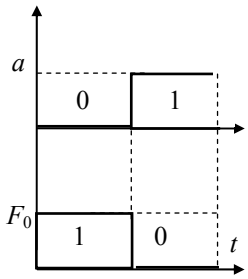
Основой булевой алгебры в электронике является эквивалентность представления логических функций в основных формах схемо- и мнемотехники, математики и физики на уровне топологии схем $F(R)$ и мнемоники таблиц $F(T)$, математических формул $F(\Phi)$ и электрических диаграмм $F(\varepsilon)$. Основные формы представления в технике и науке логических функций сложения, умножения и отрицания систематизированы в табл. 1.3 в виде аксиом, алгоритмов и структур для синтеза и анализа цифровых интегральных схем по информационной технологии проектирования микропроцессорных средств [13].

1.2.1 Логические элементы

Логические элементы [40] являются азбукой микропроцессорных средств, тремя китами которой являются элементарные логические функции [29]: ИЛИ (логическое сложение, дизъюнкция), И (логическое умножение, конъюнкция), НЕ (логическое отрицание, инверсия). На уровне информационного обеспечения [13] компонентами микропроцессорных средств служат аппаратные $F(R)$ и метрологические $F(\varepsilon)$ средства, программное $F(T)$ и математическое $F(\Phi)$ обеспечение. Основой компонент микропроцессорных средств на уровне цифрового преобразования сигнала является также комплексное представление функций ИЛИ, И, НЕ в основных формах схемо- $F(R)$ и мнемотехники $F(T)$, образах математики $F(\Phi)$ и физики $F(\varepsilon)$. Элементарные функции в электронике представляют структурными схемами в координатах пространства R, а в программировании – таблицами истинности, систематизирующими во времени T произвольные состояния. В математике функции F описывают структурными формулами в координатах логического Ф пространства, а оценку ε эффективности функции иллюстрируют семейством временных диаграмм, отражающим физические закономерности (табл. 1.3).

В топологии электроники логические функции ИЛИ, И, НЕ адекватны параллельному, последовательному и смешанному соединению электрических проводников (см. сегмент $F(R)$) и по стандартам схемотехники логические элементы сложения (дизъюнктор), умножения (конъюнктор), отрицания (инвертор) представляют структурными схемами (топологическими рисунками) с регламентированными обозначениями «1», «&», «○». Для функций F двух переменных $\{a, b\}$ физическим состояниям «включено» и «выключено» сопоставляют логические термы «1» (единица) и «0» (нуль). При выключенных ключах

1.3 Логические элементы

	1 ИЛИ	2 И	3 НЕ
Схемы $F(R)$			
Таблицы $F(T)$	a | b | F_1 0 | 0 | 0 1 | 0 | 1 0 | 1 | 1 1 | 1 | 1	a | b | $F_\&$ 0 | 0 | 0 1 | 0 | 0 0 | 1 | 0 1 | 1 | 1	a | F_0 0 | 1 1 | 0
Формулы $F(\Phi)$	$\boldsymbol{F_1 = a + b}$ $0 + 0 = 0$ $1 + 0 = 1$ $0 + 1 = 1$ $1 + 1 = 1$	$\boldsymbol{F_\& = ab}$ $0 \cdot 0 = 0$ $1 \cdot 0 = 0$ $0 \cdot 1 = 0$ $1 \cdot 1 = 1$	$\boldsymbol{F_0 = \bar{a}}$ $\bar{0} = 1$ $\bar{1} = 0$
Диаграммы $F(\varepsilon)$			

$\{a, b\} = \{0, 0\}$ через резистор ток от источника E не течет, на выходах схем ИЛИ и И в исходном состоянии присутствует нулевой потенциал земли, что соответствует логическому нулю: $F_1 = 0$ и $F_\& = 0$ (см. $1F(R)$, $2F(R)$). На выходе $F_1 = 1$, если включить ИЛИ a, ИЛИ b ключи, что обусловлено параллельным электрическим соединением, организующим элемент ИЛИ (дизъюнктор). Инверсно логическому сложению на выходе $F_\&$ конъюнктора (элемента И) появится потенциал E тогда и только тогда, если замкнуть ключи И a, И b за счет их последовательного соединения, формирующего элемент И (см. $2F(R)$). Дизъюнктор и конъюнктор реализуют в электротехнике на коммутаторах и реле, а в электронике – на диодах и транзисторах при параллельном и последовательном соединении повторителей тока или напряжения. В отличие от них инвертор (см. $3F(R)$) синтезируют в транзисторной логике по схеме с общим эмиттером (истоком), преобразующей входной сигнал a на выходе F со сдвигом по фазе на 180° или изменением входного потенциала инверсно на противоположный. При низком потенциале на базе транзистор n-p-n закрыт, имеет бесконечно высокое сопротивление, а на коллекторе присутствует потенциал E высокого уровня, что соответствует $F = 1$. Если на базу подать потенциал E логической единицы $a = 1$, то открытый транзистор из-за бесконечно малого сопротивления нулевой потенциал земли

подключает к выходу, при этом $F = 0$. В электротехнике инвертор проектируют также по схеме смешанного соединения на электромагнитных реле с нормально замкнутыми контактами.

Логику функционирования цифровых элементов поясняют на программном уровне $F(T)$ алгоритмы, систематизирующие произвольные физические в логические состояния таблиц истинности. Таблицы регламентируют логические функции сложения ИЛИ (см. $1F(T)$) дизъюнктора, умножения И (см. $2F(T)$) конъюнктора и отрицания НЕ (см. $3F(T)$) инвертора. При отрицании прямым значениям $a = \{0, 1\}$ соответствуют инверсные термы $F = \{1, 0\}$. Таблица конъюнктора (см. $2F(T)$) аналогична арифметическому умножению, а таблица дизъюнкции (см. $1F(T)$) постулирует, что $F = 1$, если ИЛИ a, ИЛИ b не равны нулю.

Технике схем $F(R)$ и таблиц $F(T)$ ставят в соответствие математические образы – структурные формулы $F(\Phi)$ операторов F_1 – ИЛИ, $F_\&$ – И, F_0 – НЕ:

$$F_1 = a + b;$$

$$F_\& = a \cdot b;$$

$$F_0 = \overline{a}.$$

На примере возможных комбинаций переменных $\{a, b\} = \overline{\{0, 1\}}$ в строке $F(\Phi)$ приведены результаты логического сложения (см. $1F(T)$), умножения (см. $2F(T)$) и инверсии (см. $3F(T)$). Тождества конъюнкции аналогичны арифметическим перемножениям чисел нуль и один. Подобны арифметическим тождествам сложения структурная формула дизъюнкции и логические уравнения сложения, однако результатом сложения единиц в булевой алгебре является также единица из-за унитарности логического пространства. В отличие от вычитания в арифметике оператор инверсии преобразует прямое значение – в противоположное ему отрицание, так как в алгебре Буля нуль является отрицанием единицы, а единица – инверсией нуля. Аналогично арифметическим операторам сложения и умножения обозначают знаками «+» и «×» операции ИЛИ (дизъюнкции) и И (конъюнкции).

Образами представления функций в физике для оценки метрологической эффективности схем, таблиц и формул служат семейства временных диаграмм $F(\varepsilon)$ функций ИЛИ, И, НЕ. Временные диаграммы иллюстрируют логические состояния $\{0, 1\}$в виде электрических сигналов с потенциалами низкого (нулевого) и высокого (единичного) уровня при положительной логике. В отрицательной логике потенциалам низкого и высокого уровня ставят в соответствие единицу и нуль логического пространства. Семейства временных диаграмм $F(\varepsilon)$ однозначно определяют логические преобразования элементов ИЛИ, И, НЕ и служат эквивалентами для оценки эффективности схем $F(R)$, программ $F(T)$ и формул $F(\Phi)$ в информационной технологии проектирования микропроцессорных средств.

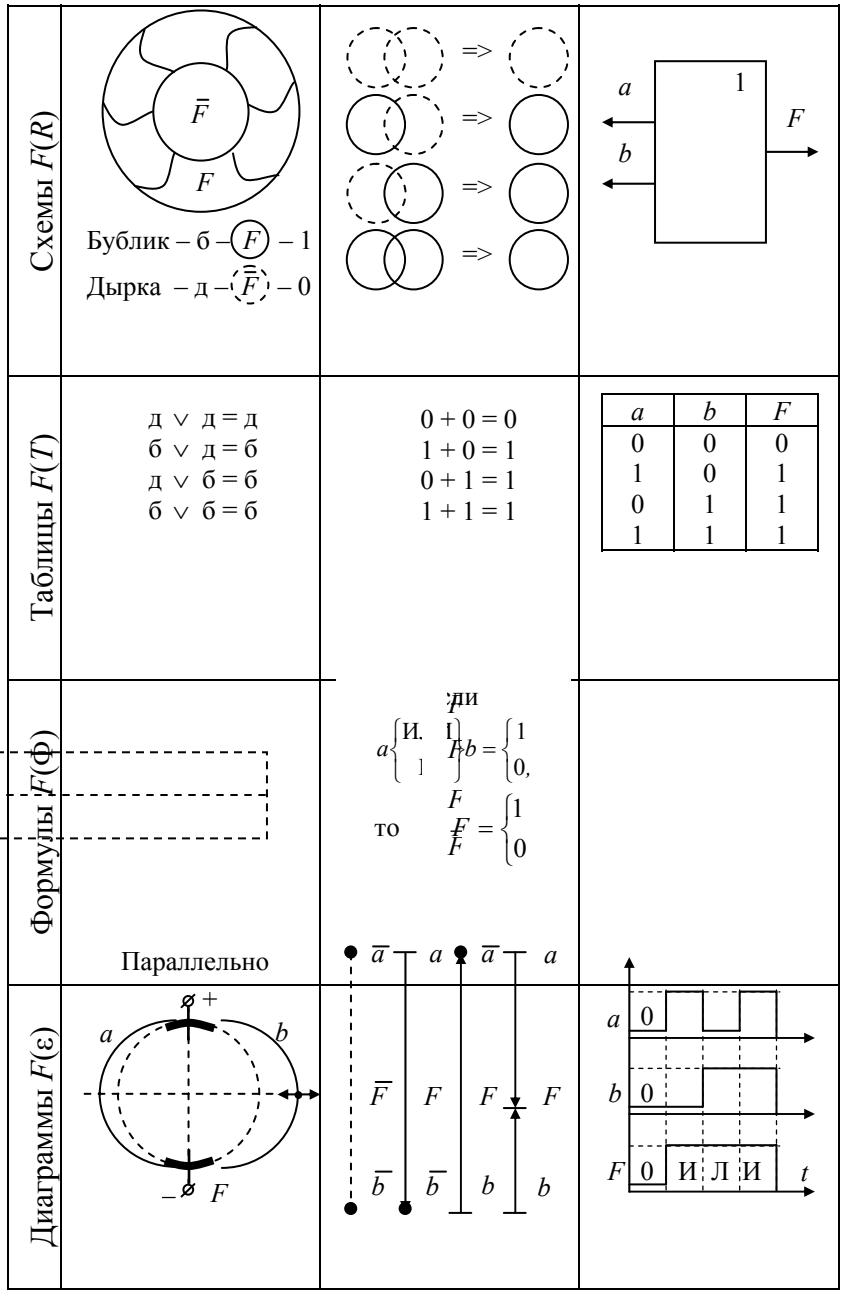
Таким образом, азбукой микропроцессорных средств являются элементы ИЛИ, И, НЕ, реализующие элементарные булевые функции дизъюнкции, конъюнкции, инверсии в унитарном логическом пространстве единиц и нулей по аналогии с алгеброй. Формами представления функций информационных процессов является неделимый комплекс структурных схем и формул, таблиц истинности и временных диаграмм, представляющих элементарный уровень аппаратных и метрологических средств, программного и математического обеспечения микропроцессорных средств. Элементарные функции логического сложения, умножения и инверсии в топологии электроники адекватны параллельному, последовательному и смешанному соединению элементов и алгоритму их функционирования.

1.2.2 Дизъюнкция

Интерпретация логического сложения (дизъюнкции, операции ИЛИ) удобна на примере модели логического пространства (табл. 1.4, сегмент $1F(R)$), включающего объединение функции $F = 1$ и ее отрица-

1.4 Дизъюнкция

	1 Аксиомы	2 Алгоритмы	3 Структуры



ние $\overline{F} = 0$, аналогичных «бублику» (б) и «дырке» (д). Аксиому сложения поясняет алгоритм объединения (там же, $2F(R)$) схем бублика « » и дырки « »). Из четырех возможных сочетаний объединение дырок приводит к дырке и к бублику при совмещении бублика с дыркой или бубликом. Алгоритм дизъюнкции систематизирован в виде мнемонических таблиц $F(T)$ высказываний (сегмент $1F(T)$), числовых тождеств ($2F(T)$) и истинности ($3F(T)$).

Однозначность таблиц истинности регламентируется правилами ее синтеза, число P которых при произвольной адресации определяется суммой факториалов столбцов S и строк $C = 2^S$ входной таблицы

$$P = S! + 2^S!.$$

Например, для двух входных переменных число произвольных правил, и соответственно, таблиц $P(2) = 26$, так как $2! + 2^2! = 2 + 24$, а для трех переменных возможных комбинаций более сорока тысяч, так как $P(3) = 3! + 8!$. Произвольное число правил синтеза затрудняет проектирование таблиц истинности, при проектировании логических функций исключает систематизацию для выявления закономерностей информационных технологий и не позволяет введение технических стандартов в мнемонику программирования.

Повышает технологичность единое простое запоминающееся правило проектирования таблиц истинности по стандартному алгоритму. Число столбцов S входной таблицы дешифратора соответствует числу входных переменных, а количество строк таблицы определяется, как правило, двоичным кодом $C = 2^S$ по числу состояний. Рационально адресовать столбцы слева направо от младшего до старшего разряда. Синтезируют таблицу дешифратора сверху вниз, начиная с нулей и заканчивая всеми единица-

ми. В младшем левом столбце последовательно записывают 1 и 0 с разрядкой в один раз, во втором столбце – с разрядкой в два раза, а в i-м столбце – с разрядкой в 2^{i-1} раза. При этом будут учтены все возможные состояния и исключается пропуск хотя бы одной комбинации, так как j-я строка таблицы соответствует числу в двоичном коде $N_2 - 1$, что удобно при анализе таблицы. Например, для двух переменных $\{a, b\}$ первая строка (см. $3F(T)$) формирует число $\{0, 0\}_2$, т.е. нуль, а третья строка отражает число $\{0, 1\}_2 = 2_{10}$ третьего состояния дизъюнкции. Возрастающий код от нуля до трех информирует о безошибочном синтезе и анализе, что соответствует правильному проектированию таблицы входов. Выходная таблица мультиплексора заполняется по необходимым логическим операторам, в приведенном примере – по функции логического сложения.

Информационная технология требует проектирования функции во всех формах представления техники и науки согласно информационной модели $F(R, T, \Phi, \varepsilon)$ [22] в топологии схем $F(R)$ и мнемонике программ $F(T)$, в образах математики $F(\Phi)$ и закономерностях физики $F(\varepsilon)$. Комплексное представление позволяет согласовать информационные процессы и формы представления функций, технологично проектировать аппаратные и метрологические средства, математическое и программное обеспечение микропроцессорных средств. Основные формы представления дизъюнкции отражают структуры (см. табл. 1.4, сектор 3) в виде структурной схемы ($3F(R)$) и таблицы истинности ($3F(T)$) в технике, структурных формул ($3F(\Phi)$) и временных диаграмм ($3F(\varepsilon)$) в науке.

Структурная схема дизъюнктора (элемента логического сложения, ИЛИ, 1) согласно стандартам [24, 66] представляется моделью «черного ящика» прямоугольной формы (структурой) в декартовой системе координат $R(X, Y)$. В верхнем правом углу структуры указывают функцию дизъюнкции «1», обозначающую логический элемент «ИЛИ». Связи отражают входы и выходы электрических сигналов, поступающих в структуру слева направо (или сверху-вниз) по электрическим проводникам. Изображают связи прямыми линиями, параллельно или перпендикулярно ординатам $R(X, Y)$ и обозначают соответствующими переменными. Входные связи адресуют строчными буквами, а выходные – прописными буквами латинского алфавита, соответствующими его началу и концу (см. $3F(R)$).

В функциональных координатах $F(\Phi)$ математики дизъюнкцию представляют структурными формулами в основных четырех формах $F(0)$ и $F(1)$, $F(\overline{\&})$ и $F(\overline{1})$. Нормальные конъюнктивная (НКФ) и дизъюнктивная (НДФ) формы описывают логическую функцию дизъюнкции в инверсных операторах нулевого F(0) и единичного $F(1)$ пространства:

$$\begin{cases} F(0) = a + b; \\ F(1) = a\bar{b} + b, \end{cases}$$

что соответствует произведению макстермов $F(0) = \prod_{j=0}^{m-1} f_j(0)$ и сумме минтермов $F(1) = \sum_{j=0}^{m-1} f_j(1)$. Дизъюнкция обозначается операторами объединения $\vee$ или сложения +, конъюнкция – операторами включения $\wedge$ или умножения «×» (*, •), а инверсия – отрицанием: чертой над буквой. Полнотой характеризуются базисы И-НЕ ($\overline{\&}$), а также ИЛИ-НЕ ($\overline{1}$) интегральных схем, реализующие дизъюнкцию соответственно в операторах умножения и сложения инверсных образов:

$$\begin{matrix} F(\overline{\&}) = \overline{\overline{a}\,\overline{b}}; \\ F(\overline{1}) = \overline{a + b}. \end{matrix}.$$

Структурные формулы $F(\overline{\&})$ и $F(\overline{1})$ являются производными из НКФ и НДФ за счет минимизаций $F(0)$ и $F(1)$ по логическим правилам, формулируемым алгоритмами и аксиомами алгебры Буля.

Алгоритм дизъюнкции выявляет закономерность между переменными $\{a, b\}$ и результатом F, очевидную из таблицы истинности ($3F(T)$). Только при $a = b = 0$ в соответствии с первой строкой таблицы истинности $F = 0$, что подтверждает основное правило логического сложения. Если a ИЛИ b соответствует единице, то объединение F равно единице. Алгоритм функции ИЛИ объединяет два условия в виде формулы ($2F(\Phi)$):

$$\text{если} \quad a \begin{Bmatrix} \text{ИЛИ} \\ \text{И} \end{Bmatrix} b = \begin{cases} 1, \\ 0, \end{cases} \quad \text{то} \quad F = \begin{cases} 1 \\ 0. \end{cases}$$

Следовательно, если *a* или *b* – единицы, то их сумма *F* – единица, в противном случае дизъюнкция *F* – нуль. Выявленная в алгоритме закономерность постулируется аксиомой (1*F*(Ф)):

$$a + \bar{a} = 1 .$$

Аксиома дизъюнкции систематизирует два тождества для переменной $a = \{\overline{0, 1}\}$. Если *a* = 0, то отрицание $\bar{a} = 1$, а их сумма равна единице. Аналогичный результат получают, если *a* = 1, а инверсия $\bar{a} = 0$ (см. сегмент 1*F*(Ф)). Учитывая унитарность логического пространства, для дизъюнкции справедливы и другие аксиомы: *a* + *a* = *a* или $\bar{a} + \bar{a} = \bar{a}$. Аксиомы функции ИЛИ необходимы для булевых преобразований при минимизации структурных формул, а также для моделирования функций в НКФ и НДФ, базисах И-НЕ и ИЛИ-НЕ. Следует отметить, что в отличие от классических методов итерационного анализа комбинаторных схем, основанных на алгебре Буля [1, 29, 67], в информационной технологии проектирования микропроцессорных средств [12] аппарат булевой алгебры не эффективен при анализе матриц размерностью более чем 4 × 4.

Структурным схеме (3*F*(*R*)) и формулам (3*F*(Ф)) в соответствии с таблицей истинности (3*F*(*T*)) дизъюнкцию представляют семейством временных диаграмм (3*F*(*ε*)). По оси абцисс – времени *t* иллюстрируют последовательность состояний слева-направо за период (цикл) *T* в соответствии с программой таблицы истинности, включающей циклическую последовательность кодов строк сверху-вниз. Временные диаграммы синтезируют транспонированием (поворот на 90° по часовой стрелке относительно правого нижнего угла и зеркальное отражение) таблицы истинности и аналогичной заменой логических термов нуль и единица соответствующими потенциалами амплитуды низкого (нулевого) и высокого (единичного) уровня. При этом первой строке {0, 0, 0} таблицы истинности (3Ф(*T*)) однозначен столбец первого состояния с нулевыми потенциалами (3*F*(ε)). Следует подчеркнуть, что из всех основных форм представления, имеющих множественный характер, только временная диаграмма инвариантно отражает физику информационного процесса в единственной форме и может служить эквивалентной мерой анализа и синтеза других форм представления функции. Для стандартизации методов проектирования, минимизирующих число циклов итерационного анализа, также необходима только единственная таблица истинности, эквивалентная семейству временных диаграмм.

Мнемоническими примерами иллюстрации логического сложения служат физическая и геометрическая интерпретация параллельных соединений. Диаграмма 1*F*(ε) показывает отсутствие тока в узле «–», когда сегменты *a* и *b* не замыкают его с узлом «+», при этом *F* = 0. Если подключить к узлам сегмент *a* или *b*, то через них потечет ток, соответственно *F* = 1. Аксиоматичен алгоритм деления длины *F* отрезка на элементы {*a*, *b*}={$\overline{0, 1}$}, см. диаграмму 2*F*(ε). Если элементы *a* и *b* нулевые, то и длина *F* = 0, но если или *a* или *b* больше нуля или единичные элементы, то нормированная длина отрезка – единица *F* = 1.

Таким образом, дизъюнкция представляется по информационной модели в виде неделимого комплекса: структурных схемы ИЛИ и формулы логического сложения, имеющих множественную форму в базисах НДФ и НКФ, И-НЕ и ИЛИ-НЕ; стандартной таблицы истинности и инвариантного семейства временных диаграмм – эквивалентов согласованных информационных процессов, организуемых по закономерностям технологии проектирования микропроцессорных средств. Булевый оператор дизъюнкции аксиоматичен алгоритму параллельного объединения элементов функции в топологии схемо- и адресации мнемотехники, регламентируемых стандартами и единой системой конструкторской документации (ЕСКД). Аксиомы и алгоритмы алгебры Буля моделируют и минимизируют структурные формулы дизъюнкции для итерационного анализа комбинаторной и релейной логики, причем булевы операторы не эффективны в информационной технологии проектирования микропроцессорных средств, реализуемых на программируемых матрицах многомерного адресного пространства.

1.2.3 Конъюнкция

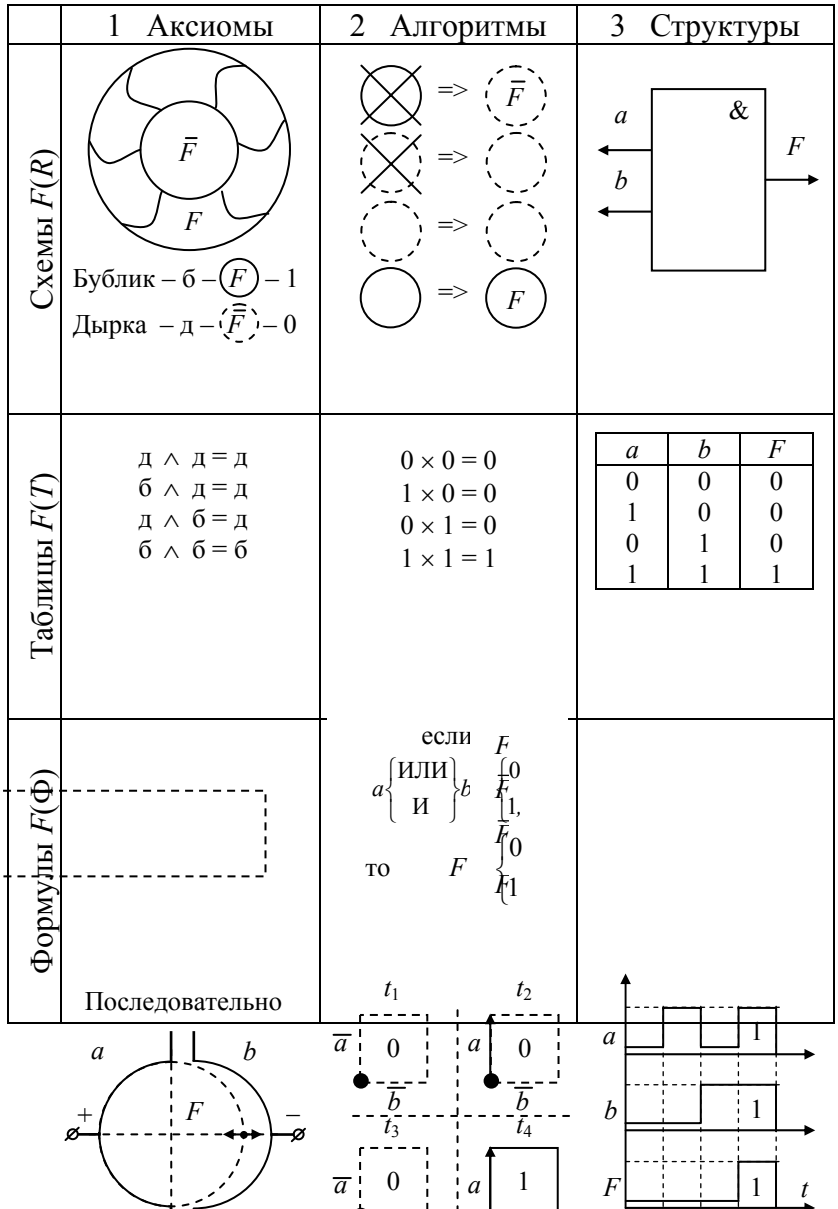
Функцию логического умножения – конъюнкцию несложно проиллюстрировать в координатах логического пространства *F* = 1 (бублика), включающего отрицание $\bar{F} = 0$ (дырку), см. табл. 1.5, сегмент 1*F*(*R*). Алгоритм конъюнкции (см. 2*F*(*R*)) схемотехники показывает, что *F* = 1, если включить в логическое пространство бублик , и оно будет пустым *F* = 0 при включении в него только дырки или при исключении из него дырки или бублика . Приведенный алгоритм моделируют переменными оператором «включить-исключить» и признаками полноты пространства «дырка-бублик», однако его можно интерпретировать более просто и наглядно постоянным оператором «умножить» по аналогии с ал-

гебраическим исчислением. При этом справедливы аксиомы (см. 1$F(T)$): если дырку (д) или бублик (б) перемножить на дырку, то результатом будет дырка, но если перемножить бублик на бублик, то логическое пространство будет включать бублик. Аксиомы конъюнкции алгоритмизированы числовыми тождествами (2$F(T)$), из которых следует, что только при перемножении единиц – результат единица, а при умножении на нуль – тождество равно нулю. Мнемонические правила конъюнкции систематизированы в таблице истинности (3$F(T)$) во временных координатах $F(T)$ мнемотехники. Входная таблица дешифратора спроектирована стандартным образом в двоичном коде, возрастающем по линейному закону от нуля {0 0}до трех {1 1}. В таблице мультиплексора единица присутствует при условии, если И a, И b равны единице, в противном случае $F = 0$. Поэтому конъюнкцию также называют «функцией И» или оператором логического умножения, а условно обозначают знаком «&» или «И».

Таблица истинности получила название от систематизации истинных (1) и ложных (0) логических суждений [40], утверждающих, что истинному заключению $F = 1$ соответствуют единогласные истинные суждения $a_i = a_{i+1} = 1$ для $i = \overline{0, n-1}$, а при наличии хотя бы одного ложного суждения $a_j = 0$ истинность заключения ставится под сомнение и принимается ложным $F = 0$. В мнемотехнике микропроцессорных средств таблица истинности служит программой анализа и синтеза алгоритма информационного процесса, элементарными из которых являются логические функции конъюнкции, дизъюнкции и инверсии.

В схемотехнике интегральных схем конъюнктивной таблице истинности (3$F(T)$) ставят в соответствие структурную схему (3$F(R)$), аналогичную дизъюнктору (см. табл. 1.4, 3$F(R)$), но вместо «1» в правом верхнем углу помещают символ «&». Структурная схема по стандартам и ЕСКД схемо- и мнемотехники, также как и конъюнктивная таблица изображает логическую функцию в декартовой системе координат слева-направо или сверху-вниз, а для читабельности и явного понимания функции входные и выходные связи обозначают прямыми линиями с названием переменных a, b, F над ними. Число входов и выходов структурной схемы (3$F(R)$) однозначно соответствует числу

1.5 Конъюнкция



a	b	F
0	0	0
1	0	0
0	1	0
1	1	1

Диаграммы $F(\varepsilon)$			

столбцов таблиц дешифратора и мультиплексора, составляющих таблицу истинности. Согласно стандартам, структуры и связи отображают в черном цвете, причем расстояние между соседними линиями должно быть кратно расстоянию $c \geq 5$, так как минимальный шрифт равен 3 мм. Высоту h структуры определяют максимальное число n связей по входу или выходу $h = (n + 1)c$, ширина g выбирается пропорционально высоте $h = pg$, где $p = 1, 2, 3 \ldots$

Таблице и схеме конъюнктора в математических образах $F(\Phi)$ ставятся в соответствие структурные формулы (см. $3F(\Phi)$), алгоритмы (см. $2F(\Phi)$) и аксиомы (см. $1F(\Phi)$). Структурные формулы синтезируют, как правило, по таблице истинности в основных операторах алгебры Буля: инверсии, дизъюнкции и конъюнкции, – последний из которых в НДФ регламентирует логическое умножение (включение)

$$F(1) = ab.$$

Функцию включения (оператор «И») обозначают знаком «&», а по аналогии с алгеброй и арифметикой – символами умножения: × и *, • и ( ) – пропуском между буквами минтермов структурной формулы. Производными от НДФ являются структуры в НКФ – $F(0)$, базисах И/НЕ – $F(\overline{\&})$ и ИЛИ/НЕ – $F(\bar{1})$, эквивалентные логическому умножению $F(1)$:

$$\begin{cases} \overline{F}(0) = \bar{b} + \bar{a}b; \\ \overline{F}(\&) = \overline{ab}; \\ F(\bar{1}) = \overline{\bar{a} + \bar{b}}. \end{cases}$$

Правило синтеза логического умножения систематизирует алгоритм функции И

$$\text{если} \quad a \begin{Bmatrix} \text{ИЛИ} \\ \text{И} \end{Bmatrix} b = \begin{cases} 0 \\ 1 \end{cases}, \quad \text{то} \quad F = \begin{cases} 0 \\ 1 \end{cases}.$$

Алгоритм конъюнкции утверждает, что суждение истинно $F = 1$ тогда и только тогда, если и a и b, исходные включения, истинны: $a = b = 1$. При хотя бы одном ложном суждении: a или b равно нулю, суждение считается недостоверным – равным нулю.

Систематизированная в алгоритме закономерность (см. $2F(\Phi)$) постулируется аксиомой логического умножения

$$a * \bar{a} = 0 .$$

Аксиома конъюнкции $1F(\Phi)$ включает два тождества для переменной $a = \overline{\{0, 1\}}$ и ее инверсии $\bar{a} = \overline{\{1, 0\}}$. Так как прямая или инверсная величины равны нулю, то их произведение – также нуль. В отличие от алгебры, оперирующей степенными функциями, где $a \cdot a = a^2$, в алгебре Буля из-за унитарности логического пространства степень – постоянная величина, поэтому справедливы соотношения $a \cdot a = a$ и $\bar{a} \cdot \bar{a} = \bar{a}$. Аксиомы булевой алгебры служат для синтеза и анализа, преобразования и минимизации структурных формул при доказательстве эквивалентности различных форм представления функции.

Семейство временных диаграмм однозначно отражает физику информационного процесса конъюнкции двух сигналов a, b, принимающих значение низкого (нулевого) и высокого (единичного) уровня потенциала. Временные диаграммы $3F(\varepsilon)$ адекватно иллюстрируют программу состояний таблицы истинности в виде последовательности из четырех тактов t_i, где $i = \overline{1, 4}$ за период T. Из диаграммы следует, что потенциал высокого уровня $F = 1$ формируется только при условии включения единичных сигналов

во время t_4, в противном случае нулевой потенциал заземляет до нуля $F = 0$ единичный потенциал высокого уровня (см. интервалы $t_1 - t_3$ $3F(\varepsilon)$).

Наглядно интерпретирует логическое умножение последовательное включение элементов (см. $1F(\varepsilon)$) и векторная диаграмма единичной площади (см. $2F(\varepsilon)$). Из четырех произвольных состояний t_i, $i = \overline{1,\ 4}$ видно (см. $2F(\varepsilon)$), что площадь логического пространства равна единице тогда и только тогда, если единичны векторы И a, И b при адресации {1, 1} в состоянии t_4. Для других произвольных адресов $\{a, b\}$ и позиций $t_1 - t_3$ логическое пространство нулевое, так как один из векторов – нулевой (показан в виде точки с инверсным обозначением $\bar{a}_i$). В электрических цепях функцию И эквивалентно реализует последовательное соединение проводников (см. $1F(\varepsilon)$). Из диаграммы последовательного соединения следует закономерная аксиома, что сопротивление между узлами + и – будет нулевым, если привести в контакт со сферой F проводники И a, И b.

Таким образом, конъюнкция, как и логическое сложение, представляется по информационной модели неделимым комплексом: структурных схем И и формул логического умножения, имеющих множественную форму в базисах НКФ и НДФ, И-НЕ и ИЛИ-НЕ; стандартной таблицей истинности и инвариантного семейства временных диаграмм – эквивалентов согласованных информационных процессов, организуемых по закономерностям технологии проектирования микропроцессорных средств. Оператор конъюнкции аксиоматичен алгоритму последовательного включения элементов функции в топологии схемо- и адресации мнемотехники, регламентируемых стандартами и ЕСКД.

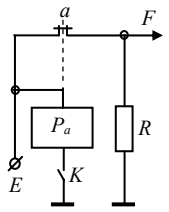
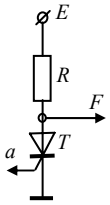
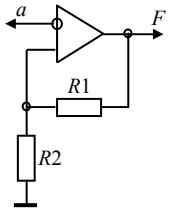
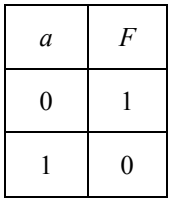
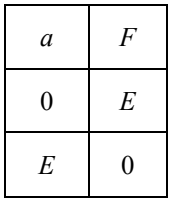
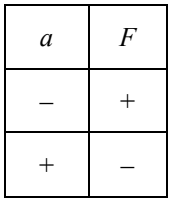
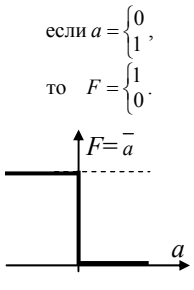
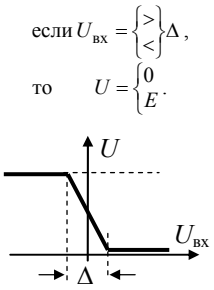
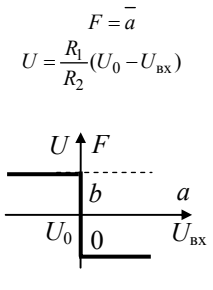
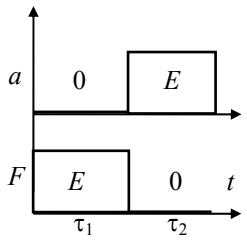
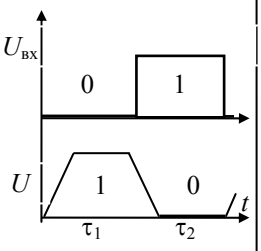
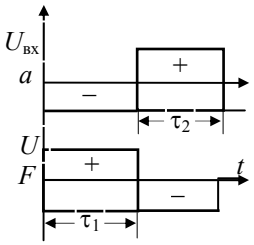
1.2.4 Инверсия

В булевой алгебре под инверсией понимают отрицание заданной величины [29, 40]. В унитарном логическом пространстве $F(1)$, аналогичном «бублику», его инверсией (отрицанием) является «дырка» $\bar{F} = 0$ (см. табл. 1.4 и 1.5, $1F(R)$). Справедливо и обратное утверждение: инверсией «дырки» – служит «бублик». Для значений алгебры Буля соответствует, что инверсия логической единицы $\bar{1}$ приводит к логическому нулю, а его отрицанием может быть только логическая единица. Инверсии функционального пространства ставят в соответствие логический элемент – инвертор в мнемонике схемотехники. Инвертор организуют смешанным соединением активных элементов, на базе электрических и электронных реле, тиристорах и транзисторах. Дизъюнкторы и конъюнкторы, в отличие от инверторов, могут быть реализованы при параллельном и последовательном включении пассивных ключей. Проиллюстрируем инверторы на примере электромагнитного реле (табл. 1.6, $1F(R)$), тиристорного ключа ($2F(R)$) и транзисторного триггера ($3F(R)$) на операционном усилителе.

Инверсию в релейно-контактной логике выполняют нормально закрытые контакты « », которым соответствует инверсный терм $\bar{a}$ в функциональных $1F(\Phi)$ координатах [12, 24]. В исходном (нормальном) состоянии катушка P_a реле $1F(R)$ обесточена, так как ключ K разомкнут и источник питания E не подключен к ее электрической обмотке. При этом в цепи управления протекает электрический ток от источника E через нормально закрытые контакты $\bar{a}$ и сопротивление R нагрузки. Сопротивление контактов бесконечно мало и все напряжение питания E распределяется на нагрузке R и соответственно подключено к выходу F. Потенциалу высокого уровня $F = E$ [В] физики сопоставляют в булевой алгебре логическую единицу. Следовательно, в исходном состоянии для $K = 0$ [л] на выходе $F = 1$ [л] за счет нормально закрытых контактов $\bar{a}$. Исходное состояние $a = 0$ для выхода $F = 1$ отражено в первой строке таблицы истинности $1F(T)$ и на первом интервале $\tau_1(0)$ семейства временных диаграмм $1F(\varepsilon)$. На входной диаграмме $a(\tau_1)$ логическому нулю соответствует потенциал низкого уровня 0 [л] $\Rightarrow$ 0 [В], а на выходе $F(\tau_1)$ присутствует потенциал высокого уровня, соответствующий логической единице 1 [л] $\Rightarrow E$ [В].

На первом адресе $a = 1$ [л] ключ замыкают $K = 1$ и через катушку P_a течет ток I, что приводит к переключению нормально замкнутых контактов « » в открытые « » с бесконечно высоким сопротивлением. Ток в управляющей цепи прекращается, а на сопротивлении R присутствует потенциал земли низкого уровня 0 [В], что тождественно логическому нулю на выходе $F = 0$ [л]. Это отражено во второй строке

1.6 Инверсия

	1 Реле	2 Тиристор	3 Триггеры
F(*R*)			
F(*T*)			
F(Ф)			
F(ε)			

a	*F*
0	1
1	0

a	*F*
0	*E*
E	0

a	*F*
–	+
+	–

если $a=\begin{cases}0\\1\end{cases}$, то $F=\begin{cases}1\\0\end{cases}$.

если $U_{вх}=\begin{cases}>\\<\end{cases}\Delta$, то $U=\begin{cases}0\\E\end{cases}$.

$F=\bar{a}$

$U=\frac{R_1}{R_2}(U_0-U_{вх})$

таблицы истинности 1*F*(*T*) и на семействе временных диаграмм 1*F*(ε) во втором такте τ_2, где по первому адресу $a(\tau_2) = E$ на выходе сформирован нулевой потенциал $F(\tau_2) = 0$ [В] или логический нуль. Характеристика реле и его контактов 1*F*(Ф) имеет ярко выраженный П-образный характер с двумя устойчивыми состояниями $U_{вых} = = \{0, E\}$ [В] = {0, 1} [л] относительно $U_{вх}$ при его сравнении с пороговым напряжением U_0. Статическая релейная характеристика соответствует алгоритму работы реле:

$$\text{если } U_{вх}\begin{cases}>\\<\end{cases}U_0, \text{ то } U_{вых}\begin{cases}0\\E\end{cases},$$

откуда следует логический алгоритм:

$$\text{если} \quad a = \begin{cases} 0 \\ 1 \end{cases}, \quad \text{то} \quad F = \begin{cases} 1 \\ 0 \end{cases}$$

и структурная формула инверсии

$$F = \overline{a} \, .$$

Следовательно, нормально закрытые контакты электромагнитного реле реализуют функцию инверсии в основных формах схемо- и мнемотехники, математики и физики на уровне смешанных соединений активных элементов и таблиц истинности, структурных формул и семейства временных диаграмм.

Инверсия на тиристорах реализована бесконтактной релейной логикой, организованной в виде управляемого делителя напряжения $2F(R)$ со входом a и выходом F. Инверсию формируют за счет изменения проводимости тиристора по адресному входу a при подаче на управляющий вход потенциалов низкого (нулевого) и высокого (единичного) уровней $a = \{0, E\}$ [В] = {0, 1} [л]. Для приведенной схемы $2F(R)$ проводимость близка к нулю, а сопротивление бесконечно большое $R_T \to \infty$, если на входе присутствует нулевой потенциал $a = 0$. Ток через тиристор не протекает и все напряжение источника E приложено к тиристору и выходу инвертора $F = E$. Нулевое состояние систематизировано в первой строке таблицы истинности $2F(T)$ и на первом интервале τ_1 семейства временных диаграмм $2F(\varepsilon)$. Логическому нулю на входе $a[\tau_1] = 0$ [л] тиристора соответствует логическая единица на выходе инвертора $F(\tau_1) = 1$. Следует отметить трапецеидальный характер сигнала на выходе F инвертора, что определяется задержкой включения (выключения) тиристора из-за установления динамического равновесия носителей электрических зарядов в областях p-n переходов.

Задержку на порог Δ иллюстрирует статическая характеристика $2F(\Phi)$ тиристора, имеющая гистерезисный характер трапецеидальной формы. Алгоритм работы тиристорного делителя в соответствии со статической характеристикой имеет вид:

$$\text{если} \quad U_{\text{вх}} \begin{Bmatrix} > \\ < \end{Bmatrix} \Delta, \quad \text{то} \quad U_{\text{вых}} = \begin{cases} 0 \\ E \end{cases},$$

или в логическом пространстве:

$$\text{если} \quad a = \begin{cases} 0 \\ 1 \end{cases}, \quad \text{то} \quad F = \begin{cases} 1 \\ 0 \end{cases},$$

откуда следует структурная формула инверсии:

$$F = \overline{a} \, .$$

Следовательно, управляемый делитель напряжения на тиристоре также реализует инверсию в бесконтактной релейной логике.

Инверсию на триггере выполняет компаратор с положительной обратной связью на резисторах R_1 и R_2. Сигнал a поступает на инверсный вход компаратора и сравнивается с опорным уровнем b по алгоритму

$$\text{если} \quad a \begin{Bmatrix} < \\ > \end{Bmatrix} b, \quad \text{то} \quad F = \begin{cases} 1 \\ 0 \end{cases}.$$

Это следует из алгоритма работы инвертора на компараторе, сравнивающего входной сигнал a амплитудой $U_{\text{вх}}$ с опорной мерой b амплитудой U_0, формируемой из выходного сигнала F амплитудой $|E|$:

$$\text{если} \quad U_{\text{вх}} \begin{Bmatrix} < \\ > \end{Bmatrix} U_0, \quad \text{то} \quad U = \begin{cases} E \\ -E \end{cases}.$$

Алгоритмы инвертора определяются статической характеристикой компаратора [22]

$$U = \frac{R_1}{R_2}(U_0 - U_{\text{вх}}),$$

тождественными ей функцией сравнения

$$F = (b - a)$$

и структурной формулой инверсии

$$F = \overline{a}$$

для $b = 0$ и заменой арифметического оператора вычитания – логическим инвертированием (см. $3F(\Phi)$).

Если сигнал a логического нуля отрицательного уровня «–», то за счет инверсии и положительной обратной связи на выходе формируется положительный потенциал насыщения $U = +E$ логической единицы $F = 1$. Устойчивое состояние «+» сохраняется на выходе до тех пор, пока на входе a не сформируется положительный уровень. На выходе сигнал инвертируется и скачком установится на отрицательном «–» уровне насыщения до следующего переключения при появлении сигнала a отрицательного уровня. Второе состояние отражено во второй строке таблицы истинности $3F(T)$ и на втором интервале τ_2 временных диаграмм $3F(\varepsilon)$. Из диаграмм видно, что по входному сигналу $a(\tau_2)$ положительного потенциала логической единицы на выходе устанавливается сигнал $F(\tau_2)$ отрицательного уровня логического нуля. Этому способствует релейная характеристика $3F(\Phi)$ электронного триггера на компараторе, выполненном на операционном усилителе с избыточным коэффициентом усиления.

Следовательно, электронный триггер на компараторе с положительной обратной связью реализует оператор инверсии в бесконтактной релейной логике.

Таким образом, оператор инверсии, в отличие от дизъюнкции и конъюнкции, организуют смешанным (параллельно-последовательным) соединением только активных элементов релейной логики: электрических реле и тиристорных ключей, транзисторных каскадов с общим эмиттером (истоком) и электронных триггеров на компараторах с положительной обратной связью. Инверсия представляется в основных формах науки и техники для организации согласованных компонент микропроцессорных средств по информационной технологии их проектирования.

Выводы

1 По вектору информативности рационально классифицировать схемы на комбинаторные, релейные и матричные с оптимальными методами проектирования итерацией по эквивалентам, алгебры Буля и математики образов в основных формах представления логических функций дизъюнкции, конъюнкции, инверсии.

2 Формы представления логических функций отражают на уровне аксиом компоненты микропроцессорных средств в виде многогранного неделимого комплекса структурных схем и формул, таблиц и диаграмм энергетических и информационных преобразователей на реле и полупроводниковых приборах, на ИС, СИС и БИС.

3 Аксиомы дизъюнкции, конъюнкции, инверсии постулируют алгоритмы параллельного, последовательного, смешанного объединения элементов функций в топологии схемо- и адресации мнемотехники, регламентируемых стандартами и единой системой конструкторской документации в формах НКФ и НДФ, базисов И-НЕ и ИЛИ-НЕ.

2 ТРИГГЕРЫ

Последовательностные схемы являются логическим развитием комбинационных при введении обратной связи для организации памяти. Элементами последовательностных схем служат триггеры, выполненные на двух логических элементах ИЛИ-НЕ (или И-НЕ), последовательно включенных друг за другом [29, 30, 63, 67, 71]. Триггером называют элемент памяти с двумя устойчивыми состояниями – нуль и единица. На уровне интегральных схем триггеры делят на статические (простые) и динамические (сложные), конструируемые из нескольких статических при их последовательном, параллельном и смешанном включении.

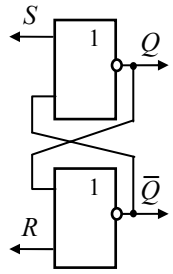
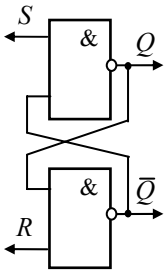
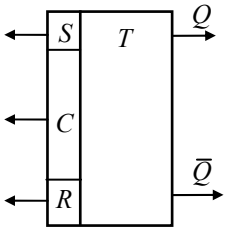
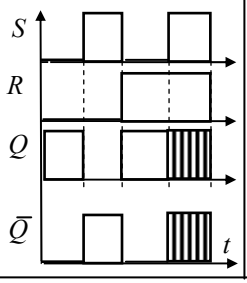
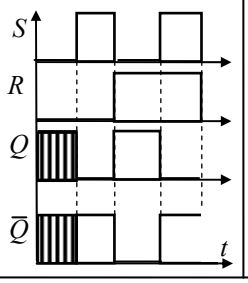
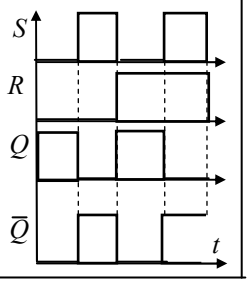
Простые триггеры [29, 30] классифицируют по статическим входам S и R, C и D, на RS-, RSC- и D-триггеры. Кроме статических S- и R-входов, динамические триггеры [30, 63] содержат информационные входы данных T и D, J и K, по которым их делят на T-, D- и JK-триггеры. Триггеры содержат прямой Q

и инверсный $\overline{Q}$ выходы, соответствующие статическим входам: прямому S и инверсному R, называемых по начальным буквам английских слов «*Set* – прямой» и «*Reset* – обратный».

Состояние триггера определяют по потенциалу на его прямом выходе. При переключении триггера в состояние логической единицы, на прямом выходе $Q = 1$ – потенциал высокого уровня. Если триггер в нулевом состоянии, то на прямом выходе $Q = 0$ – потенциал низкого уровня, а на инверсном $\overline{Q} = 1$ – высокий потенциал логической единицы. Состояние с одинаковыми сигналами на выходах триггера ($Q = \overline{Q}$) считают запрещенным, обозначают символом «∞» и исключают из алгоритма его функционирования согласно определению триггера. Неопределенные состояния формируются в простейших *RS*-триггерах, их запрещают входом синхронизации *C* или входом задержки данных *D*. Токовый *J* и потенциальный *K* входы расширяют функции *JK*-триггера в два раза относительно *D*-триггера, который содержит вдвое больше состояний *T*-триггера.

Информационная технология [16, 22] проектирования микропроцессорных средств систематизирует закономерности анализа и синтеза потенциальных триггеров в основных формах представления схемо- и мнемотехники, математики и физики. Ниже будет показано повышение технологичности проектирования схем в зависимости от увеличения степени интеграции в них функций от ИС и СИС к БИС и МИС. Однако, объединение закономерностей в правила и алгоритмы, методы и технологию требует исследования функции триггера в неделимом комплексе представления науки и техники на иерархических уровнях интеграции базисных структур микроэлектроники.

2.1 Триггеры

<table>
<tr><th></th><th>1 RS</th><th>2 RS</th><th>3 RSC</th></tr>
<tr><td>Схемы F(R)</td><td> 
</td><td> 
</td><td> 
</td></tr>
<tr><td>Таблицы F(T)</td><td>
<table>
<tr><th>S</th><th>R</th><th>Q_{k+1}</th></tr>
<tr><td>0</td><td>0</td><td>Q_k</td></tr>
<tr><td>1</td><td>0</td><td>0</td></tr>
<tr><td>0</td><td>1</td><td>1</td></tr>
<tr><td>1</td><td>1</td><td>∞</td></tr>
</table>
</td><td>
<table>
<tr><th>S</th><th>R</th><th>Q_{k+1}</th></tr>
<tr><td>0</td><td>0</td><td>∞</td></tr>
<tr><td>1</td><td>0</td><td>0</td></tr>
<tr><td>0</td><td>1</td><td>1</td></tr>
<tr><td>1</td><td>1</td><td>Q_k</td></tr>
</table>
</td><td>
<table>
<tr><th>S</th><th>R</th><th>Q_{k+1}</th></tr>
<tr><td>0</td><td>0</td><td>Q_k</td></tr>
<tr><td>1</td><td>0</td><td>0</td></tr>
<tr><td>0</td><td>1</td><td>1</td></tr>
<tr><td>1</td><td>1</td><td>$\overline{Q}_k$</td></tr>
</table>
</td></tr>
<tr><td>Формулы F(Ф)</td><td>$\begin{cases}\boldsymbol{Q}_{k+1}=[\overline{\boldsymbol{S}}(\boldsymbol{R}+\boldsymbol{Q})]_k\\ \overline{\boldsymbol{Q}}_{k+1}=[\overline{\boldsymbol{R}}(\boldsymbol{S}+\overline{\boldsymbol{Q}})]_k\end{cases}$
$\{0,0\}\Rightarrow \overline{0}(0+Q)=Q$
$\{1,0\}\Rightarrow \overline{1}(0+Q)=0$
$\{0,1\}\Rightarrow \overline{0}(1+Q)=1$
$\{1,1\}\Rightarrow \overline{1}(1+Q)=0$</td><td>$\begin{cases}\boldsymbol{Q_{k+1}}=\left(\overline{\boldsymbol{S}}+\boldsymbol{RQ}\right)_{\boldsymbol{k}}\\ \overline{\boldsymbol{Q}}_{\boldsymbol{k+1}}=\left(\overline{\boldsymbol{R}}+\boldsymbol{S}\overline{\boldsymbol{Q}}\right)_{\boldsymbol{k}}\end{cases}$
$\{0,0\}\Rightarrow \overline{0}+0Q=1$
$\{1,0\}\Rightarrow \overline{1}+0Q=0$
$\{0,1\}\Rightarrow \overline{0}+1Q=1$
$\{1,1\}\Rightarrow \overline{1}+1Q=Q$</td><td>$\boldsymbol{Q_{k+1}}=\left\{\boldsymbol{C}\left(\overline{\boldsymbol{S}}\boldsymbol{Q}+\boldsymbol{R}\overline{\boldsymbol{Q}}\right)\right\}_{\boldsymbol{k}}$
если $\boldsymbol{C},\boldsymbol{F}=\begin{cases}\boldsymbol{0}\\ \boldsymbol{1}\end{cases}$,
то $\boldsymbol{Q_{k+1}}=\begin{cases}\boldsymbol{Q_k}\\ (\boldsymbol{Q},\boldsymbol{S},\boldsymbol{R})\end{cases}$</td></tr>
<tr><td>Диаграммы F(ε)</td><td> 
</td><td> 
</td><td> 
</td></tr>
</table>

В табл. 2.1 систематизированы статические *RS*- и *RSC*-триггеры в основных формах представления функции: структурные схемы $F(R)$ и формулы $F(\Phi)$, таблицы состояния $F(T)$ и временные диаграммы $F(\varepsilon)$. Простейшие триггеры представлены в базисах интегральных схем ИЛИ-НЕ $F(\bar{1})$ и И-НЕ $F(\overline{\&})$ при их последовательном соединении друг за другом. По принципу аналогии структуры и связи ИС включены симметрично, что повышает наглядность и технологичность проектирования основных форм представления функций *RS*-триггера. Элементарные последовательностные преобразователи проектируют по аналогии с комбинационными схемами, но в отличие от них на следующем $(k + 1)$-м шаге учитывают состояние функции F_{k+1} в предыдущий k-й момент времени. Пропорционально числу обратных связей удваивается адресное пространство программирования, которое сокращают векторные таблицы при систематизации k-х состояний функции в вектор F_k.

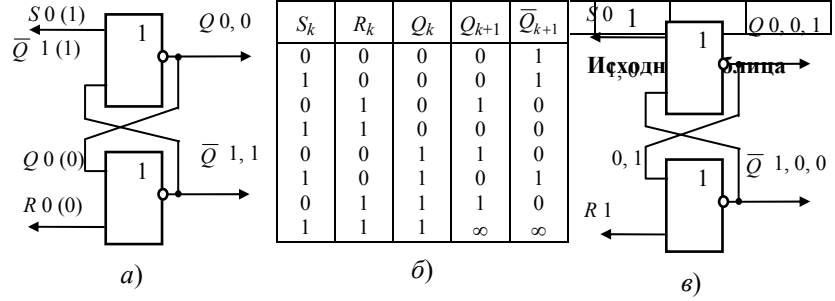
2.1 БАЗИС ИЛИ-НЕ

Проанализируем структурную схему *RS*-триггера [13, 29, 63, 67] в базисе ИЛИ-НЕ (см. табл. 2.1, $1F(R)$) по алгебре Буля методом единиц и нулей, синтезируя таблицу истинности переключения из исходного k-го в следующее $(k + 1)$-е устойчивое состояние. Исходные состояния в k-й момент времени по статическим S_k- и R_k-входам и прямому выходу Q_k сведем в таблицу дешифратора, систематизированную в адресном пространстве по двоичному коду стандартным образом (рис. 2.1). Таблицу мультиплексора составим из двух столбцов для иллюстрации $(k + 1)$-х состояний прямого Q_{k+1} и инверсного $\overline{Q}_{k+1}$ выходов *RS*-триггера (рис. 2.2, *а*). Таблица (рис. 2.1) сложную задачу для трех входов разделила на восемь аналогичных простых задач, адресованных линейно по возрастанию адресов $\{S, R, Q\}$ от нуля {0, 0, 0} до семи {1, 1, 1}.

S_k	R_k	Q_k	Q_{k+1}	$\overline{Q}_{k+1}$
0	0	0		
1	0	0		
0	1	0		
1	1	0		
0	0	1		
1	0	1		
0	1	1		

Алгоритм анализа структурной схемы и синтеза таблицы истинности состояний методом единиц и нулей (М10) следующий.

1 Начальный адрес {0, 0, 0} таблицы (рис. 2.1 и 2.2, *б*) перенесем на схему (рис. 2.2, *а*) и сопоставим исходному k-му состоянию связей: $S = 0$, $R = 0$, $Q = 0$ и логическую единицу на инверсном выходе, $\overline{Q} = 1$ согласно определению триггера.



S_k	R_k	Q_k	Q_{k+1}	$\overline{Q}_{k+1}$
0	0	0	0	1
1	0	0	0	1
0	1	0	1	0
1	1	0	0	0
0	0	1	1	0
1	0	1	0	1
0	1	1	1	0
1	1	1	∞	∞

а) *б*) *в*)

Рис. 2.2 Триггер на элементах ИЛИ-НЕ:
а – структурная схема с адресами {0, 0, 0} и {1, 0, 0};
б – таблица истинности состояний; *в* – структурная схема с адресом {0, 1, 0}

2 Определим для элементов ИЛИ-НЕ аналогичные операторы логического сложения с инверсией: $Q(\bar{1}) = \overline{S + \overline{Q}}$ и $\overline{Q}(\bar{1}) = \overline{R + Q}$, и вычислим следующие состояния на выходе $Q(\bar{1}) = \overline{0+1} = \bar{1} = 0$ и с учетом этого значения на выходе $\overline{Q}(\bar{1}) = \overline{0+0} = \bar{0} = 1$. В $(k + 1)$-м состоянии сохранилось исходное нулевое состояние, которое отразим по адресу {0, 0, 0} на выходах $\{Q, \overline{Q}\} = \{0, 1\}$ таблицы мультиплексора (рис. 2.2, *б*).

3 Учтем симметрию схемы (рис. 2.2, *а*) *RS*-триггера, а также инверсию элементов ИЛИ-НЕ и запишем по адресу {0, 0, 1} инверсное решение $\{Q, \overline{Q}\}=\{1, 0\}$ (рис. 2.2, *б*).

4 По пунктам 2, 3 проанализируем аналогично другие состояния (рис. 2.2, *а*).

а) Для адреса {1, 0, 0} в исходном состоянии (показано в скобках на рис. 2.2, *а*) $S = 1$, $R = 0$, $Q = 0$, а $\overline{Q} = 1$. На прямом выходе сформируется значение нуля, так как $Q(\overline{1}) = \overline{1+1} = \overline{1} = 0$, и логической единицы на инверсном выходе, так как $\overline{Q}(\overline{1}) = \overline{0+0} = \overline{0} = 1$, которые занесем во вторую строку по выходам $\{Q, \overline{Q}\}=\{0, 1\}$. Эта же комбинация соответствует адресу {1, 0, 1} из-за симметричной структуры *RS*-триггера, так как $Q(\overline{1}) = \overline{1+0} = 0$, а $\overline{Q}(\overline{1}) = \overline{0+0} = 1$.

б) При адресации {0, 1, 0} триггер из нулевого состояния переключается в единичное за две итерации (рис. 2.2, *в*). В первой итерации $Q_1 = \overline{0+1} = 0$ не изменяется, но становится противоположным $\overline{Q_1} = \overline{1+1} = 0$. Во второй итерации формируется устойчивое состояние $\{Q_2, \overline{Q_2}\} = \{1, 0\}$, так как $Q_2 = \overline{0+0} = 1$, $\overline{Q_2} = \overline{1+1} = 0$, которое переносим в таблицу (рис. 2.2, *б*). По принципам симметрии и аналогии по адресу {0, 1, 1} устанавливается то же состояние {1, 0}.

в) Аномальными для триггера являются адреса {1, 1, 0} и {1, 1, 1}, создающие на выходах равные нулевые состояния $\{Q, \overline{Q}\}=\{0, 0\}$. Это обусловлено наличием единиц на входах *R* и *S*, при которых оператор инверсного сложения инвертирует логическую единицу в нуль, так как $Q = \overline{Q} = \overline{1+a} = 0$, где *a* – любое значение выходов *Q* и $\overline{Q}$ *RS*-триггера. По определению «триггер – это элемент с двумя устойчивыми состояниями», поэтому аномальные состояния помечают символами «∞» (см. 8-ю строку, рис. 2.2, *б*), а третий и седьмой адреса запрещают.

Таким образом синтезированная таблица истинности состояний является результатом анализа структурной схемы М10.

2.1.1 Синтез структурных формул

Из анализа третьего пункта алгоритма синтеза таблицы (рис. 2.2, *б*) следуют закономерности тождественных состояний $\{Q, \overline{Q}\}=\{0, 1\}$ (см. 2-ю и 6-ю строки), которые не зависят от исходных состояний *RS*-триггера и определяются только адресом {1, 0}= {*S*, *R*} статических входов. Инверсному состоянию $\{Q, \overline{Q}\}=\{1, 0\}$ (см. 3-ю и 7-ю строки) соответствует второй адрес {0, 1} на входах *S* и *R*. Запрещенные состояния $\{Q, \overline{Q}\}=\{\infty, \infty\}$ формируются третьим адресом {*S*, *R*}={1, 1}, а неизменные исходные состояния $Q_k = Q_{k+1}$, $\overline{Q}_k = \overline{Q}_{k+1}$ инициируются по нулевому адресу {*S*, *R*}={0, 0} (см. 1-ю и 5-ю строки).

Выявленные закономерности систематизируем в таблице состояния с двухадресным пространством {*S*, *R*} (рис. 2.3) по аналогии с векторной таблицей проектирования мультиплексора [15, табл. 2.4]). В отличие от таблицы истинности, таблица триггера отражает установившиеся значения состояний Q_{k+1} на $(k + 1)$-й итерации, инициированные адресом {*S*, *R*} в *k*-м исходном состоянии.

S	*R*	Q_{k+1}
0	0	Q_k
1	0	0
0	1	1
1	1	∞

Рис. 2.3 Таблица состояния *RS*-триггера

Докажем эквивалентность полной (рис. 2.2, *б*) и векторной (рис. 2.3) таблиц состояния.

Теорема об эквивалентности таблиц: векторная и полная таблицы состояния эквивалентны, если они тождественны одной структурной схеме.

Докажем теорему методом структурных формул. Создадим эквивалентную структурную формулу в процессе анализа структурной схемы (см. рис. 2.2, *а*) *RS*-триггера для прямого выхода *Q*:

$$Q_{k+1} = \left(\overline{S + \overline{R + Q}}\right)_k .$$

По теореме Деморгана преобразуем эту формулу в базисе $Q(\overline{1})$ в минимизированный вид НДФ:

$$Q_{k+1} = \left[\overline{S}(R + Q)\right]_k .$$

Синтезируем структурную формулу Q^*_{k+1} в НДФ по полной таблице состояния (см. рис. 2.2, *б*):

$$Q^*(1) = q_2(1) + q_4(1) + q_6(1),$$

где $q_2(1) = \overline{S}R\overline{Q}$; $q_4(1) = \overline{S}\,\overline{R}Q$; $q_6(1) = \overline{S}RQ$.

Подставляя минтермы, получим формулу для $Q^*(1)$ в НДФ на $(k + 1)$-м состоянии $Q^*(1) = Q^*_{k+1}$:

$$Q^*_{k+1} = \left(\underline{\overline{S}R\overline{Q}} + \overline{S}\,\overline{R}Q + \underline{\overline{S}RQ}\right)_k.$$

Минимизируем формулу Q^*_{k+1}, вынося за скобку $\overline{S}$ и объединив первое и третье слагаемые

$$Q^*_{k+1} = \left\{\overline{S}\left[R\left(\overline{Q} + Q\right) + \overline{R}Q\right]\right\},$$

что по аксиоме дизъюнкции $\overline{Q} + Q = 1$ соответствует

$$Q^*_{k+1} = \left[\overline{S}\left(R + \overline{R}Q\right)\right]_k.$$

Преобразуем выражение по теореме Деморгана

$$Q^*_{k+1} = \left[\overline{S + \overline{R}\ \ (R + \overline{Q})}\right]_k = \left(\overline{S + \overline{R}\ \ \overline{Q}}\right)_k.$$

После второго преобразования по теореме Деморгана находим минимизированную структурную формулу в НДФ:

$$Q^*_{k+1} = \left[\overline{S}\left(R + Q\right)\right]_k,$$

что тождественно эквиваленту $Q^*_{k+1} = Q_{k+1}$.

Синтезируем также по векторной таблице состояния (см. рис. 2.3) структурную формулу для Q^0_{k+1} в НДФ:

$$Q^0(1) = q_0(1) + q_2(1),$$

где $q_0(1) = \overline{S}\,\overline{R}Q$; $q_2(1) = \overline{S}R$.

После подстановки минтермов находим $Q^0_{k+1} = Q^0(1)$:

$$Q^0_{k+1} = \left[\overline{S}\left(\overline{R}Q + R\right)\right]_k.$$

Дважды применив теорему Деморгана и используя аксиому конъюнкции, получим минимизированное решение

$$Q^0_{k+1} = \left[\overline{S}\left(R + Q\right)\right]_k,$$

которое также тождественно эквиваленту $Q^0_{k+1} = Q_{k+1}$. Отсюда следует тождественность функций $Q^0_{k+1} = Q^*_{k+1}$ и эквивалентность полной и векторной таблиц состояния, что и требовалось доказать.

Векторная таблица по адресам $\{S, R\}$ в компактной форме систематизирует возможные состояния *RS*-триггера трехадресного $\{S, R, Q\}$ пространства полной таблицы, поэтому в мнемотехнике структурной схеме последовательстного элемента целесообразно сопоставить векторную таблицу, используя методы булевой алгебры: структурных формул и единиц и нулей.

2.1.2 Анализ структурных формул

Анализ структурных формул возможен методами алгебры Буля: МСФ и М10 и аналогии (МА). Анализ основан на эквивалентности исследуемой формы образцовой мере в одной из форм представления функции.

В методе структурных формул сравнивают математические образы исследуемой и образцовой функций в одной из форм НКФ или НДФ, И-НЕ или ИЛИ-НЕ. Один из примеров рассмотрен выше при доказательстве тождественности полной и векторной таблиц состояния.

Метод единиц и нулей М10 позволяет синтезировать таблицу состояния в процессе анализа структурных формул по адресам состояний триггера. Синтезированная таблица Q^* оценивается на тождественность с эквивалентной Q по алгоритму:

$$\text{если } Q^* \begin{Bmatrix} = \\ \neq \end{Bmatrix} Q, \text{ то } Q_{k+1} \begin{cases} \text{верна} \\ \text{неверна} \end{cases}.$$

Если таблицы тождественны $Q^* = Q$, то структурная формула Q_{k+1} спроектирована верно, в противном случае $Q^* \neq Q$, структурная схема Q_{k+1} неверна и ее анализ необходимо повторить.

Проектирование методом М10 проведем на примере анализа структурной формулы

$$Q_{k+1} = \left[\bar{S}\,(R + Q)\right]_k .$$

Для этого подготовим таблицу Q^* (рис. 2.4), заполненную по входу стандартным образом по четырем возможным состояниям с нулевого по третий адрес. Коду $\{S, R\}$ сопоставим из первой строки таблицы адрес $\{0, 0\}$ и разместим логические нули по соответствующим позициям анализируемой формулы

$$Q_{k+1}\{0, 0\} = \left[\bar{0}\,(0 + Q)\right]_k = (1 \cdot Q)_k = Q_k .$$

S		Q^*_{k+1}
0	0	Q_k
1	0	0
0	1	1
1	1	0

Рис. 2.4 Синтезируемая таблица

Оператор счисления показывает, что по нулевому адресу функция Q_{k+1} устанавливается в состояние Q_k, т.е. $Q_{k+1}\{0, 0\} = Q_k$. Это решение систематизируем в выходной столбец синтезируемой таблицы (рис. 2.4) по адресу $\{0, 0\}$.

Аналогично вышеописанному анализируем состояние по первому адресу $\{S, R\}=\{1, 0\}$:

$$Q_{k+1}\{1, 0\} = \left[\bar{1}\,(0 + Q)\right]_k = (0 \cdot Q)_k = 0 .$$

Подстановка логических единиц и нуля по позициям $\{S, R\}$ структурной формулы показывает нулевое состояние RS-триггера $Q_{k+1}\{1, 0\} = 0$, которое заносим во вторую строку выходного столбца таблицы (рис. 2.4).

На следующих адресах $\{0, 1\}$ и $\{1, 1\}$ проводим подобные итерации (см. $1F(\Phi)$ табл. 2.1) и значения состояний $Q_{k+1}\{0, 1\} = 1$ и $Q_{k+1}\{1, 1\} = 0$ заносим на соответствующие знакоместа синтезируемой таблицы (рис. 2.4).

Из сравнения созданной Q^* (рис. 2.4) и эквивалентной Q (табл. 2.1, $1F(T)$) таблиц, отмечаем их тождественность, так как $Q_{k+1}\{1, 1\} = 0$ соответствует неопределенности ∞. Если $Q^* = Q$, то утверждаем, что структурная формула Q_{k+1} по прямому выходу RS-триггера спроектирована верно.

Следовательно, методы структурных формул МСФ и единиц и нулей М10 алгебры Буля анализируют структурные формулы в процессе синтеза таблицы состояния. Анализ основанна эквивалентности исследуемой формы образцовой мере в адресном пространстве мнемотехники.

2.1.3 Проектирование временных диаграмм

Семейство временных диаграмм адекватно отражает физику функционирования последовательностного элемента. Временные диаграммы RS-триггера проектируют по аналогии с логическими элементами, но учитывают на $(k + 1)$-м интервале состояние на k-м шаге.

Наиболее наглядно проектирование при синтезе диаграмм $F(\varepsilon)$ по таблице состояния $F(T)$. Выбираем систему координат. По оси ординат откладываем входы S, R и выходы Q, $\overline{Q}$ от времени t. По оси абцисс за период T отмечают равные интервалы $t_j = t_{j+1}$ по числу n возможных состояний, определяемых количеством входов по двоичному коду. Для двух входов R и S число состояний $n = 2^2$, что соответствует четырехадресному пространству состояний диаграммы (см. табл. 2.1, $1F(\varepsilon)$) и таблицы (см. $1F(T)$). По оси ординат мерой служит уровень потенциала: низкий для логического нуля и высокий для логической единицы.

Для синтеза диаграмм $F(\varepsilon)$ таблицу $F(T)$ транспонируют и накладывают на координатную матрицу по возрастанию адресов по оси времени с первого $t(0, 0)$ до четвертого $t(1, 1)$ такта за период T. Входные диаграммы S и R формируют согласно адресам, заменяя потенциалами низкого и высокого уровня логические нули и единицы (см. табл. 2.1, $1F(\varepsilon)$). Выходные диаграммы синтезируют аналогично, начиная со второго $t(1, 0)$ и третьего $t(0, 1)$ интервала, чтобы наглядно проиллюстрировать состояния на первом $t(0, 0)$ и четвертом $t(1, 1)$ интервалах. Если на втором интервале состояние $Q_k\{1, 0\} = 0$ низкого уровня, то для наглядности переключения RS-триггера на первом интервале $Q_{k+1}\{0, 0\} = = Q_k$ целесообразно вектору состояния присвоить потенциал высокого уровня, а по инверсному $\overline{Q}_{k+1}\{0, 0\} = \overline{Q}_k$ выходу показать низкий уровень потенциала. Четвертый интервал $t(1, 1)$ с неопределенностью ∞ выходных состояний следует заполнить последовательностью импульсов по прямому Q и инверсному $\overline{Q}$ выходам (см. $1F(\varepsilon)$).

Анализ правильности синтеза семейства временных диаграмм организуют методами МСФ и М10 при синтезе структурных формул в НДФ и НКФ и таблицы состояния, которые оценивают по тождественности соответствующим эквивалентам. Проектировать временные диаграммы $F(\varepsilon)$ можно по аналогии с таблицами $F(T)$ в процессе анализа структурной схемы $F(R)$ методом единиц и нулей или методом аналогии.

Следовательно, проектирование семейства временных диаграмм аналогично синтезу таблицы истинности по эквивалентам методами структурных формул МСФ, единиц и нулей М10 и методами аналогии.

Таким образом, RS-триггер является развитием логических элементов и представляется неделимым комплексом структурных схем и формул, векторной таблицы состояния и семейства временных диаграмм, которые проектируются методами булевой алгебры и математики образов по аналогии с комбинационными интегральными схемами. Особенности проектирования последовательностных элементов обусловлены следующими отличиями:

– структурная схема содержит обратную связь с выхода на вход для организации управляемой памяти;

– таблица состояния систематизирует функцию в адресном пространстве не только топологии, но и времени;

– структурная формула моделирует установившееся значение функции в зависимости от вектора состояния на предыдущем шаге;

– семейство временных диаграмм отражает не только статику регламентированных состояний, но и динамику функции переключения для изменяемых значений вектора состояния.

2.2 БАЗИС И-НЕ

Последовательностный элемент можно сконструировать в базисе И-НЕ при последовательном включении логических конъюнкторов с инверсией (см. табл. 2.1, $2F(R)$). Логические элементы размещают симметрично статическим входам S и R, а также выходам: прямому Q и инверсному $\overline{Q}$ простого RS-триггера. Для комплексного представления функции переключения спроектируем таблицу состояния, структурные формулы и семейство временных диаграмм методами структурных формул МСФ, единиц и нулей М10 булевой алгебры, а также методами аналогии математики образов.

2.2.1 Метод структурных формул

Структурные формулы синтезируют с выхода на вход заменяя связи (входы и выходы) и структуры (логические элементы) интегральных схем аналогичными математическими образами: выходными пе-

ременными S, R и выходными векторами состояния Q, $\overline{Q}$ (связями, сигналами) и операторами (структурами) счисления (дизъюнкции, конъюнкции, инверсии).

Прямой выход Q через элемент И-НЕ соединен с прямым выходом S и инверсным выходом $\overline{Q}$ (см. табл. 2.1, $2F(R)$). Заменяя элементы НЕ – инверсией, а И – конъюнкцией, запишем для прямого выхода функцию $Q(\overline{\&})$

$$Q(\overline{\&}) = \overline{S \cdot \overline{Q}}.$$

В том же базисе $\overline{\&}$ выход $\overline{Q}$ связан с инверсным входом R и прямым выходом Q (см. $2F(R)$). По методу МСФ для инверсного выхода по аналогии справедлива зависимость:

$$\overline{Q}(\overline{\&}) = \overline{R \cdot Q}.$$

Подставляя второе выражение в первое, получим формулу в базисе И-НЕ для прямого выхода:

$$Q(\overline{\&}) = \overline{S \cdot \overline{RQ}},$$

а после подстановки первого оператора во второй находим зависимость $\overline{Q}(\overline{\&})$ для инверсного выхода

$$\overline{Q}(\overline{\&}) = \overline{R \cdot \overline{SQ}}.$$

Используя теорему Деморгана, преобразуем полученные решения в НДФ:

$$\begin{cases} Q(1) = \overline{S} + RQ; \\ \overline{Q}(1) = \overline{R} + S\overline{Q}. \end{cases}$$

Отразим состояния переменных на k-м шаге и векторов состояний на $(k + 1)$-м шаге соответствующими индексами, что соответствует структурным формулам RS-триггера в НДФ

$$\begin{cases} Q_{k+1}(1) = \left(\overline{S} + RQ\right)_k; \\ \overline{Q}_{k+1}(1) = \left(\overline{R} + S\overline{Q}\right)_k. \end{cases}$$

Для удобства и простоты представления структурных формул индексы k-го шага переменных правой части формулы выносят за скобки, а форму представления указывают в скобках вектора состояния выходной функции. Анализируют структурные формулы RS-триггера методами алгебры Буля и аналогии по тождественности эквивалентам структурных схем и формул, таблиц состояния и временных диаграмм, синтезируемых при анализе математических образов.

Следовательно, структурные формулы проектируют аналогично комбинационным схемам, но в отличие от них для последовательностных элементов отражают установившиеся состояния векторов в зависимости от исходного значения.

2.2.2 Метод единиц и нулей

Метод единиц и нулей М10 позволяет синтезировать таблицы состояния и временные диаграммы при анализе структурных схем и формул RS-триггера подобно комбинационным преобразователям. Анализ М10 универсален для любых структур в форме НДФ $F(1)$ и НКФ $F(0)$, в базисах ИЛИ-НЕ $F(\overline{1})$ и И-НЕ $F(\overline{\&})$. Проведем проектирование методом единиц и нулей М10 на примере анализа структурных формул RS-триггера в НДФ $F(1)$ и И-НЕ $F(\overline{\&})$ в процессе синтеза семейства временных диаграмм $F(\varepsilon)$.

Структурные формулы RS-триггера в единичной форме $F(1)$ имеют вид:

$$\begin{cases} Q_{k+1} = \left(\overline{S} + RQ\right)_k; \\ \overline{Q}_{k+1} = \left(\overline{R} + S\overline{Q}\right)_k. \end{cases}$$

Подготовим матрицу семейства временных диаграмм по двум входам R, S и выходам Q, $\overline{Q}$. По оси времени t разделим период T на четыре равных интервала $t_j = t_{j+1} = T/4$, $j = \overline{1, 4}$, по двоичному коду $N_2(2) = 2^2$ двух входных переменных S и R (рис. 2.5). Входные диаграммы синтезируем стандартным образом по линейному закону возрастания адресов t $\{R, S\}$ от нуля на первом интервале t_1 $\{0, 0\}$ до трех на четвертом t_4 $\{1, 1\}$. Адресное пространство представим аналогичными физическими образами: потенциалами высокого E (единичного) и низкого 0 (нулевого) уровня, соответствующими логическим единице и нулю алгебры Буля. При этом на первом интервале t_1 $\{0, 0\}$ сформированы потенциалы низкого уровня на входах $\{S, R\} = \{0, 0\}$, на четвертом t_4 $\{1, 1\}$ – высокого $\{S, R\} = \{E, E\}$ и т.д. Адресовать интервалы целесообразно или в логическом $F(T)$ или $F(\varepsilon)$ физическом пространствах. Временная диаграмма дифференцирует сложную задачу на аналогичные итерации для j-го интервала времени. Последовательность синтеза выходных диаграмм начинают со статики регламентированных состояний, продолжают для динамических векторов состояния и заканчивают интервалом с неопределенными состояниями. Последовательность синтеза от простого к сложному рациональна не только с дидактической точки зрения, но и для повышения эффективности проектирования по метрологическим, технологическим и эргономическим показателям. Как правило, статическими состояниями простого RS-триггера наделены промежуточные интервалы t_2 и t_3 с адресами $\{1, 0\}$ и $\{0, 1\}$, а первый t_1 и последний t_4 интервалы периода соответствуют динамическим состояниям (рис. 2.6). Поэтому начнем синтез выходных диаграмм от статики к динамике (от простого к сложному).

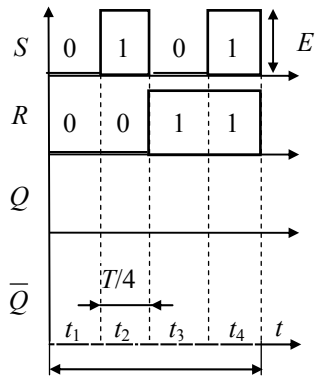


Рис. 2.5 Матрица диаграмм

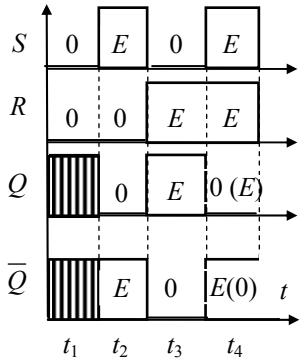


РИС. 2.6 СИНТЕЗИРОВАННЫЕ ДИАГРАММЫ

Для второго интервала t_2 на входах присутствуют потенциалы высокого E и низкого уровня $\{S, R\} = \{E, 0\}$, что в логическом пространстве соответствует адресу $\{1, 0\} = \{S, R\}$, т.е. $S = 1$, $R = 0$. Подставим в структурные формулы логические переменные согласно позициям S и R:

$$\begin{cases} Q_{k+1}\{1, 0\} = \left(\overline{1} + 0Q\right)_k = 0\,[\text{л}] = 0\,[\text{В}]; \\ \overline{Q}_{k+1}\{1, 0\} = \left(\overline{0} + 1\overline{Q}\right)_k = 1\,[\text{л}] = E\,[\text{В}]. \end{cases}$$

В квадратных скобках отражены меры пространств, соответственно [л] – логика в логических нулях или единицах и [В] – физика в вольтах потенциалов низкого и высокого уровня. Результаты $\{Q, \overline{Q}\}_{k+1} = \{0, E\}$ систематизируем в выходные диаграммы (см. рис. 2.6) во втором интервале t_2 $\{1, 0\}$.

Аналогично решаем задачу для третьего интервала t_3 $\{0, 1\}$ с потенциалами низкого (нулевого) и высокого (единичного) уровня на входах $\{S, R\} = \{0, E\}$ [В] = $\{0, 1\}$ [л], подставляя на позиции переменных S и R системы уравнений логические нуль и единицу:

$$\begin{cases} Q_{k+1}\{0, 1\} = \left(\overline{0} + 1Q\right)_k = 1\,[\text{л}] = E\,[\text{В}]; \\ \overline{Q}_{k+1}\{0, 1\} = \left(\overline{1} + 0\overline{Q}\right)_k = 0\,[\text{л}] = 0\,[\text{В}]. \end{cases}$$

На третьем интервале $t_3\{0, E\}$ выходных диаграмм (рис. 2.6) сформируем статические состояния высокого и низкого уровня $\{Q, \overline{Q}\}_{k+1} = \{E, 0\}_k$. Аналогичными итерациями М10 синтезируют состояния в динамике.

Динамические состояния на первом t_1 и четвертом t_4 интервалах выходных диаграмм (рис. 2.6) синтезируем при анализе М10 структурных формул $F(\overline{\&})$. В базисе И-НЕ структурные формулы RS-триггера ($2F(R)$ табл. 2.1) без индексов состояний представлены системой

$$\begin{cases} Q(\overline{\&}) = \overline{S \cdot \overline{RQ}}; \\ \overline{Q}(\overline{\&}) = \overline{R \cdot \overline{S\overline{Q}}}. \end{cases}$$

Для первого интервала t_1 (0, 0) с нулевыми потенциалами на входах $\{S, R\} = \{0, 0\}$ [В] = $\{0, 0\}$ [л] переменным S и R соответствуют логические нули. Заменим в формулах $\overline{\&}$ значения переменных S и R логическими нулями:

$$\begin{cases} Q\{0, 0\} = \overline{0 \cdot \overline{0\overline{Q}}} = \overline{0 \cdot \overline{1}} = 1\ [\text{л}] = E\,[\text{В}]; \\ \overline{Q}\{0, 0\} = \overline{0 \cdot \overline{0\overline{Q}}} = \overline{0 \cdot \overline{1}} = 1\ [\text{л}] = E\,[\text{В}]. \end{cases}$$

Из определения следует, что триггер не может иметь одинаковые состояния, поэтому заменим потенциалы высокого уровня неопределенностью $\{Q, \overline{Q}\} = \{E, E\} = \{\infty, \infty\}$, а на выходных диаграммах (рис. 2.6) в первом интервале t_1 отразим неопределенность последовательностью импульсов.

Аналогично решаем задачу для четвертого интервала t_4 (1, 1) с высокими потенциалами на входах $\{S, R\} = \{E, E\}$ [В] = $\{1, 1\}$ [л], подставляя на позиции переменных S и R системы уравнений логические единицы:

$$\begin{cases} Q\{1, 1\} = \overline{1 \cdot \overline{1\overline{Q}}} = \overline{1 \cdot \overline{\overline{Q}}} = Q\ [\text{л}] = \{0, E\}\,[\text{В}]; \\ \overline{Q}\{1, 1\} = \overline{1 \cdot \overline{1\overline{Q}}} = \overline{1 \cdot Q}\ [\text{л}] = \{E, 0\}\,[\text{В}]. \end{cases}$$

Результаты решения $\{Q, \overline{Q}\} = \{0, E\}$ переменных векторов состояния отразим на четвертом интервале t_4 (рис. 2.6), выбрав для наглядности инверсные уровни $\{\overline{E}, \overline{0}\} = \{0, E\}$ потенциалов третьего интервала t_3.

Синтезированные диаграммы (рис. 2.6) сравним с эквивалентом (см. табл. 2.1, $2F(\varepsilon)$) и установим их тождественность за период T в управляемом адресном пространстве. Из тождества синтезируемых и эквивалентных временных диаграмм следует, что анализируемые методом единиц и нулей структурные формулы *RS*-триггера в формах НДФ и И-НЕ синтезированы вцелом верно.

По методу единиц и нулей при анализе структурных формул и схем по аналогии с семейством временных диаграмм $2F(\varepsilon)$ можно синтезировать таблицы истинности $F(T)$. При этом потенциалы низкого (нулевого) и высокого (единичного) уровня заменяют аналогичными образами алгебры Буля: логическими нулями и единицами.

Следовательно, метод единиц и нулей позволяет синтезировать временные диаграммы и таблицы состояния *RS*-триггера при анализе по адресам структурных схем и формул в НДФ и НКФ, ИЛИ-НЕ и И-НЕ. При синтезе таблиц и диаграмм тождественных эквивалентам заключают о правильности проектирования структур в образах схемотехники и математики.

2.2.3 Структуры и технология

Множественность структурных схем и формул в цифровой и микропроцессорной технике определяется широким разнообразием элементной базы (электростатические и электромагнитные элементы, электровакуумные и полупроводниковые приборы) и уровнем интеграции базисов (ПП и ИС, СИС и БИС), топологией схем (комбинаторные, релейные, матричные) и мнемоникой программ (таблицы истинности и состояния, векторные таблицы и блок-схемы). Многообразие структур и форм их представления затрудняет выявление и систематизацию закономерностей в алгоритмы и методы для организации информационной технологии проектирования микропроцессорных средств.

Наглядным примером множества тождественных топологических структур являются структурные схемы и формулы *RS*-триггеров в базисах И-НЕ $F(\overline{\&})$ и ИЛИ-НЕ $F(\overline{1})$, формах НКФ $F(\&)$ и НДФ $F(1)$, представленные в табл. 2.2. Таблица иллюстрирует 16 вариантов интегральных схем и их алгоритмов, систематизированных в матрицу строк и столбцов форматом 4 × 4, адресованных по базисам ИС ИЛИ-НЕ и И-НЕ, НКФ и НДФ (по строкам) и формам структур $F(1)$ и $F(0)$, $F(\overline{\&})$ и $F(\overline{1})$. Несложно видеть существенные отличия тождественных схем: громоздкость и аппаратную избыточность форм единичных $F(1)$ и нулевых $F(0)$ функций во всех базисах НДФ и НКФ, И-НЕ и ИЛИ-НЕ (первый и второй столбцы табл. 2.2), а также базисов НКФ и НДФ в анализируемых формах $F(1)$ и $F(0)$, $F(\overline{\&})$ и $F(\overline{1})$ (первая и вторая строки табл. 2.2) для выявления закономерностей и систематизации в перспективные алгоритмы и методы. Другие решения нетехнологичны и нерациональны (нижняя левая и верхняя правая четверти табл. 2.2), составляют половину банка данных среднестатистических аналогов и могут служить только для оценки эффективности прототипов.

Следует отметить простоту и симметричность структур в базисах И-НЕ и ИЛИ-НЕ форм $F(\overline{\&})$ и $F(\bar{1})$ (нижний правый квадрант), что обусловлено минимизацией операторов счисления в процессе преобразования нормальных форм НДФ и НКФ по аксиомам и теоремам алгебры Буля и серийным изготовлением ИС в базисах И-НЕ и ИЛИ-НЕ. Однако, изящность изобретений скрывает творческие приемы разрешения противоречий и не позволяет оценить и систематизировать новаторские алгоритмы в новую методику проектирования. Кроме того, рациональное решение для интегральных схем может быть неудачным и даже неприемлемым не только для других схем комбинаторики, но и для упорядоченных структур релейной и матричной логики.

Число структур увеличивается до необозримого множества, если решению на ИС добавить комбинаторные схемы в диодной, транзисторной и диодно-транзисторной логике; релейные схемы на компараторах, таймерах и генераторах в релейном, тиристорном и транзисторном исполнении; матричную логику ИС, СИС и БИС. Необозримость тезауруса данных приводит к бессистемным и трудоемким поискам решений методами проб и ошибок, итерационного анализа и последовательных приближений. Венцом эвристических методов является изобретательство, основанное на озарении свыше по мистическим принципам без понимания объективных закономерностей.

Схемотехнику называют искусством (изобретательством) схем [1, 39, 50, 63, 72] из-за многогранности форм представления несистематизированных образов науки и техники. Одной из форм сокращения банка данных являются единая система конструкторской документации (ЕСКД) и стандарты, регламентирующие проектирование и конструирование схем и программ, операторов вычисления и оценок эффективности. Любая систематизация стандартизирует закономерность в рациональный алгоритм, как целенаправленную последовательность элементарных операций для получения заданного решения. Мерой оценки эффективности множественности структур и форм функции может служить семейство временных диаграмм, адекватно отражающих физику информационных процессов (функций) преобразования и управления, программирования и вычисления. В отличие от многообразия структурных схем и формул, алгоритмов и программ, имеет однозначное представление только семейство временных диаграмм, эквивалентных статике, кинетике и динамике объективных, не зависящих от субъекта, физических явлений.

Комплексное представление функций в систематизированном адресном пространстве топологии $F(R)$ и мнемоники $F(T)$, математики $F(\Phi)$ и метрологии $F(\varepsilon)$ в виде информационной модели $F(R, T, \Phi, \varepsilon)$ позволяет эвристическую цифровую технику перевести из ранга изобретательских проблем в инженерную методику синтеза и анализа согласованных схем и программ, формул и диаграмм [16]. Выявление и изучение объективных закономерностей в комплексной форме представления функции интегрируют на уровне информационного обеспечения аппаратные и метрологические средства, программное и математическое обеспечение для систематизации правил и алгоритмов в методики и методы информационной технологии проектирования микропроцессорных средств [15 – 24].

Следовательно, информационная технология проектирования преобразует эвристическую цифровую технику в систематизированную микросхемотехнику микропроцессорных средств.

Информационная технология – это не только использование персональных компьютеров, а создание банка систематизированных компонент и форм представления, моделей и алгоритмов проектирования, методов и принципов созидания, объединенных концепцией перспективного развития.

Микросхемотехника микропроцессорных средств продиктована современным этапом научно-технической революции (НТР) – информатизацией [16]. Идеологией микросхемотехники является развитие информационных процессов за счет интеграции функций, систематизированных в информационную концепцию микроэлектроники. Информационная концепция объясняет становление компонент на уровне аппаратных средств (ПП, ИС, СИС) и программного обеспечения (БИС), математического обеспечения (ПК) и метрологических средств (МИС), которые организует в информационную модель. Информационная модель в координатах функционально-пространственно-временного континуума дифференцирует компоненты МС в топологию $F(R)$ схемотехники и мнемонику $F(T)$ программирования (мнемотехники), в образы счисления $F(\Phi)$ математики и оценок эффективности $F(\varepsilon)$ физики.

Информационная модель $F(R, T, \Phi, \varepsilon)$ интегрирует компоненты микропроцессорных средств в неделимый комплекс информационного обеспечения, а на нижнем уровне ИС – в основные формы представления функций структурных схем $F(R)$ и формул $F(\Phi)$, таблиц состояния векторов $F(T)$ и семейства временных диаграмм $F(\varepsilon)$. В основу анализа и синтеза компонент микропроцессорных средств и форм представления функций положены закономерности проектирования, систематизирующие объективные физические явления в информационные принципы: аналогии и эквивалентности, инверсии и симметрии. Принципы постулируют закономерности схемо- и мнемотехники, логики и математики, физики и

метрологии в целенаправленные алгоритмы, мнемонические правила, методы анализа и синтеза функций и компонент микропроцессорной техники. Систематизация перспективных методов эквивалентности аналогов формирует инженерные методики проектирования, организованные в информационную технологию проектирования форм представления функций интегральных схем и компонент информационного обеспечения микропроцессорных средств [20 – 24].

Таким образом, информационная технология проектирования микропроцессорных средств кроме использования компьютеров основана на информационных концепции и процессах, модели и обеспечении, принципах и методах, обусловленных информатизацией научно-технической революции.

2.3 *RSC*-ТРИГГЕР

Синхронный *RSC*-триггер [13, 29, 30, 67] исключает неопределенное состояние за счет введения тактового входа *C* (см. табл. 2.1, 3*F*(*R*)). Таблица состояния (см. табл. 2.1, 3*F*(*T*)) отражает алгоритм функционирования *RSC*-триггера для открытого входа $C = 1$. При наличии на тактовом входе нулевого потенциала ($C = 0$) на его выходах состояния не изменяются $Q_{k+1} = Q_k$, $\overline{Q}_{k+1} = \overline{Q}_k$. Управляют работой триггера тактовыми (синхронизующими) импульсами *F* по входу *C*: потенциалами низкого и высокого уровня, соответствующими логическим нулю и единице $F = \{\overline{1,\ 0}\}$. Алгоритм работы *RSC*-триггера (см. 3*F*(Ф)) имеет вид:

$$\text{если} \quad C,F = \begin{cases} 0 \\ 1, \end{cases} \quad \text{то} \quad Q_{k+1} = \begin{cases} Q_k \\ (Q,S,R)_k. \end{cases}$$

Проектирование структурных формул *RSC*-триггера проведем методами булевой алгебры (МСФ и М10) и аналогии (МА).

2.3.1 Структурные формулы

Проектирование включает синтез структурных формул в НДФ *F*(1) и НКФ *F*(0) и анализ их тождественности по алгоритму:

$$\text{если} \quad F(1)\begin{cases} = \\ \neq \end{cases}F(0), \quad \text{то проектирование} \quad \text{СФ}\begin{cases} \text{верно} \\ \text{неверно.} \end{cases}$$

При тождественности решений утверждают, что структурная формула спроектирована верно, в противном случае необходимо повторить синтез исследуемой или эквивалентной структуры. В данном методе не принципиально, какое из решений является исследуемой функцией, а какое принято за нормированную меру – эквивалент.

Синтез структурных формул в НДФ *F*(1) включает сумму минтермов единичных $f_j(1)$ функций. Согласно таблице состояния (табл. 2.1, 3*F*(*T*)) для прямого выхода *Q* в НДФ справедлива формула

$$Q(1) = f_0(1) + f_2(1) + f_3(1)$$

с единичными минтермами

$$f_0(1) = \overline{S}\,\overline{R}\,Q, \ f_2(1) = \overline{S}\,R, \ f_3(1) = S\,R\,\overline{Q}.$$

После подстановки минтермов находим структурную формулу $Q_{k+1}(1)$ на $(k + 1)$-м состоянии в НДФ

$$Q_{k+1}(1) = \left(\overline{S}\,\overline{R}\,Q + \overline{S}\,R + S\,R\,\overline{Q}\right)_k.$$

Стандартное решение предполагает четное число слагаемых за счет использования аксиомы дизъюнкции ($1 = Q + \overline{Q}$) для приведения подобных членов:

$$Q_{k+1}(1) = \left[\left(\overline{S}\,\overline{R}\,Q + \overline{S}\,R\,Q\right) + \left(\overline{S}\,R\,\overline{Q} + S\,R\,\overline{Q}\right)\right]_k.$$

Группировка приводит к виду

$$Q_{k+1}(1) = \left[\overline{S}Q\left(\overline{R}+R\right)+R\overline{Q}\left(\overline{S}+S\right)\right]_k,$$

удобному для минимизации по аксиоме дизъюнкции в НДФ

$$Q_{k+1}(1) = \left(\overline{S}Q + R\overline{Q}\right)_k.$$

Применение теоремы Деморгана позволяет синтезировать из полученного решения также структурную формулу для инверсного выхода

$$\overline{Q}_{k+1}(1) = \left(\overline{\overline{S}Q + R\overline{Q}}\right)_k,$$

что соответствует после преобразования виду в НКФ

$$\overline{Q}_{k+1}(1) = \left[\left(S+\overline{Q}\right)\left(\overline{R}+Q\right)\right]_k.$$

Синтез в НКФ $F(0)$ организуют из произведений нулевых $f_j(0)$ функций, составленных из макстермов. По таблице (см. $3F(T)$) инверсному выходу $\overline{Q}$ в НКФ соответствует произведение

$$\overline{Q}(0) = f_0(0)f_2(0)f_3(0),$$

где $f_0(0) = S + R + \overline{Q}$, $f_2(0) = S + \overline{R}$, $f_3(0) = \overline{S} + \overline{R} + Q$.

Подставляя макстермы в произведение, получим структурную формулу на $(k + 1)$-м состоянии

$$\overline{Q_{k+1}}(0) = \left[\left(S+R+\overline{Q}\right)\left(S+\overline{R}\right)\left(\overline{S}+\overline{R}+Q\right)\right]_k.$$

Для упрощения преобразований используем теорему Деморгана и аксиому дизъюнкции:

$$\overline{Q_{k+1}}(0) = \left[\overline{\overline{S}\,\overline{R}Q + \overline{S}R(Q+\overline{Q}) + SR\overline{Q}}\right]_k.$$

После группировки подобных членов с учетом аксиомы дизъюнкции находим структурные формулы для инверсного выхода

$$\overline{Q}_{k+1}(0) = \left(\overline{\overline{S}Q + R\overline{Q}}\right)_k$$

и прямого в НДФ

$$Q_{k+1}(0) = \left(\overline{S}Q + R\overline{Q}\right)_k.$$

Теорема Деморгана преобразует выражение для инверсного выхода в минимизированную структуру НКФ

$$\overline{Q}_{k+1}(0) = \left[\left(S+\overline{Q}\right)\left(\overline{R}+Q\right)\right]_k.$$

Анализ структурных формул в НДФ для прямых выходов и НКФ для инверсных выходов показывает тождественность структурных формул

$$\begin{cases} Q_{k+1}(1) = Q_{k+1}(0); \\ \overline{Q}_{k+1}(1) = \overline{Q}_{k+1}(0). \end{cases}$$

Тождественность форм в НДФ $F(1)$ и НКФ $F(0)$ доказывает правильность синтеза в частности и проектирования в целом структурных формул.

Таким образом, метод структурных формул реализует алгоритм их проектирования при тождественности анализируемых логических образов исследуемой и эквивалентной функций, синтезированных в НДФ и НКФ.

Анализ структурных формул методом единиц и нулей основан на тождественности эквиваленту синтезируемой таблицы состояния исследуемой структуры, систематизирующей в вектор результаты подстановок логических термов при последовательной итерации адресного пространства. Результирующие термы находят по операторам счисления в НДФ или НКФ, ИЛИ-НЕ или И-НЕ при замене логическими единицами и нулями аналогичных переменных исследуемой функции.

М10 в НДФ рассмотрим на примере анализа структурной формулы $Q_{k+1}(1)$ синхронного *RS*-триггера для прямого выхода

$$Q_{k+1}(1) = \left(\overline{S}Q + R\overline{Q}\right)_k .$$

Подготовим для анализа М10 таблицу состояния, синтезированную по входу стандартным образом: линейно возрастающими адресами с нуля до трех в двоичном коде (см. рис. 2.7). Таблица дифференцировала сложную задачу на четыре аналогичных итераций по произвольным адресам. В первой строке по нулевому адресу $\{S, R\}=\{0, 0\}$ на позиции входных переменных S и R в структурную формулу $Q_{k+1}(1)$ подставляют логические нули:

$$Q_{k+1}(1) = \left(\overline{S}Q + R\overline{Q}\right)_k = Q^*(1) = \overline{0}Q + 0\overline{Q} .$$

S_k	R_k	Q^*_{k+1}	$(\overline{S}Q + R\overline{Q})_k$	$= Q^*(1)$
0	0	Q_k	$\overline{0}Q + 0\overline{Q}$	$= 1Q + 0$
1	0	0	$\overline{1}Q + 0\overline{Q}$	$= 0Q + 0$
0	1	1	$\overline{0}Q + 1\overline{Q}$	$= Q + \overline{Q}$
1	1	$\overline{Q}_k$	$\overline{1}Q + 1\overline{Q}$	$= 0Q + \overline{Q}$

Рис. 2.7 Синтезируемая таблица состояния

По операторам инверсии получим $\overline{0} = 1$, конъюнкции – $1Q = Q$ и $0\overline{Q} = 0$, а дизъюнкции $Q + 0 = Q$. Результат исчисления занесем в первую строку выходного столбца Q^*_{k+1}. По аналогии проведем счисления для других адресов и результаты систематизируем в таблицу состояния (рис. 2.7).

Сопоставим исследуемую таблицу (рис. 2.7) с эквивалентной (табл. 2.1, $3F(T)$) для оценки их тождественности. Из анализа следует тождественность исследуемой и эквивалентных таблиц $Q^*_{k+1} = Q_{k+1}$, что подтверждает правильность анализа М10 в частности и проектирования в целом структурной формулы *RSC*-триггера в НДФ по прямому выходу.

Из анализа М10 структурной формулы в НДФ следуют закономерности метода аналогии. Закономерности наиболее очевидны в методах алгебры Буля при сопоставительном анализе таблицы состояния и структурной схемы. Сопоставим на примере *RSC*-триггера таблицу $3F(T)$ и структурную формулу в НДФ для прямого выхода

$$Q_{k+1}(1) = \left(\overline{S}\,\overline{R}Q + \overline{S}R + S\overline{R}\overline{Q}\right)_k .$$

Во второй строке таблицы по адресу $\{1, 0\}=S\overline{R}$ минтерма $Q_{k+1} = 0$. Минтерм $S\overline{R}$ не принадлежит структурной формуле единичной функции. Другие адреса и их минтермы q_j служат слагаемыми исследуемой функции $Q(1)$ и определяют ее вектор состояния Q_k, 1, $\overline{Q}_k$ соответственно минтермам $\overline{S}\,\overline{R}$, $\overline{S}R$, SR по адресам $\{0, 0\}$,$\{0, 1\}$, $\{1, 1\}$. Выявленную закономерность в НДФ систематизируем в итерационном алгоритме метода аналогии:

$$\text{если} \quad q_j \begin{Bmatrix} \in \\ \notin \end{Bmatrix} Q(1), \quad \text{то} \quad Q_{k+1} = \begin{cases} Q_k(1) \\ 0 \end{cases}.$$

Из алгоритма следует, что если минтерм q_j принадлежит единичной функции $Q(1)$, то по исследуемому адресу $Q_{k+1}= Q_k(1)$. Функция Q_{k+1} на $(k + 1)$-м шаге тождественна вектору состояния $Q_k(1)$, в простейшем случае – логической единице. В противном случае для $q_j \notin Q(1)$ функция Q_{k+1} тождественна логическому нулю.

Проиллюстрируем метод аналогии для нулевого адреса:

$$\{0, 0\} \to \overline{S}\,\overline{R}\,Q = q_0 \in Q(1), \quad \text{то} \quad Q_{k+1} = Q_k.$$

Адресу {0, 0} соответствует в НДФ минтерм $q_0 = \overline{S}\,\overline{R}\,Q$, который принадлежит единичной функции $Q(1)$, поэтому $Q_{k+1} = Q_k(1)$.

Для первого адреса{1, 0} $Q_{k+1} = 0$, так как

$$\{1, 0\} \to S\,\overline{R} = q_1 \notin Q(1).$$

Эта закономерность в неявной форме сохраняется для минимизированной структурной формулы в НДФ. Для прямого выхода Q синхронного *RSC*-триггера минимизированное решение имеет вид:

$$Q_{k+1}(1) = \left(\overline{S}Q + R\overline{Q}\right)_k.$$

Для нулевого адреса по алгоритму МА следует

$$\{0, 0\} \to \overline{S}\,\overline{R} \to \overline{S}\,Q = q_0 \in Q(1), \quad \text{тогда} \quad Q_{k+1} = Q_k.$$

По первому адресу {1, 0} с минтермом $q_1 = S\,\overline{R}$ принадлежит двум слагаемым $\overline{S}\,Q$ и $R\,Q$ структурной формулы. При этом $Q_{k+1} = (Q + \overline{Q})_k$ зависит от суммы прямой и инверсной величины, что по аксиоме дизъюнкции соответствует логической единице, т.е. $Q_{k+1} = 1$.

Следовательно, по методу аналогии анализа структурной формулы в НДФ синтезируют таблицу состояния триггера по итерационному алгоритму при последовательной адресации. Функция тождественна вектору состояния для минтерма, составляющего формулу, в противном случае выходу соответствует логический нуль.

2.3.2 Структурная схема

Структурную схему в интегральном исполнении синтезировать по методу аналогии рационально по структурной формуле [15, 29]. Сущность метода [15] заключается в замене структур (операторов И, ИЛИ, НЕ) и связей (входных *S*, *C*, *R* и выходных *Q*, $\overline{Q}$ переменных) математических образов $F(\Phi)$ эквивалентными им в топологии $F(R)$ структурами (элементами &, 1, $\overline{A}$) и связями (сигналами из последовательности потенциалов низкого и высокого уровня). Последовательностную интегральную схему можно проектировать в НДФ $F(1)$ и НКФ $F(0)$, И-НЕ $F(\overline{\&})$ и ИЛИ-НЕ $F(\overline{1})$ по аналогии с комбинационными ИС, от которых *RSC*-триггеры отличаются наличием обратных связей.

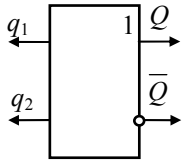
Синтез структурной схемы последовательностного элемента приведем на примере проектирования *RSC*-триггера по структурной формуле в НДФ:

$$Q_{k+1}(1) = \left\{C\left(\overline{S}\,Q + R\,\overline{Q}\right)\right\}_k.$$

Формула состоит из двух слагаемых, что соответствует замене оператора дизъюнкции логическим элементом ИЛИ (1), который связан с прямым Q и $\overline{Q}$ инверсным выходами (рис. 2.8). Входы логического сумматора объединяют сигналы минтермов q_1 и q_2. При этом в первом ранге НДФ выполняется суммирование

$$\begin{cases} Q = q_1 + q_2; \\ \overline{Q} = \overline{q_1 + q_2}. \end{cases}$$

Второй ранг схемы состоит из двух конъюнкторов &, формирующих минтермы q_1 и q_2 за счет логического умножения входных C, S, R и выходных Q, $\overline{Q}$ сигналов (переменных) $q_1 = C\overline{S}Q$, $q_2 = CR\overline{Q}$. Каждый из элементов & – трехвходовой, так как минтермы q_1, q_2 организованы из трех переменных. Соединим входы конъюнкторов & согласно q_1 и q_2. Входы первого элемента формируют q_1 при объединении первого входа с выходом Q, соединении второго проводника через инвертор «○» к прямому статическому S входу и подключении третьего входа к тактовому входу C (рис. 2.9). Первый вход второго конъюнктора & подключим к синхронизирующему входу C, второй проводник используем как инверсный статический R вход, а третий вход объединим с инверсным $\overline{Q}$ выходом для реализации минтерма q_2 (рис. 2.9). По методу аналогии на выходах конъюнкторов выполняются функции умножения



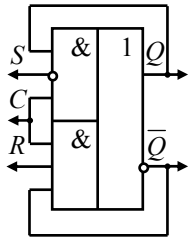


Рис. 2.8 Первый ранг схемы

Рис. 2.9 Интегральная схема в НДФ

$$q_1 = C\overline{S}Q \quad \text{и} \quad q_2 = CR\overline{Q},$$

а на прямом Q выходе RSC-триггера реализуется релейная функция переключения

$$Q(1) = C\overline{S}Q + CR\overline{Q},$$

эквивалентная исходной структурной формуле $Q_{k+1}(1)$ в НДФ.

2.3.3 Таблица состояния

Синтезируем таблицу состояния RSC-триггера в процессе анализа методом единиц и нулей М10 структурной схемы. Для этого сформируем адресное пространство, соответствующее трем переменным (входам) при $C = 1$. По статическим выходам S, R заполним таблицу дешифратора стандартным образом в двоичном коде с линейным возрастанием адресов от нуля $\{S, R\} = \{0, 0\}$ до трех $\{1, 1\}$. Выходной столбец таблицы мультиплексора в исходном состоянии пустой, который заполняется при итерации адресного пространства в процессе анализа структурной схемы М10 (рис. 2.9, 2.10). Таблица состояния (рис. 2.11) свела сложную задачу к решению четырех аналогичных итераций. На рис. 2.10 по входу S приведена последовательность инверсных значений второго столбца таблицы, так как $\overline{S} = \{1, 0, 1, 0\}$. На инверсном R статическом входе сформирована последовательность $R = \{0, 0, 1, 1\}$ третьего столбца входной таблицы (рис. 2.9). Тактовый C вход открыт единичным потенциалом, а на входах обратных связей изображены переменные векторы состояния Q и $\overline{Q}$.

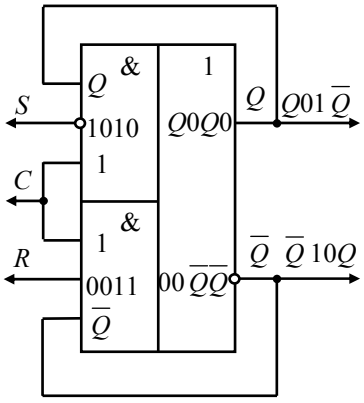


Рис. 2.10 Анализ схемы М10

C	S	R	Q^*_{k+1}
1	0	0	Q_k
	1	0	0
	0	1	1
	1	1	$\overline{Q_k}$

Рис. 2.11 Синтез таблицы М10

По нулевому адресу $\{S, R\} = \{0, 0\}$ на входах первого (верхнего) элемента & присутствуют сомножители $\{Q, \overline{S}, C\} = \{0, 1, 1\}$, результатом произведения которых $Q\overline{S}C = Q \cdot 1 \cdot 1$ является вектор состояния Q. На выходе второго конъюнктора формируется 0, полученный при перемножении $CR\overline{Q} = 1 \cdot 0 \cdot Q$ (см. рис. 2.10). Дизъюнктор 1 складывает в унитарном логическом пространстве сигналы конъюнкторов
$\{q_1, q_2\} = \{Q, 0\}$. При логическом сложении на выходе Q появляется прямое состояние вектора Q, так как $q_1 + q_2 = Q + 0 = Q$. На другом выходе $\overline{Q}$ устанавливается инверсное состояние $\overline{Q}$, $\overline{q_1 + q_2} = \overline{Q}$. Результаты решения по адресу $\{0, 0\}$ зафиксируем в первой строке выходного столбца $Q^*_{k+1} = Q_k$ (см. рис. 2.11).

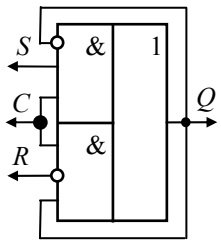
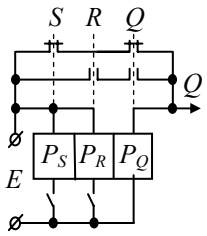
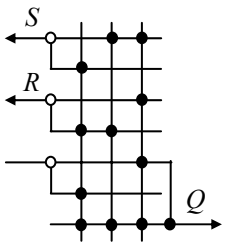
Результаты по другим адресам приведены на рис. 2.10 в виде последовательностей конъюнкции $q_1 = \{0, Q, 0\}$ и $q_2 = \{0, \overline{Q}, \overline{Q}\}$ на выходах элементов & и дизъюнкции $Q = \{0, 1, \overline{Q}\}$ и $\overline{Q} = \{1, 0, Q\}$ прямого Q и инверсного $\overline{Q}$ состояний на выходах элемента 1. Последовательность значений прямого вектора состояния Q систематизирована в выходном столбце $Q^*_{k+1} = \{0, 1, \overline{Q}\}_k$ таблицы состояния (см. рис. 2.11). Сопоставительный анализ синтезированной (см. рис. 2.11) и эквивалентной (табл. 2.1, $3F(T)$) таблиц состояния показывает их тождественность, откуда следует, что структурная схема (см. рис. 2.9, 2.10) синтезирована в частности и спроектирована в целом верно.

Таким образом, *RSC*-триггер проектируют подобно комбинационным схемам в комплексном представлении схемо- и мнемотехники, математики и физики с отличительными особенностями структурных схем и формул, таблиц состояния и временных диаграмм последовательностных элементов памяти.

2.4 РАЗВИТИЕ *RSC*-ТРИГГЕРОВ

Вектор развития *RSC*-триггеров направлен на повышение эффективности метрологических и технологических характеристик в процессе совершенствования схемных решений по упорядоченности и универсальности. Упорядоченность достигается заменой комбинаторных ИС решениями в релейной и матричной логике (табл. 2.3) за счет технологической и информационной интеграции базисных структур по пути ИС – СИС – БИС. Универсальность определяется дополнительными возможностями функции переключения при устранении неопределенных состояний *RS*-триггера, который заменяют синхронным *RSC*-триггером с унитарным и бинарными состояниями (см. табл. 2.4).

2.3 *RSC*-триггеры

	1 ИС	2 РЛ	3 ПЛМ
Схемы $F(R)$			
Таблицы $F(T)$	S | R | Q_{k+1} 0 | 0 | Q_k 1 | 0 | 1 0 | 1 | 0 1 | 1 | $\overline{Q_k}$	S | R | Q_{k+1} 0 | 0 | $\overline{Q_k}$ 1 | 0 | 0 0 | 1 | 1 1 | 1 | Q_k	S | R | Q_{k+1} 0 | 0 | $\overline{Q_k}$ 1 | 0 | 1 0 | 1 | 0 1 | 1 | Q_k
Формулы $F(\Phi)$	$\boldsymbol{Q_{k+1} = [C(\overline{R}Q + S\overline{Q})]_k}$ **если** $\boldsymbol{C = \begin{cases}0\\1\end{cases}}$, **то** $\boldsymbol{Q_{k+1} = \begin{cases}0\\(SRQ)_k\end{cases}}$	$\boldsymbol{Q_{k+1} = (\overline{S}\,\overline{Q} + RQ)_k}$ $\{0,0\} \Rightarrow \overline{0}\,\overline{Q} + 0Q = \overline{Q}$ $\{1,0\} \Rightarrow \overline{1}\,\overline{Q} + 0Q = 0$ $\{0,1\} \Rightarrow \overline{0}\,\overline{Q} + 1\overline{Q} = 1$ $\{1,1\} \Rightarrow \overline{1}\,\overline{Q} + 1\overline{Q} = Q$	$\boldsymbol{Q_{k+1} = [C(\overline{S}\,\overline{R}\,\overline{Q} + S\overline{R} + SRQ)]_k}$ $\boldsymbol{C = 1}$

В табл. 2.3 систематизированы *RSC*-триггеры по упорядоченности структур: в комбинаторике 1ИС (первый столбец), релейной логике 2РЛ и матричной логике на программируемых логических матрицах 3ПЛМ. Решения в комбинаторной, релейной, матричной логике приведены в основных формах представления функций: в виде схем $F(R)$ и таблиц $F(T)$, формул $F(\Phi)$ и диаграмм $F(\varepsilon)$. Проектирование триггеров аналогично комбинационным схемам, но следует учитывать обратные связи и вызванные ими кинетику и динамику векторов состояния.

2.4.1 Релейная логика

В релейной логике 2РЛ схема триггера $F(R)$ реализует обратную связь катушкой реле P_Q вместо резистивной нагрузки комбинационных схем [12] и группой нормально закрытых $\overline{Q}$ или открытых Q контактов. Контакты обратной связи проектируют по методам аналогии, соответствующим синтезу и анализу комбинационных схем релейной логики. Релейную схему синтезируют по методам аналогии топологии $F(R)$, мнемоники $F(T)$ и физики $F(\varepsilon)$ при проецировании планов таблицы состояния или тождественных ей временных диаграмм на матрицу релейной структуры $F(R)$. По таблицам переходов организуют замену логических нулей (нулевых потенциалов) и единиц (потенциалов высокого уровня) соответствующими им нормально закрытым и открытым контактами. К катушке обратной связи Q (или $\overline{Q}$) подключают в НДФ единичные минтермы, а в НКФ – нулевые макстермы. По структурным формулам $F(\Phi)$ синтезируют схемы $F(R)$ соответствующим методом булевой алгебры, основанной на эквивалентности математики, логики и топологии.

Анализ синтезированных образов схемо- и мнемотехники, математики и физики *RSC*-триггеров адекватен анализу комбинационных схем в релейной логике методами делителей напряжения и токов комбинаторики, единиц и нулей и структурных формул алгебры Буля, аналогии и эквивалентности ПЛМ.

2.4.2 Матричная логика

Матричная логика 3ПЛМ схемы триггера $3F(R)$ отличается от комбинационных топологий [13, 29] введением обратных связей Q, $\overline{Q}$ между дополнительной строкой Q в матрице И/НЕ-И и выходом Q [12, 13, 15] матрицы ИЛИ (см. 3ПЛМ, $F(R)$).Синтезируют триггер в ПЛМ методами аналогии [15] из таблицы состояния $3F(T)$ или семейства диаграмм $3F(\varepsilon)$, наложением их плана на скелетную матрицу И/НЕ-И/ИЛИ $3F(R)$. На матрице логического умножения «прожигают» знакоместа согласно минтермам: прямому a – на матрице И, инверсному $\overline{a}$ – на И-НЕ, а в матрице логического сложения ИЛИ программируют единичные минтермы. Знакоместа векторов состояний синтезируют аналогично минтермам комбинационных схем. Методом структурных формул синтезируют ПЛМ $3F(R)$ по формулам $F(\Phi)$ за счет тождественных преобразований математической логики и топологии. Анализ функции *RCS*-триггера во всех формах представления адекватен аналогичным операторам комбинационных схем по

эквивалентам. При тождественности анализируемой структуры эквиваленту положительно оценивают синтез в частности и проектирование в целом исследуемой формы функции.

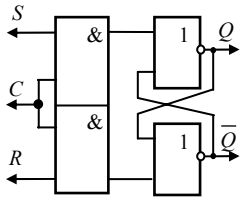
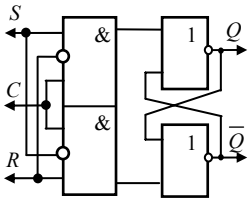
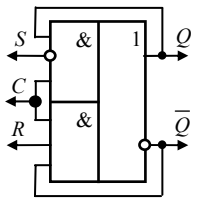
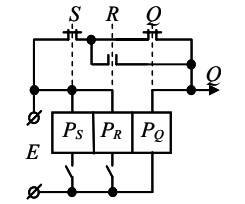
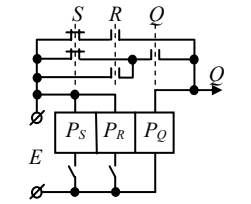
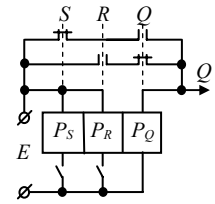
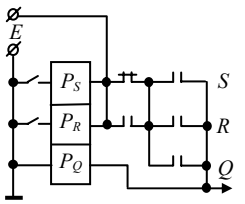
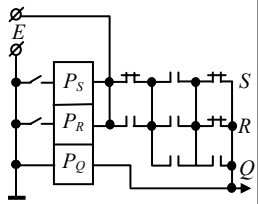
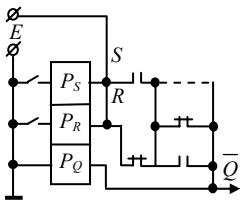
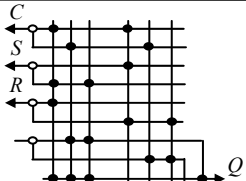
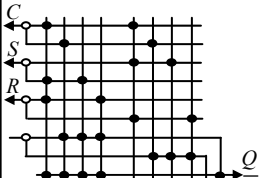
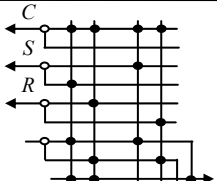
В табл. 2.3 систематизированы решения по упорядоченности от комбинаторной 1ИС до матричной 3ПЛМ логики на примере универсального *RSC*-триггера с бинарными состояниями на прямом и инверсном выходах. Таблица иллюстрирует повышение технологичности конструктивов от ИС к РЛ и ПЛМ, а также методов их проектирования от итерационного анализа ИС до эквивалентных образов ПЛМ. Из-за различных метрологических характеристик (точность, надежность, быстродействие) релейная логика рациональна в нано- и мегатехнологиях информационной и энергетической электроники. Для жестких структур предпочтительны ИС, а ПЛМ – основа гибкой архитектуры СИС, БИС микропроцессорных средств с высокими оперативностью, надежностью и точностью.

2.4.3 Синхронные триггеры

Табл. 2.4 иллюстрирует развитие универсальности *RSC*-триггеров от синхронного с неопределенными состояниями 1ИС через унитарный с фиксированными состояниями 2ИС к универсальному с бинарными состояниями, управляемыми по программе 3ИС. В таблице также систематизировано по строкам совершенствование структур по их упорядоченности от комбинаторики интегральных схем ИС $F(1)$ до упорядоченных в адресное пространство архитектур матричной логики МЛ $F(1)$ через схемы релейной логики РЛ в НДФ $F(1)$ и НКФ $F(0)$. Сущность информационной технологии проектирования последовательностных схем рассмотрена выше, а ниже анализируются решения по расширению адресного пространства эффективного функционирования статических триггеров.

RCS-триггеры в базисах И-НЕ $F(\bar{\&})$ и ИЛИ-НЕ $F(1)$ имеют одно из четырех состояний неопределенности, инициирующее состязание на лидерство переключения, которое запрещают программно или аппаратно. Для расширения адресного пространства предложен [2, 29, 30, 63] синхронный *RCS*-триггер за счет тактирования статических *R*, *S*-входов дополнительным входом синхронизации *C* (см. табл. 2.4, столбец 1).

2.4 *RSC*-триггеры

	1 Синхронный	2 Унитарный	3 Бинарный
ИС $F(1)$	 $\begin{cases} Q_{k+1}=[(\bar{C}+\bar{S})(CR+Q)]_k \\ \bar{Q}_{k+1}=[(\bar{C}+\bar{R})(CS+\bar{Q})]_k \end{cases}$	 $\begin{cases} Q_{k+1}=(C\bar{S}R+\overline{CS\bar{R}}Q)_k \\ \bar{Q}_{k+1}=(CS\bar{R}+\overline{C\bar{S}R}\bar{Q})_k \end{cases}$	 $Q_{k+1}=[C(\bar{S}Q+R\bar{Q})]_k$
РЛ $F(1)$	 $Q_{k+1}=[\bar{S}(R+Q)]_k$	 $Q_{k+1}=[\bar{S}R+(\bar{S}+R)Q]_k$	 $Q_{k+1}=(\bar{S}Q+R\bar{Q})_k$
РЛ $F(0)$	 $Q_{k+1}=[(\bar{S}+R)(S+R+Q)]_k$	 $Q_{k+1}=[(S+R+Q)(\bar{S}+R)(\bar{S}+\bar{R}+Q)]_k$	 $\bar{Q}_{k+1}=[(S+\bar{Q})(\bar{R}+Q)]_k$
МЛ $F(1)$			

Задача решена введением коммутатора на двух входовых элементах И (см. 1ИС) в НДФ $F(1)$ или двух элементов ИЛИ-НЕ в базисе $F(\overline{\&})$. Из схемы синхронного триггера (см. 1ИС) следует система структурных формул для выходов F_i (i = 1, 2) коммутатора:

$$F_1 = CS,\ F_2 = CR,$$

для которых тождественна векторная таблица истинности (рис. 2.12).

Из анализа структурных формул и таблицы следует, что для четырех дополнительных адресов $\{C, S, R\}=\{0, X, X\}$ с нулевым потенциалом (логическим нулем) $C = 0$ для любых «X» значений статических входов $S = R = X$ состояние RCS-триггера не изменяется $Q_{k+1} = Q_k$, так как $F_1 = F_2 = 0$. При этом выходам соответствует система подобных уравнений:

$$\begin{cases} Q_{k+1}(\overline{1}) = \left(\overline{F_1 + \overline{Q}}\right)_k = Q_k; \\ \overline{Q}_{k+1}(\overline{1}) = \left(\overline{F_2 + Q}\right)_k = \overline{Q}_k. \end{cases}$$

При поступлении на вход C логической единицы (потенциала высокого уровня) выходы F_1 и F_2 тождественны статическим входам S и R (рис. 2.12), что также следует из структурных формул $\{F_1$ и $F_2\} = \{S, R\}$ для $C = 1$. Для различных комбинаций RCS-триггер функционирует стабильно, но как и в RS-триггере по адресу $F_1 = F_2 = 1$ наблюдается неопределенность $Q_k = \overline{Q}_k = 0$. Это очевидно из структурных формул, так как $Q_{k+1} = \left(\overline{1 + \overline{Q}}\right)_k = 0$ и $\overline{Q}_{k+1} = \left(\overline{1 + Q}\right)_k = 0$.

Следовательно, синхронный RCS-триггер расширяет адресное пространство за счет коммутации статических входов тактирующим входом, но состояние неопределенности не исключает.

Унитарный RCS-триггер (см. табл. 2.4, столбец 2) исключает неопределенность синхронного триггера за счет запрещения и замещения адреса состязаний (для приведенного примера $F_1 = F_2 = 1$) инверсным адресом ($F_1 = F_2 = 0$) с неизменным состоянием Q. Для этого модифицируют коммутатор RCS-триггера согласно таблице истинности (рис. 2.13),

C	F_1	F_2
0	0	0
1	S	R

Рис. 2.12 Таблица коммутатора

S	R	C	F_1	F_2
X	X	0	0	0
0	0	1	0	0
1	0	1	1	0
0	1	1	0	1
1	1	1	0	0

Рис. 2.13 Таблица истинности

учитывающей по адресу $\{C, S, R\} = \{1, 1, 1\}$ неопределенность формированием логических нулей $\{F_1, F_2\} = \{0, 0\}$. По таблице рационально синтезировать структурные формулы модифицированного коммутатора в НДФ:

$$F_1(1) = CS\overline{R},\ F_2(1) = C\overline{S}R,$$

что по методу структурных формул соответствует структурной схеме из двух трехвходовых элементов конъюнкции (см. табл. 2.4, 2ИС). В отличие от триггера с неопределенностью (1ИС) введены две связи,

трансформирующие адрес {*C*, *S*, *R*} = {1, 1, 1} с неопределенным состоянием в знакоместо {1, 0, 0} с заданным значением вектора состояния *Q*. При этом $F_1 = F_2 = 0$, так как $F_1 = CS\overline{R} = 1 \cdot 1 \cdot \overline{0} = 0$, $F_2 = C\overline{S}R = 1 \cdot \overline{1} \cdot 1 = 0$ и соответственно

$$\begin{cases} Q_{k+1}(\overline{1}) = \left(\overline{F_1 + \overline{Q}}\right)_k = \left(\overline{0 + \overline{Q}}\right)_k = Q_k; \\ \overline{Q}_{k+1}(\overline{1}) = \left(\overline{F_2 + Q}\right)_k = \left(\overline{0 + Q}\right)_k = \overline{Q}_k. \end{cases}$$

Из анализа таблицы (рис. 2.13) и схемы 2ИС *F*(1) видно, что кроме разнополярных адресов {*S*, *R*}={1, 0} (или {0, 1}) во всем адресном пространстве сохраняются неизменные унитарные состояния $Q_{k+1} = Q_k$. Различные значения тождественных структурных схем формул (табл. 2.4) доказывают множественность структур *RCS*-триггеров в комбинаторной, релейной и матричной логике.

2.5 УНИВЕРСАЛЬНЫЕ ТРИГГЕРЫ

Универсальным RCS-триггером является структура с бинарными (дуальными) состояниями *Q* и $\overline{Q}$ на прямом выходе (см. табл. 2.4, 3ИС), проектирование которого приведено выше (см. п. 2.4.3) и систематизировано в табл. 2.3 в основных формах представления релейной функции. Универсальность анализируемого триггера следует из его аналогии с *JK*-триггером.

2.5.1 *JK*-триггер

JK-триггер [13, 29, 30, 67] организуют за счет формальной замены обозначений статических входов *S* и *R* на токовый *J* и потенциальный *K*, при этом таблица состояния (табл. 2.4, 1*F*(*R*)) с учетом входа синхронизации *C* примет вид [13], приведенный на рис. 2.14. Соответственно с этим изменяются структурные формулы:

$$\begin{cases} Q_{k+1} = \left[\overline{C}Q + C\left(\overline{J}Q + K\overline{Q}\right)\right]_k; \\ \overline{Q}_{k+1} = \left[\overline{C}\,\overline{Q} + C\left(JQ + \overline{K}\,\overline{Q}\right)\right]_k, \end{cases}$$

что эквивалентно формам [29, 30]. Из таблицы (рис. 2.14) и значений входов $C = 0$ состояние триггера не $C = 1$, то универсальный триггер лам:

J	*K*	*C*	Q_{k+1}
X	*X*	0	Q_k
0	0	1	Q_k
1	0	1	0
0	1	1	1
1	1	1	$\overline{Q}_k$

Рис. 2.14 Таблица *JK*-триггера

представления динамического *JK*-триггера формул следует, что для любых «*X*» зна*J*, *K* при нулевом потенциале на входе изменяется: $Q_{k+1} = Q_k$, $\overline{Q}_{k+1} = \overline{Q}_k$. Если функционирует как *JK*-триггер по форму-

$$\begin{cases} Q_{k+1} = \left[C\left(\overline{J}Q + K\overline{Q}\right)\right]_k; \\ \overline{Q}_{k+1} = \left[C\left(JQ + \overline{K}\,\overline{Q}\right)\right]_k, \end{cases}$$

тождественным для *RSC*-триггера статического

(табл. 2.4, 3*F*). Структурная схема *JK*-триггера и семейство временных диастатическом триггере переключение из инициируют импульсом синхронизации *K*-входах согласно таблице (рис. 2.14) *JK*данной комбинации. В динамическом *JK*формируется для заданной адресации *J*, *K*нерации фронта (или среза) импульса наличием задержки, организованной импульсов, выполненным на логических 29, 30] в комбинаторике ИС и релейной

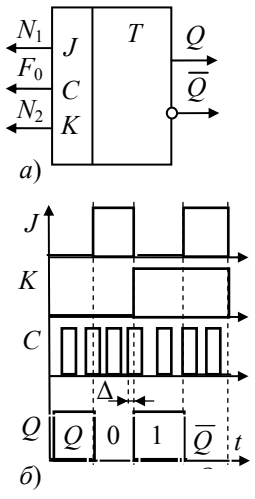


Рис. 2.15 *JK*-триггер:
а – структурная схема;
б – временные диаграммы

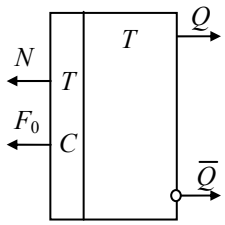
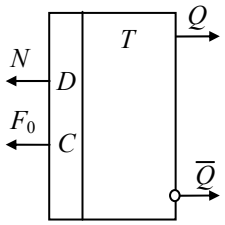
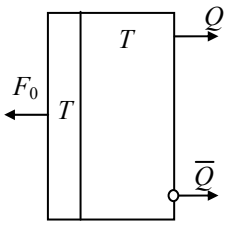
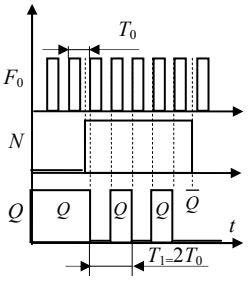
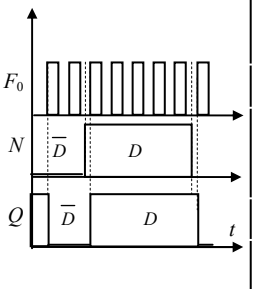
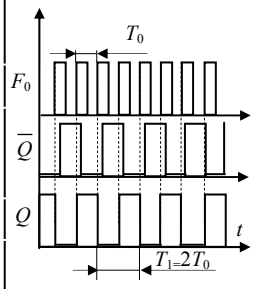
грамм приведены на рис. 2.15. В исходного в следующее состояние триггера в момент появления затриггере следующее состояние входов с задержкой в момент гесинхронизации *C*. Это обусловлено дополнительным формирователем элементах, триггере или таймере [2, логике.

2.5.2 *TC*-триггер

TC-триггер [13, 29], синхронизируемый *T*-триггер (табл. 2.5, столбец 1), конструируют из универсального триггера при объединении токового *J* и потенциального *K* входов, последний называют синхронизирующим входом *T*. Для заданных условий $J = K = T$ структурные формулы *TC*-триггера имеют вид:

$$\begin{cases} Q_{k+1} = \left[\overline{C}Q + C\left(\overline{T}Q + T\overline{Q}\right)\right]_k; \\ \overline{Q}_{k+1} = \left[\overline{C}\,\overline{Q} + C\left(TQ + \overline{T}\,\overline{Q}\right)\right]_k, \end{cases}$$

2.5 *JK* - триггеры

	1 *TC*-триггер	2 *D*-триггер	3 *T*-триггер
F(*R*)			
F(*T*)	C | T | Q_{k+1} 0 | 0 | Q_k 1 | 0 | Q_k 0 | 1 | Q_k 1 | 1 | $\overline{Q}_k$	C | Q_{k+1} 0 | Q_k 1 | D_k	T | Q_{k+1} 0 | Q_k 1 | $\overline{Q}_k$
F(Φ)	$J = K = T$ $\begin{cases} Q_{k+1} = [\overline{C}Q + C(\overline{T}Q + T\overline{Q})]_k \\ \overline{Q}_{k+1} = [\overline{C}\,\overline{Q} + C(TQ + \overline{T}\,\overline{Q})]_k \end{cases}$	$J = K = D$ $\begin{cases} Q_{k+1} = (\overline{C}Q + CD)_k \\ \overline{Q}_{k+1} = (\overline{C}\,\overline{Q} + C\overline{D})_k \end{cases}$	$D_k = \overline{Q}_k; C = T$ $\begin{cases} Q_{k+1} = (\overline{T}Q + T\overline{Q})_k \\ \overline{Q}_{k+1} = (\overline{T}\,\overline{Q} + TQ)]_k \end{cases}$
F(ε)			

а программа функционирования определяется таблицей состояния (см. табл. 2.5, 1*F*(*T*)). Из анализа этих форм видно, что синхронизируемый *T*-триггер сохраняет исходное состояние $Q_{k+1} = Q_k$ для большинства комбинаций адресного пространства. Переключение в инверсное состояние $Q_{k+1} = \overline{Q}_k$ происходит в том случае, когда $T = 1$ при поступлении счетного импульса $C = 1$. Условное обозначение *TC*-триггера приведено в табл. 2.5, 1*F*(*R*), а его семейство временных диаграмм – там же 1*F*(ε). В адресном пространстве

$\{C, T\} = \{0, 1\}$ и $\{1, 1\}$ *TC*-триггер делит входную частоту периодом T_0 на два, т.е. $T_1 = 2T_0$, но из-за его относительной сложности счетчики и регистры проектируют на *T*- и *D*-триггерах.

2.5.3 *D*-триггер

D-триггер [13, 29, 30, 67] задержки (табл. 2.5, столбец 2) создают из *JK*-триггера при объединении статических входов через инвертор, т.е. $\bar{J} = K = D$. Данные *N* по входу *D* на выходе *Q* формируются с задержкой (см. табл. 2.5, 2*F*(ε)) только при появлении фронта (или среза) частоты F_0 на счетном входе *C*. Это регламентировано структурными формулами *D*-триггера:

$$\begin{cases} Q_{k+1} = \left(\overline{C}\, Q + CD\right)_k; \\ \overline{Q}_{k+1} = \left(\overline{C}\, \overline{Q} + C\overline{D}\right)_k \end{cases}$$

при замене входов $\bar{J}$ и *K* входом задержки *D* данных *N*. Например, для прямого выхода, при соответствующей замене в структурной формуле *JK*-триггера, получим

$$Q_{k+1} = \left[\overline{C}Q + C\left(DQ + D\overline{Q}\right)\right]_k .$$

После приведения подобных, с учетом аксиомы дизъюнкции, находим

$$Q_{k+1} = \left[\overline{C}Q + CD\left(Q + \overline{Q}\right)\right]_k = \left(\overline{C}Q + CD\right)_k .$$

Структурные формулы и схемы *D*-триггера систематизированы в 2*F*(Ф) и 2*F*(*R*) табл. 2.5, там же приведена таблица состояния 2*F*(*T*) в векторной форме. Из форм представления следуют правила работы
D-триггера, реализуемые в регистре:

$$Q_{k+1} = D_k \quad \text{при} \quad C = F_0 = 0\ 1.$$

Триггер переключится в состояние $D = \{0, 1\}$, если на счетном входе *C* генерируется фронт импульса частоты F_0. Выявленная закономерность обусловлена алгоритмом функционирования *D*-триггера:

$$\text{если} \quad C = F_0 = \begin{cases} 0 \\ 1 \end{cases}, \quad \text{то} \quad Q_{k+1} = \begin{cases} Q_k \\ D_k \end{cases}.$$

Правило работы *D*-триггера позволяет копировать данные *N* со входа *D* на выход *Q* каждым синхроимпульсом (см. 2*F*(ε)). На выходе *Q* значение не изменяется, а подтверждается в каждом $(k + 1)$-м состоянии при $C = 1$, если информация *N* на входе *D*-задержки постоянна. Правило запоминают по мнемонической аналогии с пословицей: «Солдат спит – служба идет», а применяют его при программировании последовательностных ИС, СИС, БИС матричной логики.

2.5.4 *T*-триггер

T-триггер [13, 29, 30, 63] со счетным входом в комбинаторике ИС (см. табл. 2.5, столбец 3) проектируют из *D*-триггера (2*F*(*R*)) при введении жесткой обратной связи между входом *D* и выходом $\overline{Q}$ ($D = \overline{Q}$) и переименовании синхронизирующего входа *C* на *T*, т.е. $C = T$. В этом случае структурные формулы *D*-триггера превращаются в систему уравнений *T*-триггера:

$$\begin{cases} Q_{k+1} = \left(\overline{T}\, Q + T\overline{Q}\right)_k; \\ \overline{Q}_{k+1} = \left(\overline{T}\, \overline{Q} + TQ\right)_k, \end{cases}$$

а таблица состояния определяет переключение триггера каждым импульсом частоты F_0, поступающим на счетный вход (см. табл. 2.5, 3*F*(*T*)). Семейство временных диаграмм 3*F*(ε) иллюстрирует функциони-

рование *T*-триггера, организующего деление тактовой частоты F_0 на два: $F_1 = F_0/2$, а его структурная схема приведена в сегменте 3*F*(*R*).

Из анализа работы счетного триггера следует закономерность его переключения, систематизированная в правила проектирования счетчика в комбинаторной логике:

$$Q_{k+1} = \overline{Q}_k \quad \text{при} \quad T = F_0 = 0\ 1.$$

T-триггер переключается в противоположное состояние каждым счетным импульсом тактовой частоты в момент формирования фронта. Закономерность следует из алгоритма функционирования *T*-триггера:

$$\text{если} \quad T = F_0 = \begin{cases} 0 \\ 1 \end{cases}, \quad \text{то} \quad Q_{k+1} = \begin{cases} Q_k \\ \overline{Q}_k \end{cases}.$$

Данное правило более жестко регламентирует функцию переключения за счет введения жесткой обратной связи $D_k = \overline{Q}_k$ и не позволяет по программе изменять состояние *T*-триггера в отличие от *D*-триггера задержки. На *T*-триггерах реализуют счетчики, поэтому они менее гибкие, чем регистры, создаваемые на *D*-триггерах с программным управлением информации по входу данных *D*.

Из сопоставительного анализа с *TC*-триггером очевидны преимущества *T*-триггера, обусловленные более простыми формами представления (см. табл. 2.5, 1*F* и 3*F*) в явном виде. Это позволяет минимизировать счетчики в комбинаторике ИС и проектировать более рациональные схемы последовательностных преобразователей.

Таким образом, *RSC*-триггеры развиваются по гибкости от синхронных и унитарных к универсальным *JK*-триггерам, а по упорядоченности – от комбинаторных и релейных ИС к матричной логике ПЛМ. Универсальные *JK*-триггеры за счет введения обратных связей организуют счетные *TC*- и *T*-триггеры для построения счетчиков, а также программируемый *D*-триггер для реализации регистров. По гибкости и универсальности статические триггеры модифицируются в динамические *JK*-, *D*- и *T*-триггеры, посредством временной задержки дополнительных формирователей импульсов для повышения метрологической эффективности ИС в комбинаторной и релейной логике. В процессе анализа развития триггеров в комбинаторике ИС выявлены закономерности функций переключения, систематизированные в правила анализа и синтеза универсальных триггеров, счетчиков и регистров в матричной логике ПЛМ микропроцессорных средств.

Выводы

1 Методы структурных формул МСФ, единиц и нулей М10 алгебры Буля анализируют структурные схемы и формулы в процессе синтеза таблицы состояния и семейства временных диаграмм в адресном пространстве топологии, мнемоники и логики по тождественности исследуемой формы эквиваленту образцовой меры.

2 Информационная технология проектирования микропроцессорных средств, кроме использования компьютеров, основана на информационных концепции и процессах, модели и обеспечении, принципах и методах, обусловленных информатизацией научно-технической революции.

3 Простые статические триггеры развиваются по универсальности в сложные динамические *T*-, *D*-, *JK*-триггеры от комбинаторных и релейных ИС к матричной логике ПЛМ, проектируются по информационной технологии комбинационных схем в комплексном представлении схемо- и мнемотехники, математики и физики с отличительными особенностями структурных схем и формул, таблиц состояния и временных диаграмм последовательностных элементов памяти.

3 СРЕДНИЕ ИНТЕГРАЛЬНЫЕ СХЕМЫ

Триггеры служат основой СИС в комбинаторной логике [1, 13, 25 – 30, 34, 37, 63, 66]. Это счетчики и регистры, буферные и запоминающие устройства, таймеры времени и делители частоты, линии за-

держки и генераторы импульсов. Последовательностные СИС в комбинаторике анализируют по правилам переключения универсальных триггеров, а синтезируют по устоявшимся определениям, интегрирующим закономерности конструирования управляемых преобразователей сигнала. Счетчики и регистры в комбинаторной логике СИС определены линейкой триггеров ИС, включенных по счетному входу соответственно последовательно и параллельно [13, 24].

Приведем синтез и анализ счетчиков и регистров из линейки *D*-триггеров ИС на примере проектирования делителей частоты с коэффициентом деления $N = 5$. Деление частоты организуют на интегрировании импульсов постоянной частоты F_0 за время, пропорциональное их числу. Это следует из определения частоты $f = N/T$, как числа импульсов N за время T. Принимая $f = F_0$, а $T = 1/F$, находим характеристики в виде обратной $F = F_0/N$ и прямой $T = NT_0$ зависимостей.

3.1 СЧЕТЧИКИ

Счетчик является функциональным преобразователем последовательностного типа и относится к базису СИС. В комбинаторной логике ИС счетчик представляют линейкой триггеров, соединенных последовательно по счетному входу, в отличие от регистров с параллельным включением универсальных триггеров. Последовательное соединение синхронных *T*-триггеров организует счет последовательности импульсов в двоичном коде $N_2 = \sum_{i=0}^{n-1} \xi_i 2^i$ посредством их суммирования, вычитания или реверса, поэтому счетчики классифицируют на суммирующие, вычитающие и реверсивные. Реверсивные счетчики объединяют в себе функции суммирования и вычитания последовательности импульсов. Вид арифметической операции счетчика зависит от способа переключения триггеров, организуемого подключением счетных входов C к выходам Q, $\overline{Q}$ предыдущих триггеров (рис. 3.1).

Конструируют динамические триггеры с переключением по фронту 0 1 (с нулевого логического потенциала на единичный) и срезу 1 0 с соответствующим обозначением на структурных схемах тактового входа C: «/» (▹) или «\» (◃). Последовательное соединение в линейку триггеров по счетному входу C можно организовать через прямые Q или инверсные $\overline{Q}$ выходы. Из четырех возможных комбинаций включения реализуют два вида суммирующих и вычитающих счетчиков со стандартной «+», «–» и нестандартной «∑», «Δ» архитектурой (см. рис. 3.1). Стандартную и нестандартную архитектуру определяет тактовый вход C переключением триггера соответственно по срезу (с единицы на нуль) и по фронту

C	0 1	1 0
Q	Δ	+
$\overline{Q}$	Σ	–

Рис. 3.1 Классификация счетчиков

(с нулевого на единичный уровень потенциала). В стандартной архитектуре суммирование «+» и вычитание «–» импульсов в счетчике обусловлено по срезу линейным законом возрастания и убывания с нулевой {00…0} (единичной) до единичной {11…1} (нулевой) комбинации в двоичном коде N_2. С нестандартной архитектурой счетчики создают на динамических триггерах, переключающихся по фронту. При нестандартном суммировании ∑ и вычитании Δ, в отличие от стандартных счетчиков, таблица состояния на половину такта (шага программы) опережает или отстает от классического решения. Если в комбинаторной логике вид счетчика определяют типы триггеров и схемы их включения, то матричную логику регламентирует архитектура, систематизирующая в таблице состояния программу соответствующего оператора исчисления.

3.1.1 Суммирующий счетчик

Счетчик синтезируют по его определению как линейку триггеров, включенных последовательно по счетному входу [13, 24]. Для определенности прямые выходы Q_i триггеров соединяем со счетными входами C_{i+1}, управляемыми по срезу импульсов. Инверсные выходы $\overline{Q}_k$ подключаем к входам задержки D_k для организации *T*-триггеров (рис. 3.2). Проанализируем работу счетчика методом единиц и нулей, а результаты систематизируем в таблицу состояния. Исходная таблица (рис. 3.3) содержит число столбцов, равное количеству исследуемых узлов счетчика со входа F_0 до выхода F, которые определяются выходами Q_i триггеров и счетным входом счетчика. Пусть начальные состояния

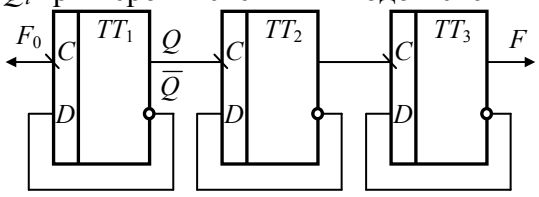


Рис. 3.2 Счетчик на *D*-триггерах

F_0	Q_1	Q_2	Q_3	N
0 1	0 0	0 0	0 0	0
0 1	1 1	0 0	0 0	1
0 1	0 0	1 1	0 0	2
0 1	1 1	1 1	0 0	3
0 1	0 0	0 0	1 1	4
0 1	1 1	0 0	1 1	5
0 1	0 0	1 1	1 1	6
0 1	1 1	1 1	1 1	7

Рис. 3.3 Таблица состояния счетчика

триггеров TT_i нулевые $Q_i = Q_{i+1} = 0$, что соответствует обнулению счетчика. Согласно правилам функционирования T-триггеров: статика отключена $S = R = 0$, введена аппаратно обратная связь $D_k = \overline{Q}_k$, поэтому достаточно анализировать условие появления среза импульсов тактовой частоты F_0. Условие переключения T-триггера регламентировано алгоритмом:

$$\text{если } F_0 = C = \begin{cases} 0\,1 \\ 1\,0 \end{cases}, \quad \text{то } Q_{k+1} = \begin{cases} Q_k \\ \overline{Q}_k \end{cases}.$$

При нулевом потенциале на входе счетчика, в исходном состоянии он обнулен (см. рис. 3.3, строка 1). Появлению импульса F_0 на счетном входе первого триггера TT_1 соответствует изменение нулевого потенциала на единичный, т.е. фронту 0 1, что не изменяет состояние $Q_{k+1} = Q_k$ счетчика. Все триггеры обнулены, так как первый триггер не переключался, при этом код N счетчика равен нулю $N = 0$ (см. строки 1 и 2 рис. 3.3). По окончанию первого импульса единичный потенциал изменяется на нулевой, что соответствует срезу импульса на счетном входе C первого триггера TT_1 счетчика. В момент появления среза импульса F_0 первый триггер переключается из нулевого в инверсное единичное состояние (см. рис. 3.3, строка 3) $Q_1 = 1$. Другие триггеры остаются в исходном нулевом состоянии, так как на счетных входах C второго триггера TT_2 формируется фронт импульса, а третьего триггера TT_3 – потенциал не изменяется. По четвертой строке (рис. 3.3) состояние счетчика не изменяется, так как на счетном входе C первого триггера TT_1 при появлении второго импульса F_0 генерируется фронт.

После второго импульса, по его срезу, триггер TT_1 переходит из единичного в нулевое состояние, инициируя на счетном входе C второго триггера TT_2 срез. Выполняется условие переключение триггера TT_2 из нулевого в единичное состояние, а на входе C третьего триггера появляется фронт, не изменяющий состояния триггера TT_3. Состояние счетчика остается прежним {0, 1, 0} при появлении третьего импульса тактовой частоты F_0, так как фронтом триггеры не переключаются. При этом бинарный код $\{Q_1, Q_2, Q_3\} = \{0, 1, 0\}$ счетчика равен в десятичном счислении двум $N = 2$, а предыдущее состояние {1, 0, 0} – соответствует одному $N = 1$.

Код счетчика увеличивается на единицу ($N = 3$) при переключении первого триггера в момент среза третьего импульса частоты F_0, так как $\{Q_1, Q_2, Q_3\} = \{1, 1, 0\}$. При этом второй и третий триггеры не изменяют состояния из-за отсутствия на их счетных входах среза импульса. Каждым импульсом тактовой частоты F_0 код счетчика по бинарному счислению увеличивается на единицу. Закономерность суммирования импульсов систематизирована в таблице состояния (см. рис. 3.3) в виде классической таблицы дешифратора двоичного кода. Таблица начинается с нулей и заканчивается единицами при чередовании нулей и единиц потенциалами импульсов тактовой частоты F_0 на счетном входе C счетчика и их разрядкой в два раза на выходах Q_i триггеров TT_i. Анализ выходного кода показывает его увеличение по срезу каждого импульса частоты F_0, поэтому исследуемый счетчик является суммирующим «+» стандартным счетчиком (см. рис. 3.2).

3.1.2 Делитель частоты

Делитель частоты на счетчике в комбинаторной логике ИС проектируют [24] по структурной схеме (см. рис. 3.2) и таблице состояния (рис. 3.3) с учетом коэффициента деления $N = 5$, по которому выбирают число разрядов Q_i (триггеров TT_i) счетчика и количество состояний таблицы переключений (см. рис. 3.3). Число разрядов счетчика пропорционально количеству триггеров и двоичному логарифму коэффициента N деления:

$$n = [\log_2 N],$$

где квадратные скобки обозначают округление до большего целого числа. Формула следует из равенства коэффициента деления двоичному коду $N = N_2 = 2^n$ и логарифмирования по бинарному основанию.

Для анализируемого примера $\log_2 5 = 2{,}32$, что соответствует округлению до трех $n = 3$, поэтому выбираем счетчик на трех триггерах (см. рис. 3.2). Трехразрядный счетчик емкостью $E = 2^3 = 8$ без дополнительных структур делит тактовую частоту на восемь, что видно из таблицы состояния (см. рис. 3.3).

Делитель частоты с заданным коэффициентом деления конструируют с использованием декодера, формирующего импульс управления состояниями счетчика при выявлении кода, тождественного коэффициенту деления. В простейшем случае декодер трансформируется в мультиплексор на многовходовом логическом элементе. Мультиплексор синтезируют по таблице счетчика, из которой выбирают любую последовательность из шести состояний, на единицу больше коэффициента деления. Для суммирующего счетчика рационально принять за исходное состояние нулевой код {0, 0, 0} и ограничить таблицу шестым состоянием при появлении числа пять {1, 0, 1} $N = 5$ (см. рис. 3.3). Таблица состояния для шести импульсов (см. табл. 3.1, $1F(T)$) создана по эквивалентной таблице (см. рис. 3.3). Для организации из шести состояний заданного коэффициента необходимо совместить в одно исходное и конечное состояния. Это соответствует системе уравнений:

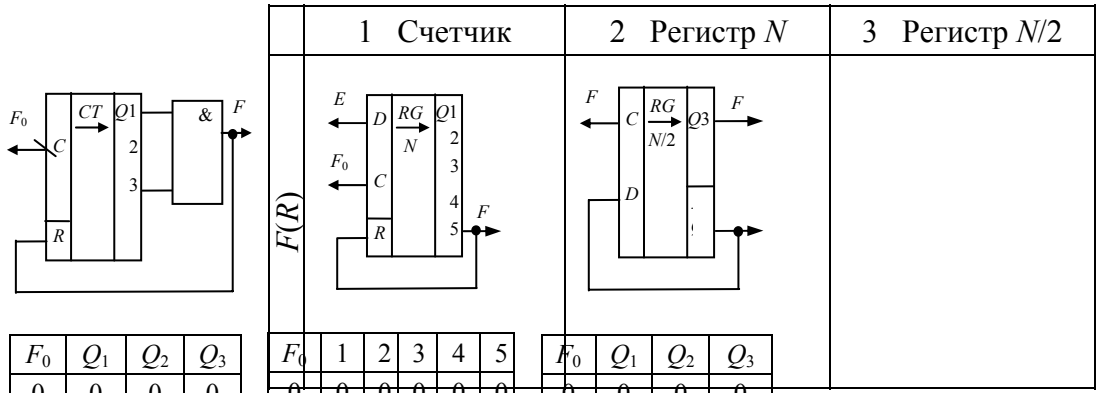
$$\begin{cases} F(0) = \overline{Q}_1 \overline{Q}_2 \overline{Q}_3; \\ F(5) = Q_1 \overline{Q}_2 Q_3, \end{cases}$$

регламентирующей обнуление триггеров счетчика $F(0)$ при выявлении кода $F(5)$. Решение с заданным коэффициентом деления может быть реализовано на тождественном элементе $N = \{1, 0, 1\} \equiv \{Q_1, \overline{Q}_2, Q_3\} =$
$= F(5)$. Мультиплексор, реализующий обнуление счетчика по функции $F(5)$, является трехвходовым логическим конъюнктором. Его выход соединяют с нулевым статическим входом R счетчика, а входы – с соответствующими выходами триггеров $\{Q_1, \overline{Q}_2, Q_3\}$. Решение можно минимизировать, исключив неизменное второе состояние $\overline{Q}_2$ (см. табл. 3.1, $1F(\Phi)$) для реализации функции $F(\&) = Q_1 Q_3$. Минимизированную функцию выполняет двухвходовый логический конъюнктор, обнуляющий счетчик по входу R при появлении единичных потенциалов на выходах первого и третьего триггеров (см. $1F(R)$).

Временные диаграммы $1F(\varepsilon)$ синтезируют по таблице $1F(T)$ в процессе анализа схемы счетчика $1F(R)$ методом алгебры Буля. Семейство временных диаграмм иллюстрирует деление на два входной частоты F_0 на выходе триггеров ($F_1 = F_0/2$, $F_2 = F_1/2$, $F_3 = F_2/2$) без дополнительных связей и устройств. Из диаграмм видно генерацию импульсов обратной связи на выходе мультиплексора & в момент среза пятого импульса при состоянии $\{Q_1, Q_2, Q_3\} = \{1, 0, 1\}$. Действительно, пятым импульсом изменяется состояние счетчика на единицу за счет переключения первого триггера TT_1 в единицу. При равенстве потенциалов $Q_1 = Q_3 = 1$ конъюнктор & формирует управляющий импульс, который обнуляет по входу R первый и третий триггеры и соответственно счетчик. Период T следования импульсов на выходе элемента & кратен пяти тактам $5T_0$ входной частоты F_0. Из равенства $T = 5T_0$ следует тождественность исследуемого коэффициента $N^* = T/T_0 = 5$ заданному эквиваленту $N = 5$, что соответствует правильности проектирования схемы делителя частоты на счетчике $1F(R)$ в основных его формах представления техники $1F(T)$ и науки $1F(\Phi)$, $1F(\varepsilon)$.

Таким образом, показано проектирование счетчика на примере управляемого делителя частоты в комбинаторной логике по правилам динамической работы универсальных триггеров базиса ИС.

3.1 Делители частоты



F_0	Q_1	Q_2	Q_3
0	0	0	0
1	1	0	0
2	0	1	0
3	1	1	0
4	0	0	1

F_0	1	2	3	4	5
0	0	0	0	0	0
1	1	0	0	0	0
2	1	1	0	0	0
3	1	1	1	0	0
4	1	1	1	1	0

F_0	Q_1	Q_2	Q_3
0	0	0	0
1	1	0	0
2	1	1	0
3	1	1	1
4	0	1	1

$F(T)$			
$F(\Phi)$			
$F(\varepsilon)$			

3.1.3 Алгоритм проектирования счетчика

1 Задают вид счетчика и тип универсальных триггеров, класс многофункционального преобразователя и коэффициент деления.

2 Выбирают число триггеров (разрядов счетчика) по коэффициенту деления.

3 Синтезируют обобщенную и комбинаторную структурные схемы счетчика по определению последовательного включения триггеров ИС.

4 Анализируют алгоритм функционирования счетчика по правилам переключения триггеров в динамическом режиме.

5 Систематизируют алгоритм исчисления в таблицу состояния счетчика.

6 Оценивают оператор исчисления на тождественность проектируемого счетчика эквиваленту по классификационной таблице.

7 Строят таблицу состояния с заданным коэффициентом деления по эквивалентной таблице счетчика.

8 Проектируют мультиплексор по начальному и конечному состояниям функции деления.

9 Конструируют многофункциональный преобразователь на счетчике с обратной связью на мультиплексоре.

10 Синтезируют методом аналогии семейство временных диаграмм и оценивают коэффициент деления анализируемой функции.

11 Сопоставляют коэффициент деления анализируемой и заданной функций на тождественность, при их неравенстве корректируют решение по пунктам 7 – 11.

12 Утверждают правильность проектирования многофункционального преобразователя в основных формах науки и техники при эквивалентности заданию синтезированного коэффициента деления.

Анализ алгоритма позволяет систематизировать закономерности проектирования счетчика в комбинаторной логике. В отличие от комбинационных СИС, проектируемых методами булевой алгебры по

математическим моделям, синтезируют последовательностный преобразователь СИС по правилам переключения триггеров базиса ИС, а анализируют оператор исчисления по эквивалентам таблиц классификации и состояния счетчиков. Синтезируют функцию счетчика в основных формах схемо- и мнемотехники, математики и физики, а оценивают эффективность проектирования по тождественности заданному эквиваленту синтезируемой функции, полученной в процессе анализа семейства временных диаграмм. Единственность временных диаграмм и их адекватность физике информационных процессов позволяет принять их за достоверный эквивалент, в отличие от множественности форм других представлений функций в науке и технике.

3.2 РЕГИСТРЫ

Программно управляемый цифровой преобразователь сигнала для регистрации импульсов в различных кодах счисления называют регистром [13, 24]. В отличие от счетчика, работающего в двоичном коде без дополнительных аппаратных средств, регистр реализует по программе множество форм счисления, включая и бинарный код. Регистры классифицируют по способам преобразования сигнала и хранения информации на время-, число-, кодоимпульсные последовательного, параллельного, смешанного действия. Регистры параллельного действия преобразуют информацию в кодоимпульсной форме по шинной структуре. Сдвиговые регистры последовательного действия оперируют с числоимпульсным кодом. Универсальные регистры смешанного действия инициируют время-импульсные преобразования при число- и кодоимпульсном вводе-выводе информации по магистрали.

В комбинаторной логике ИС регистр представляют линейкой триггеров, соединенных по счетному входу параллельно, в отличие от счетчиков, конструируемых последовательным включением динамических триггеров [24]. Поэтому точность, надежность и оперативность регистров выше метрологических характеристик счетчиков, имеющих низкую помехозащищенность из-за эффекта гонок при последовательном переключении сложных триггеров. Благодаря параллельному соединению триггеров регистры не критичны к характеристикам синхронизирующих импульсов и для них не имеет принципиального значения наличие фронта или среза. Для переключения разрядов регистра существен факт наличия импульса из последовательности тактовой частоты. Сдвиговые и универсальные регистры в базисе ИС реализуют на универсальных динамических *JK*- и *D*-триггерах, а параллельные регистры создают, как правило, на простых статических *RS*-, *RSC*- и *D*-триггерах.

Параллельные регистры [1, 13, 29, 30, 63, 67] служат основой запоминающих устройств интерфейсов памяти [41 – 43, 67] и ввода-вывода [3, 22, 30], а на динамических регистрах создают функциональные, временные и пространственные преобразователи. Сдвиговые регистры [13, 30, 67] являются базой микропроцессоров [7 – 21, 46 – 56] и знакогенераторов [22, 63], линий задержки [13] и программируемых таймеров [24], генераторов импульсов [13, 63] и декодеров сигнала [67]. Универсальные регистры применяют для проектирования программируемых портов и коммутаторов канала, цифровых компараторов и арифметико-логических устройств, дешифраторов кода и счетчиков импульсов [13]. Регистры уступают счетчикам только по емкости хранения информации, так как двоичный код наиболее компактный.

Регистр в комбинаторной логике синтезируют, как и счетчик [24], по определению в виде линейки триггеров, включенных параллельно по счетному входу. *D*-входы *i*-х триггеров соединяют с прямыми выходами $(i-1)$-х триггеров, состояние которых в *k*-м шаге определяет состояние регистра на $(k+1)$-м интервале времени, $Q_{k+1} = D_k$. За счет параллельного включения счетных входов $C_i = C_{i+1}$ запись в триггеры регистра происходит по импульсам тактовой частоты F_0 без дифференциации фронта или среза, принципиальных для счетчиков. Анализ работы триггеров регистра организуют по правилам их переключения при появлении импульса $C = F_0$ с учетом информации на входах задержки D_k, когда статические входы *S* и *R* отключены.

Схема регистра в комбинаторной логике из линейки *D*-триггеров приведена на рис. 3.4. Проанализируем работу регистра по правилам переключения *D*-триггеров и систематизируем решения в таблицу состояния (рис. 3.5) для трех разрядов Q_1, Q_2, Q_3 с нулевым исходным состоянием $\{Q_1, Q_2, Q_3\} = \{0, 0, 0\}$. Для нормальной работы регистра в динамическом режиме необходимо наличие в преобразуемом коде из нулей хотя бы одной единицы.

Логическую единицу сформируем на *D*-входе первого триггера TT_1 подключением его к потенциалу *E* высокого уровня (рис. 3.4). Если вход задержки D_1 соединить с нулевым потенциалом, то состояние регистра на $(k+1)$-м такте не изменяется, так как на выходах триггеров подтвердятся логические нули $D_{k,i} = D_{k,i+1} = 0$ при появлении импульсов синхронизации F_0.

При наличии на входе D потенциала высокого уровня во всех триггерах, кроме первого, состояние не изменяется из-за подтверждения тождественности нулю. В первый триггер TT_1 перепишется логическая единица со входа задержки $Q_{k+1} = D_k = 1$, а в регистре сформируется код $\{Q_1, Q_2, Q_3\}=\{1, 0, 0\}$ (см. рис. 3.5, строка 2). Вторым импульсом подтвердятся состояния логической единицы и нуля в первом и третьем триггерах, а во второй триггер перекопируется потенциал высокого уровня с его выхода задержки. Третья строка (рис. 3.5) иллюстрирует структурный сдвиг вправо логической единицы с формированием кода $\{1, 1, 0\}$. Третий импульс подтверждает единичные состояния первого и второго триггеров и записывает логическую единицу в третий триггер, устанавливая в регистре код $\{1, 1, 1\}$ (см. рис. 3.5, строка 4).

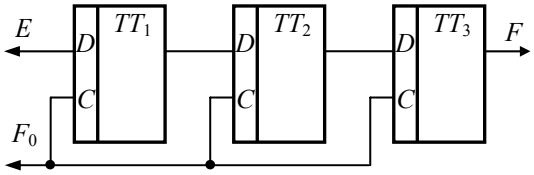


Рис. 3.4 Схема регистра

F_0	Q_1	Q_2	Q_3
0	0	0	0
1	1	0	0
2	1	1	0
3	1	1	1

Рис. 3.5 Таблица состояния

Следующие импульсы частоты F_0 не изменяют состояние регистра, так как в триггерах подтверждается логическая единица, присутствующая на их входах задержки.

3.2.1 Управляемый делитель частоты

Делители частоты на регистре конструируют [24] введением следящей обратной связи с выхода последнего триггера на статические входы регистра или на вход задержки D первого триггера. Для приведенного примера с позиционным единичным кодом мультиплексор вырождается в логический повторитель последнего разряда, минимизированную функцию коммутации в исходное состояние которого выполняет последний триггер. При выборе исходного состояния нулевым $\{0, 0, 0\}$, а конечного – единичным $\{1, 1, 1\}$, система уравнений имеет вид

$$\begin{cases} F(0) = \overline{Q}_1\overline{Q}_2\overline{Q}_3; \\ F(3) = Q_1Q_2Q_3, \end{cases}$$

что регламентирует обнуление $F(0)$ всех триггеров при появлении единичного состояния $F(3)$. Но состояние переполнения регистра определяет выход Q_3 последнего триггера TT_3, поэтому трехвходовый конъюнктор трансформируется в одновходовый повторитель в виде обратной связи с выхода F на статический вход обнуления R регистра (см. табл. 3.1, $2F(R)$). При этом минимизированная функция обнуления имеет вид $F(\&) = Q_3 = R$, а на выходе формируется частота $F = F_0/N$. Учитывая линейную зависимость единичного кода N от числа разрядов n (триггеров), видно, что коэффициент деления N преобразователя на регистре пропорционален числу разрядов или количеству триггеров $n = N$.

Управляемый делитель частоты с коэффициентом $N = 5$ приведен в основных формах науки и техники в табл. 3.1, $2F$. Регистр делителя синтезирован на пяти триггерах TT_i, $i = \overline{1, 5}$ с выходами Q_i и обратной связью $F(\&) = Q_5 = R$ (см. $2F(R)$ и $2F(\Phi)$). Таблица состояния $2F(T)$ систематизирует алгоритм деления на пять аналогично рассмотренному примеру (рис. 3.5) с коэффициентом $N = 3$. Семейство временных диаграмм $2F(\varepsilon)$, проектируемое методами аналогии по таблице состояния $2F(T)$, иллюстрирует деление тактовой частоты F_0 на заданный коэффициент $N = 5$. Это следует из оценки периода T следования импульсов с частотой F на Q_i –х выходах регистра, соответствующего пяти тактам $5T_0 = T$ частоты синхронизации F_0. Тождественность исследуемого коэффициента $T/T_0 = N^* = 5$ заданному $N = 5$ говорит о правильности проектирования делителя частоты на регистре в комбинаторной логике.

К достоинствам управляемого преобразователя относятся простота проектирования сдвигового регистра на n триггерах и программируемого делителя частоты, а недостатками являются малая емкость и аппаратурная избыточность. Исключает указанные недостатки схема кольцевого делителя на регистре из $n/2$ триггеров.

3.2.2 Кольцевой регистр

Кольцевой регистр организуют [24] из сдвигового регистра (см. рис. 3.4) при замыкании входа задержки D первого триггера TT_1 на инверсный $\overline{Q}$ выход последнего триггера TT_3. Если кольцевую обратную связь создают через прямой Q выход, необходимо использовать исходный код с наличием последовательности хотя бы одной единицы и нуля. Это обусловлено правилом программного управления D-триггера $Q_{k+1} = D_k$, подтверждающим тождественность информации при равенстве потенциалов и копировании информации при наличии инверсных логических уровней.

Схема кольцевого регистра с инверсной обратной связью приведена (рис. 3.6) на примере трех D-триггеров TT_i, $i = \overline{1, 3}$. Для сопоставления кольцевого регистра со сдвиговым синтезируем его таблицу состояния, которую сравним с таблицей сдвигового регистра. Выберем тождественные исходные состояния – нулевые, $\{Q_1, Q_2, Q_3\} = \{0, 0, 0\}$ на первом шаге, и их увеличение каждым тактовым импульсом частоты F_0. Первым импульсом в первый триггер TT_1 перепишется логическая единица (рис. 3.7) с инверсного выхода последнего триггера TT_3 регистра, $Q_{1,k+1} = D_{1,k} = 1$, так как $D_{1,k} = Q_{3,k}$ за счет введения следящей обратной связи. Другие триггеры состояния не изменяют из-за подтверждения тождественного нулевого состояния, $Q_{i,k+1} = D_{i,k} = 0$, где $i = 2, 3$. Следующие импульсы сдвигают логическую единицу вправо (рис. 3.7) до установления единичного кода $\{1, 1, 1\}$ по аналогии с таблицей состояния (см. рис. 3.5) сдвигового регистра (см. рис. 3.4). Сопоставительный анализ показывает тождественность исследуемой и эквивалентной таблиц состояния при появлении трех импульсов.

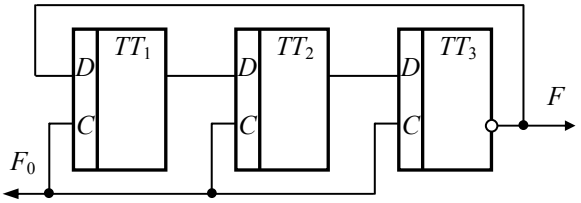


Рис. 3.6 Кольцевой регистр

Четвертый импульс частоты F_0 скопирует в первый триггер TT_1 логический нуль с инверсного выхода последнего триггера TT_3 регистра, $Q_{1,k+1} = D_{1,k} = 0$, так как $D_{1,k} = Q_{3,k} = 0$. За счет следящей обратной связи цикл работы регистра удваивается, и во втором такте единицы замещаются логическими нулями следующими тактовыми импульсами (рис. 3.7). Суммирование нулей в регистре соответствует инверсному оператору вычитания логических единиц, а второй такт заканчивается обнулением всех триггеров TT_i, $i = \overline{1, 3}$, регистра, что соответствует исходному состоянию. Удвоение цикла увеличивает в два раза период работы $T_2 = 2T_1$ и коэффициент деления $N_2 = 2N_1$ при равном числе разрядов кольцевого n_2 и сдвигового n_1 регистров $n_1 = n_2$. Соответственно при равных коэффициентах деления $N_1 = N_2$, тождественны периоды работы $T_1 = T_2$, число разрядов n_1 и код N_1 сдвигового регистра $n_1 = N_1$, в то время как число разрядов n_2 и N_2 кольцевого регистра в два раза меньше, т.е. $n_2 = n_1/2$ и $n_2 = N_2/2$.

F_0	Q_1	Q_2	Q_3
0	0	0	0
1	1	0	0
2	1	1	0
3	1	1	1
4	0	1	1
5	0	0	1

Рис. 3.7 Таблица кольцевого регистра

3.2.3 Программируемый делитель частоты

Делитель частоты на кольцевом регистре без дополнительных связей преобразует тактовую частоту F_0 только на четный коэффициент деления. Управляемый преобразователь с нечетным коэффициентом деления проектируют [13, 24, 29] на кольцевом регистре с дополнительной связью между прямым выходом Q предпоследнего триггера и инверсным входом R последнего разряда регистра (рис. 3.8). Стандартный прием позволяет обнулить регистр на один шаг раньше появления четного состояния логических нулей $\{0, 0, 0\}$. При обнулении тактовым импульсом предпоследнего TT_2 триггера формируется нестабильное состояние $\{0, 0, 1\}$, которое переводится в устойчивое состояние $\{0, 0, 0\}$ нулевым потенциалом $Q_2 = R_3 = 0$ с выхода предпоследнего TT_2 триггера на инверсный статический вход триггера TT_3 старшего разряда регистра.

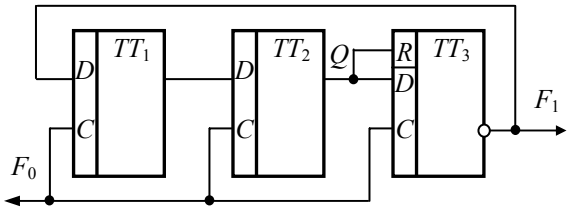


Рис. 3.8 Управляемый делитель частоты

Стандартный прием систематизирует закономерность обнуления регистра при выявлении априори заданной комбинации [24, 29]. Это видно из анализа таблицы состояния $3F(T)$ кольцевого регистра (см. рис. 3.8) на нулевом $F(0)$ и пятом $F(5)$ шагах, представляемых системой уравнений:

$$\begin{cases} F(0) = \overline{Q}_1 \overline{Q}_2 \overline{Q}_3; \\ F(5) = \overline{Q}_1 \overline{Q}_2 Q_3. \end{cases}$$

Для заданной априори комбинации $F(5)$ необходимым условием возврата в исходное состояние $F(0)$ является принудительное обнуление последнего разряда Q_3 регистра. Рациональным решением является мультиплексор $F(\&)$, тождественный аппаратной связи $Q_3 = R_3 = \overline{Q}_2$, реализующей стандартное решение подключения выхода предпоследнего триггера к входу обнуления последнего разряда (см. рис. 3.8).

В табл. 3.1 приведены основные формы $3F$ представления управляемого делителя частоты на кольцевом регистре $N/2$ [24] с коэффициентом деления $N = 5$. Число разрядов n, соответственно триггеров регистра выбирают по формуле:

$$n = [N/2],$$

где квадратные скобки регламентируют округление до целого числа. Для приведенного примера $n = 5/2 = 2{,}5$, поэтому регистр конструируют на трех D-триггерах, т.е. $n = 3$ (см. рис. 3.8). Структурная схема $3F(R)$ отражает топологию управляемого делителя на уровне СИС по правилам $3F(\Phi)$, систематизированным в виде алгоритма программы в таблице состояния $3F(T)$, и их иллюстрацией на семействе временных диаграмм $3F(\varepsilon)$.

В исходном состоянии $F(0)$ триггеры обнулены $\{0, 0, 0\}$ и последовательность импульсов $\overline{1, 3}$ частоты F_0 изменяет в регистре код до единичного значения $F(3) = \{1, 1, 1\}$. При этом логическая единица на выходе второго триггера TT_2 отключает статический вход R реверса последнего триггера TT_3. Вторая последовательность импульсов 4 – 5 уменьшает максимальный код по линейному закону до состояния $F(5) = \{0, 0, 1\}$, которое является неустойчивым. Это обусловлено инициализацией статического входа R_3 нулевым потенциалом второго разряда Q_2, который превалирует над динамикой регистра и переключает последний триггер TT_3 в нулевое состояние $F(\&) = Q_3 = R_3 = 0$. Обнуление регистра соответствует команде «ВОЗВРАТ» из конечного пятого состояния $F(5)$ в начальное нулевое состояние $F(0)$. Цикл продолжается по вышеописанному алгоритму, а на выходе преобразователя выходной период T в N раз больше периода T_0 импульсов тактовой частоты F_0.

Оценка семейства временных диаграмм $3F(\varepsilon)$ доказывает тождественность полученного коэффициента $N^* = T/T_0 = 5$ заданному эквиваленту $N = 5$. Равенство исследуемого и эквивалентного значений $N^* = N$ подтверждает правильность синтеза и анализа компонент СИС в частности и проектирование кольцевого регистра в основных формах науки и техники в целом.

Таким образом, вектор развития гибкости последовательностных СИС направлен от жесткой структуры счетчика с бинарным кодом к управляемой архитектуре кольцевого регистра с программируемым кодом. Методы синтеза комбинаторной логики с итерационным анализом булевых преобразований систематизируются в закономерности определений для синтеза СИС из упорядоченной комбинаторной структуры триггеров ИС, анализируемых по правилам их функционирования. Оценку правильности проектирования организуют по тождественности исследуемого решения эквиваленту семейства временных диаграмм.

3.3 МАТРИЧНАЯ ЛОГИКА

Показано развитие триггеров в матричной логике по закономерностям программирования архитектуры ПЛМ, систематизированным в правила технологии проектирования СИС микропроцессорных средств.

Проектирование СИС в матричной логике не рационально по структурным формулам ИС из-за сложности математических моделей и алгоритмов последовательностных преобразователей. Архитектуру СИС целесообразно моделировать в упорядоченном адресном пространстве по закономерностям булевых преобразований, систематизированных в правила анализа и синтеза образов представления функции в науке и технике [13, 24].

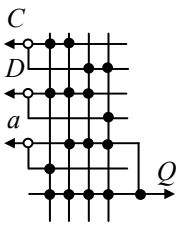
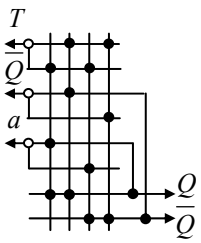
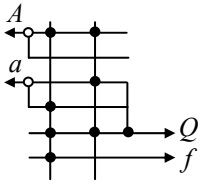
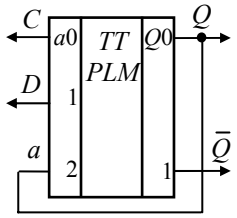
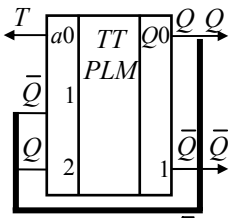
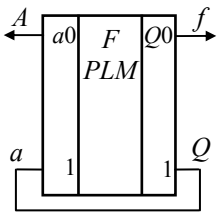
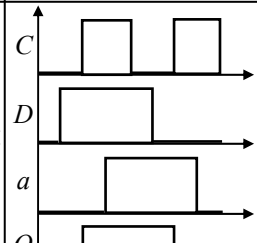
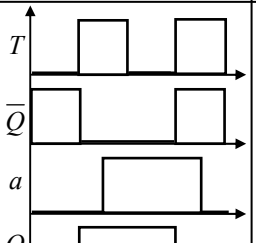
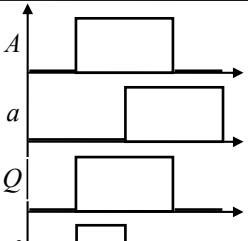
D-триггер в матричной логике (см. табл. 3.2, столбец 1) проектируют по правилам программного управления динамическим режимом [24]. Учитывая, что статика привалирует над динамикой, то для переключения по программе в адресном пространстве необходимо, во-первых, отключить статические *S* и *R*-входы:

$$S = R = 0 \text{ или } 1.$$

Первое правило динамического режима переключения требует равенства потенциалов логическому нулю (или единице) на статических входах динамического триггера для их отключения. Сканирование адресного пространства динамического триггера инициируют импульсы тактовой частоты F_0 по фронту (переключение с нуля на единицу) или срезу (с единицы на нуль), поступающие на синхронизующий *C*-вход.

Второе правило организации динамического режима требует формирования на входе *C* фронта « 0 1 » (или среза « 1 0 ») частоты F_0:

3.2 Матричная логика

	1 *D*-триггер	2 *T*-триггер	3 Модуль
F(*R*)			
F(*T*)	C | D | a | Q_{k+1} 0 0 0 0 0 1 0 0 1 1 0 1 1 1 1 1 0 1 1 1 0 0 1 1 1 0 1 0 1 0 0 0	T | D | a | Q_{k+1} 0 1 0 0 0 1 0 0 1 0 0 1 1 0 1 1 0 0 1 1 0 0 1 1 1 1 1 0 1 1 0 0	A | a | Q | f 0 0 0 0 1 0 1 1 1 1 1 0 0 1 0 0
F(Ф)	 1. $C=F_0=0\sqcup1$ 2. $Q_{k+1}=D_k$	 1. $T=F_0=0\sqcup1$ 2. $D_k=\overline{Q}_k$	 1. $A=A_i=0\sqcup1$ 2. $Q_{k+1}=A_{ki}$
F(ε)			

$C = F_0 = 0\ 1\ (1\ 0).$

Состоянием Q_{k+1} на следующем $(k + 1)$-м шаге управляет по программе логический потенциал входа задержки $D_k = \overline{\{0, 1\}}$ k-го состояния:

$$Q_{k+1} = D_k.$$

Третье правило регламентирует копирование информации в D-триггер тождественно соответствующему по программе состоянию на D-входе. Это обуславливает алгоритм переключения

$$\text{если } D_k = \begin{cases} 0 \\ 1 \end{cases}, \quad \text{то } Q_{k+1} = \begin{cases} 0 \\ 1 \end{cases}.$$

Из алгоритма следует, что D-триггер переключается в противоположное состояние только при инверсии логического потенциала на входе задержки. При неизменном потенциале на D входе состояние триггера остается постоянным, так как в него копируется инвариантный адрес. При фиксированной подстановке подтверждаются тождественные состояния $Q_{k+1} = Q_k = D_k$ замены k-го образа на эквивалентный $(k + 1)$-й образ. Это соответствует программному оператору «СТОП», когда триггер работает, а адрес не изменяется. Третье правило реализуют при программном проектировании на регистрах управляемых таймеров и делителей частоты, генераторов импульсов и линий задержки, микропроцессорных реле защиты и интерфейсов ввода-вывода. Правила функционирования D-триггера в динамическом режиме систематизированы в табл. 3.2, 1F(Ф), здесь же приведена его структурная схема ТТ на основе программируемой логической матрицы PLM.

Архитектуру триггера на программном уровне $F(T)$ поясняет таблица состояния, реализованная по правилам сканирования адресного пространства. Состояние Q_{k+1} определяют два входа C, D – синхронизации и данных информации, а также вход обратной связи a для регистрации адресного состояния. Таблица состояния 1$F(T)$ систематизирует трехадресное пространство $\{C, D, a\}$ из восьми тактов программы. Длительность программы определяется числом логических состояний по входу задержки $D = \{0, 1\}$, принимающих нулевое и единичное значение по фронту 0 1 двух импульсов синхронизации частотой F_0 на входе C. Для реализации команд «СТОП» и «ПЕРЕХОД» удваивают длительность тактовой частоты, формируемой по входу C в виде первого столбца {00110011}. Информацию на входе D формируют из равной последовательности логических единиц и нулей, начиная с исходного нулевого адреса $\{C, D, a\} = \{0, 0, 0\}$ и состояния $Q = \{0\}$ первой строки таблицы 1$F(T)$. Нулевой адрес регламентирует команду «СТОП» при $D = 0$, а первый $\{0, 1, 0\}$ – при $D = 1$ без изменения состояния $Q_1 = Q_2 = 0$. Третья и четвертая строки таблицы 1$F(T)$ формируют последовательно команды «ПЕРЕХОД» с нуля на единицу по фронту импульса {1101} и «СТОП» за счет подтверждения по входу обратной связи a единичного состояния с выхода $Q_3 = Q_4 = 1$ в адресном пространстве $\{C, D, a\} = \{1, 1, 1\}$. Пятая и шестая строки таблицы 1$F(T)$ командой «СТОП» подтверждают состояние логической единицы $Q_5 = Q_6 = 1$ при инверсии входной информации с единицы на нуль по аналогии с первым и вторым адресом. Это соответствует невыполнению второго правила из-за наличия на входе C среза и нулевого потенциала частоты F_0. По седьмому и восьмому адресам {101} и {100} реализуются правила переключения D-триггера с логической единицы $\{a, Q\} = \{0, 1\}$ на нуль $\{0, 0\}$ за счет формирования последовательности команд «ПЕРЕХОД» и «СТОП».

Несложно убедиться, что программируемая по правилам таблица 1*F*(*T*) реализует алгоритм переключения и структурную формулу *D*-триггера. Например, в НДФ структурная формула по прямому выходу имеет вид

$$Q_{k+1} = CD\bar{a} + CDa + \bar{C}Da + \bar{C}\bar{D}a\ ,$$

что после тождественных преобразований и аксиом дизъюнкции соответствует структурной схеме 1*F*(*R*) (табл. 3.2) и формуле 2*F*(Ф) (табл. 2.5) *D*-триггера

$$Q_{k+1} = \left(CD + \bar{C}Q\right)_k\ ,$$

так как $a = Q$ за счет синтеза обратной связи. Структурную схему 1*F*(*R*) в матричной логике и семейство временных диаграмм 1*F*(ε) проектируют методом аналогии по таблице состояния 1*F*(*T*). Следовательно, проектировать *D*-триггер рационально по правилам переключения методами аналогии анализа и синтеза адресного пространства программы управления, тождественной образам науки и техники алгебры Буля.

T-триггер в матричной логике (см. табл. 3.2, столбец 2) регламентирует программное в аппаратное управление функцией переключения за счет изменения третьего правила динамики работы *D*-триггера. Программное управление по *D*-входу заменяют фиксированной обратной связью с инверсным выходом:

$$D_k = \overline{Q_k}\ .$$

Третье правило для *T*-триггера (см. 2*F*(Ф)) ограничивает вдвое адресное пространство за счет жесткой обратной связи, регламентирующей состояния инверсных переключений последовательностью тактовых импульсов F_0 по счетному входу *T*. На структурной схеме 2*F*(Ф) обратная связь показана в виде шины, соединяющей адресным способом входы $\{a_1, a_2\}$ с выходами $\{\bar{Q}, Q\}$. Поэтому *T*-триггер переключается при выполнении только двух первых правил *D*-триггера: статика отключена при $S = R = 0$ и наличие фронта (или среза) импульса частоты F_0 на счетном входе *T*, т.е.

$$\sqcup\!\!\sqcap \qquad\qquad T = F_0 = 0\ 1.$$

Таблицу состояния 2*F*(*T*) проектируют по аналогии с таблицей 1*F*(*T*) *D*-триггера, при чем на *D* входе программируют состояния, тождественные инверсному $\bar{Q}$ выходу. В частном случае таблица 2*F*(*T*) сокращается до двухадресного пространства $\{T, a\}$ из четырех состояний, реализующих структурные формулы счетного триггера 3*F*(Ф), табл. 2.5:

$$\begin{cases} Q_{k+1} = \left(T\bar{Q} + \bar{T}Q\right)_k; \\ \bar{Q}_{k+1} = \left(\bar{T}\bar{Q} + T\bar{Q}\right)_k. \end{cases}$$

Это следует из минимизации 2*F*(*T*) в НДФ:

$$\begin{cases} Q_{k+1} = \left(T\bar{D} + \bar{T}\bar{D}a\right)_k = \left(T\bar{D} + \bar{T}QQ\right)_k; \\ \bar{Q}_{k+1} = \left(\bar{T}D\bar{a} + TD\right)_k = \left(\bar{T}\bar{Q}\bar{Q} + TD\right)_k, \end{cases}$$

так как $\bar{D}_k = Q_k = a_k$.

По методам аналогии из таблицы состояния 2*F*(*T*) проектируют схему *T*-триггера в матричной логике 2*F*(*R*) ПЛМ и семейство временных диаграмм 2*F*(ε) его работы. На диаграммах видно инверсное переключение выходных состояний $\{Q, \bar{Q}\}$ по бинарному коду из нуля в единицу и обратно последовательностью счетных импульсов частоты F_0 по входу *T*. Согласно правилам 2*F*(Ф) переключения *T*-триггера формируются по фронту 0 1 счетного импульса 2*F*(ε). Следовательно, проектирование *T*-триггера по правилам программирования адресного пространства рационально при создании счетчиков, организующих деление частоты по бинарному коду.

Модуль в матричной логике (см. табл. 3.2, столбец 3) дифференцирует комбинаторное представление функции до уровня ассоциативной программы, интегрирующей функции СИС и БИС при проектировании в адресном пространстве по эквивалентам, тождественным модулю [16, 22].

В основу программирования модуля положена закономерность пошагового сканирования адресного пространства за период, систематизированная алгоритмом:

$$\text{если} \quad a_{kj}\begin{Bmatrix}=\\ \neq\end{Bmatrix}Q_{k+1,j}, \quad \text{то} \quad j := \begin{cases} j - \text{СТОП} \\ j+1 - \text{ПЕРЕХОД}\end{cases}.$$

Алгоритм анализирует тождественность векторов состояния следящей обратной связи a_{kj} и модуля $Q_{k+1,j}$ для j-х ($j = \overline{1, n}$) адресов $j_k = \{A, a\}_k$, $j_{k+1} = \{a, Q\}_{k+1}$ в исходном k и установившемся $(k + 1)$ состоянии. При тождественности векторов состояния $a_{kj} = Q_{k+1,j}$ организуется команда «СТОП» за счет подтверждения исходного адреса $j := j$, т.е. адресации на тот же адрес. Если векторы состояния на j-м шаге не равны $a_{kj} \neq Q_{k+1,j}$, то программно формируется команда «ПЕРЕХОД» на следующий шаг $j := j + 1$ устойчивого состояния $a_{k,j+1} = Q_{k+1,j+1}$. Импульс перехода генерируется в момент замещения неустойчивого состояния равновесием, инициируемым командой «СТОП» по заданному уровню логического потенциала A_i на входе A.

Равновесное состояние изменяют логическим потенциалом тактового импульса $A_i = \{0, 1\}$ по входу синхронизации A аналогичным алгоритмом пошаговой адресации

$$\text{если} \quad j_k\begin{Bmatrix}=\\ \neq\end{Bmatrix}j_{k+1}, \quad \text{то} \quad j := \begin{cases} j+1 \\ j\end{cases},$$

где $j_k = \{A, a\}_k$ и $j_{k+1} = \{a, Q\}_{k+1}$ – адреса в исходном k и установившемся $(k + 1)$ состоянии. При тождественности адресов $j_k = j_{k+1}$ тактовый импульс изменяет адрес $j := j + 1$ за счет инверсии логического потенциала на входе A и его копирования на выходе Q. Это приводит к неравенству адресов $j_k \neq j_{k+1}$, подтверждающих исходный адрес $j := j$, а также нетождественности векторов состояния $a_{kj} \neq Q_{k+1,j}$, инициирующих команду «ПЕРЕХОД» $j := j + 1$ на устойчивое состояние $a_{k,j+1} = Q_{k+1,j+1}$. Устойчивость сохраняется до момента инверсии входного потенциала сигнала A_i, и функционирование модуля продолжается по вышеописанным алгоритмам до равновесного состояния с командой «СТОП».

Алгоритмы тождественности адресов и векторов состояния, переключающих модуль по программе, систематизированы в табл. 3.2 правилами 3F(Ф), регламентирующими динамический режим последовательностью импульсов A_i на входе A, т.е. $A = A_i$. Второе правило постулирует копирование уровня потенциала сигнала $A_{ik} = \{0, 1\}$ на выходе модуля в $(k + 1)$ состоянии $Q_{k+1} = A_{ik}$, где $i = \overline{1, m}$ – номер фазы электрической сети. Структурная схема модуля формирователя F сигналов f из последовательности фаз A_i на PLM (см. 3F(Ф)) реализует по программе правила переключения в двухадресном пространстве $\{A, a\} = = \{a_0, a_1\}$ со следящей обратной связью $\{a, Q\} = \{a_1, Q_1\}$. На выходе Q_0 запрограммирован сигнал f заданной длительностью, инициируемый фазой A_i по входу A с адресом $a_0 = A$.

Архитектуру модуля поясняет программа, представленная таблицей состояния 3F(T) за период T фазного сигнала A_i. Исходное состояние иллюстрирует первая строка $j = 1$ с нулевым потенциалом A, нулевым состояниями векторов $a_{k,1} = Q_{k+1,1} = 0$ обратной связи и сигнала $f = 0$. Устойчивое состояние векторов $a_{k,1} = Q_{k+1,1}$ регламентируется командой «СТОП», подтверждением исходного адреса $j_k = j_{k+1} = \{0, 0\}$ до момента инверсии потенциала фазы A_i. При появлении на входе A потенциала логической единицы импульсом A_i изменяется адрес $\{A, a\} = \{0, 0\} = j$ на $\{1, 0\} = j + 1$, что соответствует переходу на вторую строку $j = 2$ таблицы состояния. По программе единица копируется на выходе f и Q, но инициируется нестабильное состояние векторов $a_{k,2} = 0 \neq 1 = Q_{k+1,2}$. По следящей обратной связи $\{Q_1, a_1\}$ дестабилизация командой «ПЕРЕХОД» заменяется устойчивым состоянием векторов $a_{k,3} = Q_{k+1,3} = 1$ с адресом $\{1, 1\} = j + 1$ третьей строки $j = 3$. Команда «СТОП» подтверждает стабильное состояние тождественными значениями логической единицы до изменения потенциала фазного сигнала A_i. При инверсии фазы на входе A третий адрес $\{A, a\} = \{1, 1\}$ изменяется на четвертый $\{0, 1\} = j + 1$, а вектор $Q_{k+1,3} = 1$ изменяет состояние $Q_{k+1,4} = A_{ki} = 0$ на логический нуль. Неравенство исходного $j_k = \{0, 1\}_k$ и сформированного $j_{k+1} = \{1, 0\}_{k+1}$ адресов по команде «ПЕРЕХОД» переключает модуль на первый адрес $j_k = \{0, 0\}$. Заканчивается программируемый период и начинается следующий по правилам тождественности адресов и векторов состояния.

Схема модуля в матричной логике 3F(R) и семейство временных диаграмм 3F(ε) проектируют методами аналогии по эквиваленту таблицы состояния 3F(T). Сигнал f на выходе модуля контролирует наличие фазы A_i и индицирует нормальное состояние функционирования. Для многофазной сети по мо-

дулю-эквиваленту $3F(T)$ двухадресное пространство расширяют в m раз тиражированием эквивалента в адресном пространстве таблицы состояния согласно числу фаз. Пример микропроцессорной защиты трехфазной электрической сети, приведенный в книге [16], спроектирован для контроля нормального функционирования программно управляемого инвертора энергии компьютерного электропривода.

Таким образом, показано развитие универсальных триггеров в матричной логике по упорядоченности структурных формул и алгоритмов функционирования в правила организации адресного пространства по ассоциациям модулей открытого типа. Комбинаторная структура последовательностных преобразователей ИС, создаваемая методами алгебры Буля, совершенствуется в ассоциативную архитектуру СИС матричной логики ПЛМ, программируемую методами аналогии математики образов по эквивалентам, конструируемых из ассоциации программных модулей. Методы аналогии по эквивалентам архитектуры систематизируются в правила программирования анализа тождественности адресов и векторов состояния ассоциативного адресного пространства для организации информационной технологии проектирования СИС, БИС и микропроцессорных средств.

3.3.1 Счетчики

Аппаратно управляемый цифровой преобразователь для счета импульсов называется счетчиком согласно информационным концепции и технологии. В матричной логике с топологической избыточностью ПЛМ гибкой структуре сопоставляют для определенности структурную схему из комбинаторной логики, а алгоритм счета импульсов представляют программным и математическим обеспечением в адресном пространстве архитектуры.

Счетчики в матричной логике синтезируют [16, 24] на программируемых модулях с открытой архитектурой (см. табл. 3.2, столбец 3), что на порядок экономичнее конструирования его по правилам комбинаторной логики на конкретных триггерах. При этом способ переключения организуют не аппаратными, а программными средствами. По этой причине алгоритм функционирования архитектуры поясняют программой (блок-схемой, таблицей состояния или переключения), а структуре ассоциативной матрицы ставят в соответствие жесткую схему на элементах комбинаторики, не существующих в топологии ПЛМ или организованных в неявном виде.

Адресное пространство

Счетчик в матричной логике конструируют не из линейки триггеров комбинаторики, а по программе таблицы состояния, систематизирующей адресное пространство по правилам функционирования заданного оператором [24]. Упорядочивают адресное пространство многомерной системой координат векторного счисления. На рис. 3.9 приведено трехадресное пространство на примере декартовой системы координат из трех единичных векторов a_0, a_1, a_2 перпендикулярных друг другу. В трехмерной системе координат адрес точки $\{a_0, a_1, a_2\}$, определяют состояния вектора $a_i = \{0, 1\}$ в унитарном логическом пространстве нулей и единиц. Точка пересечения векторов соответствует нулевому адресу $\{0, 0, 0\}$, а по осям единичных векторов точки адресуют вершины векторов $a_0 - \{1, 0, 0\}$, $a_1 - \{0, 1, 0\}$, $a_2 - \{0, 0, 1\}$ последовательностью чисел 1, 2, 4 в двоичной системе счисления.

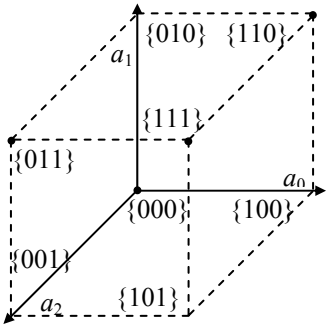


Рис. 3.9 Трехадресное пространство $\{a_0, a_1, a_2\}$

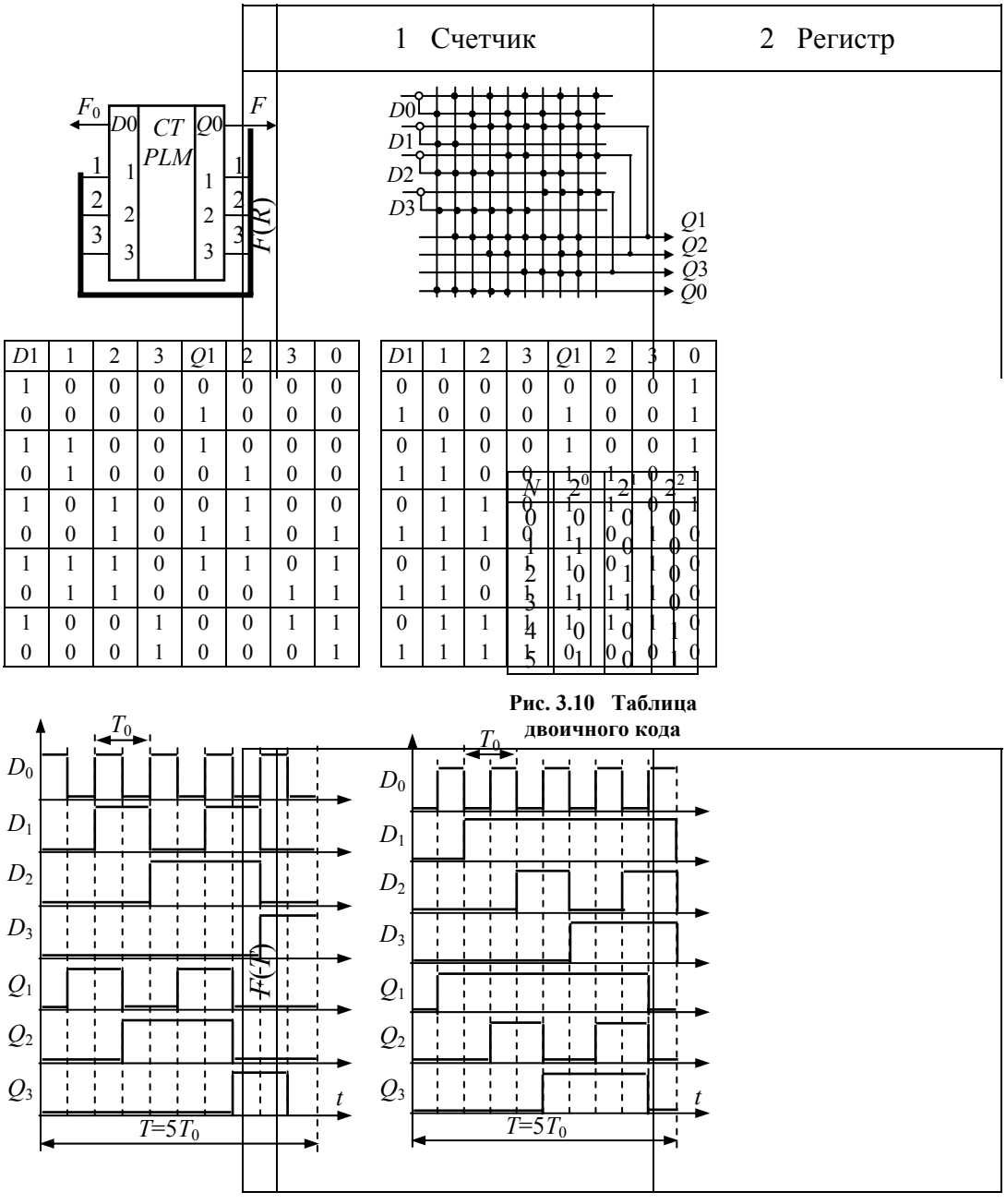
Программа отражает динамику переключения структуры СИС за один период сканирования адресного пространства, систематизируемой последовательностью подстановок адресов в таблице состояния. Правила адресации и сканирования определяет разработчик согласно удобной системе счисления и рациональной технике адресации. Наиболее наглядным является сканирование адресов по линейному закону суммирования с нулевого $\{0, 0, 0\}$ значения до максимального N, для определенности выбранного пяти $N = 5$, с шагом сканирования $\Delta N = 1$ равным единице. Период программы отражает физический цикл функционирования структуры и регламентирует начальный и конечный адреса сканирования. В приведенном примере выбираем начальный (исходный) адрес нулевой $\{0, 0, 0\}$, конечный адрес равный четырем $\{0, 0, 1\}$ для формирования пяти шагов соответственно коду $N = 5$. Адресное пространство счетчика программируют по эквиваленту таблицы двоичного кода (рис. 3.10).

Таблицу бинарного счисления синтезируем стандартным образом, начиная с нулей и заканчивая максимальным числом $N = 5$. В младшем разряде формируем последовательность из нулей и единиц с разрядкой один раз и удваиваем их разрядку согласно основанию двоичного кода (рис. 3.10). Проекти-

рование на счетчике таймера с коэффициентом деления $N = 5$, сдвигом $\Delta N = 1$ по срезу 1 0 импульса начинают с исходного устойчивого состояния по команде «СТОП» при единичном потенциале на счетном входе (см. табл. 3.3, $1F(T)$). В системе координат (рис. 3.9) нулевому значению первой строки таблицы кодов (рис. 3.10) соответствует адрес $\{a_0, a_1, a_2\} = \{0, 0, 0\}$ пересечения векторов. Команду «СТОП» для нулевого значения программируют подстановкой на тот же адрес {0, 0, 0} по выходам $\{Q_1, Q_2, Q_3\}$ обратной связи ПЛМ. Поэтому первая строка таблицы состояния $1F(T)$ состоит из нулей, подтверждающих начальный адрес при единичном потенциале счетного импульса F_0 на входе $D_0 = 1$. Устойчивое состояние по адресу {0, 0, 0} сохраняется до появления среза импульса F_0.

Вторая строка таблицы состояния $1F(T)$ инициирует команду «ПЕРЕХОД» с нулевого на первый {1, 0, 0} адрес при формировании на счетном входе D_0 частоты F_0. Подстановка моделируется из таблицы двоичного кода (рис. 3.10) увеличением исходного значения N_0 на коэффициент сдвига $\Delta N = 1$. Это соответствует следующему адресу перехода $N_1 = N_0 + \Delta N$ второй строки таблицы {1, 0, 0}, а в трехадресном пространстве (см. рис. 3.9) переходу из нулевого адреса

3.3 Таймеры

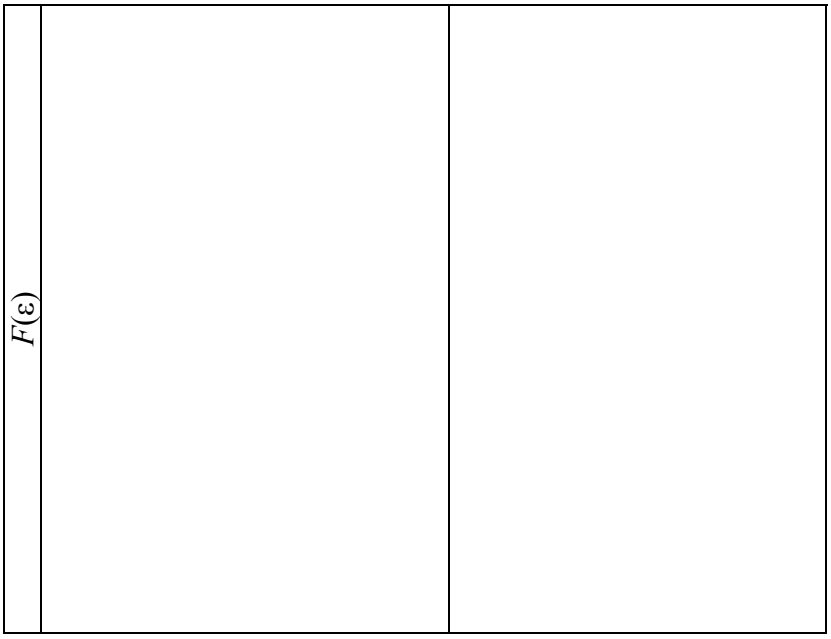


D1	1	2	3	Q1	2	3	0
1	0	0	0	0	0	0	0
0	0	0	0	1	0	0	0
1	1	0	0	1	0	0	0
0	1	0	0	0	1	0	0
1	0	1	0	0	1	0	0
0	0	1	0	1	1	0	1
1	1	1	0	1	1	0	1
0	1	1	0	0	0	1	1
1	0	0	1	0	0	1	1
0	0	0	1	0	0	0	1

D1	1	2	3	Q1	2	3	0
0	0	0	0	0	0	0	1
1	0	0	0	1	0	0	1
0	1	0	0	1	0	0	1
1	1	0	0	0	1	0	1
0	1	1	0	0	1	0	1
1	1	1	0	1	1	0	1
0	1	0	1	1	0	1	0
1	1	0	1	1	0	1	0
0	1	1	1	1	0	1	0
1	1	1	1	0	0	1	0

N	2^0	2^1	2^2
0	0	0	0
1	1	0	0
2	0	1	0
3	1	1	0
4	0	0	1
5	1	0	1

Рис. 3.10 Таблица двоичного кода



{0, 0, 0} вправо (рис. 3.11) по адресом {1, 0, 0}. Поэтому переход из нулевого адреса по адрес состояния по выходам {Q_1, по следящей обратной связи логической единицы на входе

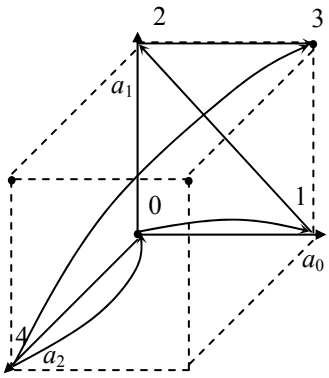


Рис. 3.11 Мнемосхема цикла переходов

Третья строка таблицы 1$F(T)$ твержднием первого адреса Таблица состояния 1$F(T)$ рис. 3.10) и контролируется адресного пространства. таблице состояния по

вектору a_0 в единичное состояние с вторая строка таблицы 1$F(T)$ указывает входам {D_1, D_2, D_3} = = {0, 0, 0} на первый Q_2, Q_3} = {1, 0, 0}. Неустойчивое состояние переходит в равновесное при появлении $D_0 = 1$ тактовой частоты.

иллюстрирует команду «СТОП» под- {1, 0, 0} по входу и выходу программы. синтезируется согласно таблице кода (см. визуально по мнемосхеме (см. рис. 3.9) Команды «СТОП» программируют в единичному потенциалу младшего адреса $D_0 = 1$, а по нулевому состоянию $D_0 = 0$ тактового импульса частоты F_0 моделируют подстановку «ПЕРЕХОД» с j-го на (j + 1)-й адрес. На рис. 3.11 показана последовательность переходов за один цикл программы в адресном пространстве таймера на счетчике с коэффициентом деления $N = 5$ и сдвигом $\Delta N = 1$. Следует отметить закольцевание программы переходом с последнего четвертого адреса {0, 0, 1} на исходный нулевой {0, 0, 0}. Последнюю подстановку за период программы иллюстрирует нижняя строка таблицы состояния 1$F(T)$, инициирующая возврат программы на исходное состояние с нулевым адресом первой строки. При отсутствии возврата с конечного на начальный адрес программа выходит из-под контроля и проектируемое устройство становится неуправляемым. Поэтому при составлении программы необходимо позаботиться о ее завершении организацией команды «ВОЗВРАТ» с конечного на исходный адрес для завершения цикла операции.

Программирование по эквивалентам

Метод программирования по эквивалентам таймера на счетчике включает:

1. Задание коэффициента деления N и сдвига ΔN шага программирования.
2. Синтез таблицы двоичного кода, при необходимости иллюстрации – мнемосхемы адресного пространства и структурной схемы.
3. Выбор техники адресации (прямая, косвенная, относительная…) и правил программирования (последовательная, произвольная, ассоциативная выборка).
4. Задание числа тактовых импульсов и динамики переключения (по фронту или срезу).
5. Синтез эквивалентов программирования подстановок «ПЕРЕХОД» и «ВОЗВРАТ», команды «СТОП».
6. Программирование эквивалентов в программу таблицы состояния по заданию и выбранным правилам.
7. Проектирование структурной и матричной схем на ПЛМ методами аналогии по таблице состояния.

8. Синтез по аналогии семейства временных диаграмм и анализ коэффициента деления на тождественность заданному значению, принятому за эквивалент.

9. Утверждение правильности проектирования таймера при тождественности эквиваленту значения коэффициента деления из семейства временных диаграмм.

Последние три пункта метода программирования по эквивалентам аналогичны проектированию комбинационных и последовательностных схем малой и средней степени интеграции. Анализ коэффициента N^* из семейства временных диаграмм $1F(\varepsilon)$ организуют при сравнении периодов T_0 следования импульсов тактовой частоты F_0 на входе D_0 и цикличности периода T созданной программы на выходах Q. Период T следования программы определяют по одинаковым фазам выходных состояний и числу в нем тактовых периодов $T = N^* T_0$. Из отношения периодов T/T_0 находят исследуемый коэффициент N^*, который сравнивают с заданным значением N. При тождественности полученного значения эквиваленту $N = N^*$ считают, что таймер на счетчике спроектирован верно, в противном случае корректируют программу в таблице состояния.

Таким образом, метод программирования по эквивалентам позволяет проектировать таймер на счетчике в матричной логике ассоциативного типа без использования комбинаторики триггеров. Ассоциативная архитектура систематизирует в программу модули-эквиваленты, организованные из последовательности команд «ПЕРЕХОД» на следующий шаг программы по заданному адресу и «СТОП» при подтверждении исходного состояния. Эквиваленты синтезируют в виде программируемых модулей открытого типа по правилам переключения триггеров в динамическом режиме, без проектирования структуры последних. В отличие от ИС комбинаторной и релейной логики, создаваемых итерационными методами по структурным формулам, СИС в матричной логике, программируемые по эквивалентам более эффективны по метрологическим, технологическим и экономическим показателям.

3.3.2 Регистры

Регистром определим программно управляемый преобразователь цифрового сигнала для регистрации импульсов. Определение по информационной концепции существенно отличается от классического для комбинаторики ИС. Это обусловлено синтезом регистров в адресном пространстве архитектуры на программируемых модулях [16, 24], а не на конкретных триггерах.

Регистры оперируют на множестве кодов в отличие от счетчиков, регламентированных двоичным кодом, по этой причине счетчики – аппаратно управляемые цифровые преобразователи, а регистры – программно управляемые последовательностные СИС. По гибкости программного управления и универсальности математического обеспечения регистры значительно превосходят счетчики, которые отличаются лишь большим объемом памяти. За счет программного управления архитектура регистров положена в основу БИС: микропроцессора, интерфейсов памяти и ввода-вывода, на которых конструируют микропроцессорные средства.

Таблица кодов

Регистр в матричной логике проектируют по аналогии со счетчиком методом программирования по эквивалентам [16], тиражированием таблицы кодов в ассоциативную программу таблицы состояния. Приведем синтез и анализ регистра на примере проектирования программируемого таймера с коэффициентом деления $N = 5$, сдвигом $\Delta N = 1$ и первым кодом Фибоначчи. Для создания эквивалента программы синтезируем ряд оснований $a(i)$ кода Фибоначчи

$$N_p = \sum_{i=0}^{n-1} \xi_i a(i),$$

$$a(i + 1) = a(i) + a(i - p).$$

Для первого кода Фибоначчи $p = 1$ основания синтезируют по алгоритму

$$a(i + 1) = a(i) + a(i - 1).$$

Следующее $(i + 1)$-е основание получают суммированием предыдущих двух оснований на позициях i и $i - 1$. Ряд формируют из единицы и множества нулей на младших позициях {0, 0, …, 0, 1}. Принимая $a(i) = 1$ и $a(i - 1) = 0$, находим по алгоритму значение второго основания $a(i + 1) = 1 + 0 = 1$. Для следующего третьего $a(i + 1)$-го основания два предыдущих равны $a(i) = a(i - 1) = 1$, тогда $a(i + 1) = 1 + 1 = 2$. Четвертое основание вычисляют сложением второго и третьего оснований: $a(i + 1) = 1 + 2 = 3$ и т.д. Ряд оснований первого кода Фибоначчи имеет вид

Σ \ a	min	max
→	+	
←		

Рис. 3.12 Таблица правил

1, 1, 2, 3, 5, 8, …

Выбираем число оснований по коэффициенту N деления из неравенства $N \le N_p$, т.е. сумма оснований должна быть не меньше кода N. Этому условию удовлетворяет сумма из первых четырех оснований {1, 1, 2, 3}, так как $N_p = 1 + 1 + 2 + 3 = 7 \ge 5 = N$. Синтезируем таблицу кода чисел Фибоначчи для веса ξ_i = {0, 1} по правилу сложения минимальных оснований ряда слева направо. Из-за множественности представлений чисел, для их однозначности, выбирают одно из четырех правил исчисления кодов Фибоначчи, систематизированных в таблице на рис. 3.12. Правила регламентируют однозначность чисел при фиксации направления суммирования Σ и исходного основания a: слева направо «→» и справа налево «←», по минимальному «min» и максимальному «max» основанию. В таблице на рис. 3.12 отмечено знаком «+» выбранное правило, по которому спроектирована таблица кодов (рис. 3.13).

Из таблицы кодов видно, что для организации коэффициента деления $N = 5$ достаточно пять чисел из этих разрядов с основаниями {1, 1, 2} или {1, 2, 3} от нуля {0, 0, 0} до четырех {1, 1, 1} или пяти {1, 0, 1}. Для определенности выбираем первый вариант, для которого синтезируем структурную схему таймера на регистре в матричной логике *PLM* (рис. 3.14) с тремя обратными связями $\overline{1, 3}$ с выходов $\{Q_1, Q_2, Q_3\}$ на входы $\{D_1, D_2, D_3\}$. Младшие разряды D_0 и Q_0 служат для синхронизации тактовой частотой F_0 и генерации импульсов частоты F с периодом следования, кратным коэффициенту N деления $T = NT_0$. Дальнейшее проектирование таймера аналогично синтезу и анализу счетчика методом программирования по эквивалентам.

Для наглядности программирования проиллюстрируем в адресном пространстве $\{a_0, a_1, a_2\}$ мнемосхему цикла переходов (рис. 3.15), включающую последовательность из пяти адресов с нулевого {0, 0, 0}

N \ a	1	1	2	3
0	0	0	0	0
1	1	0	0	0
2	1	1	0	0
3	1	0	1	0
4	1	1	1	0
5	1	1	0	1

Рис. 3.13 Таблица кодов N_p, $p = 1$

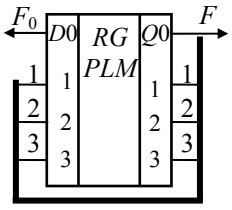


Рис. 3.14 Схема регистра

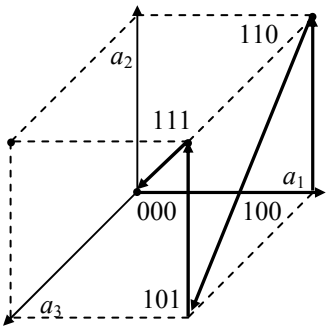


Рис. 3.15 Мнемосхема переходов

по четвертый {1, 1, 1} согласно таблице кодов (см. рис. 3.13). Команду «СТОП» подтверждения исходного адреса формируем по нулевому потенциалу тактовой частоты F_0, а подстановку «ПЕРЕХОД» организуем при появлении импульса синхронизации по единичному уровню. Начало периода определим с нулевого адреса, закончим цикл командой «ВОЗВРАТ» в нулевое состояние {0, 0, 0} из последнего в периоде четвертого адреса {1, 1, 1} (рис. 3.15).

Таблица состояния

Синтезируем таблицу $2F(T)$ таймера в матричной логике на регистре с коэффициентом деления $N = 5$, сдвигом $\Delta N = 1$ методом программирования по эквивалентам [16, 24] последовательности чисел первого кода Фибоначчи (см. табл. 3.3). В первой строке сформируем команду «СТОП» при нулевом потенциале $D_0 = 0$ тактовой частоты F_0 по нулевому адресу на входах $\{D_1, D_2, D_3\} = \{0, 0, 0\}$ и его подтверждение на выходах $\{Q_1, Q_2, Q_3\}$ вектором состояния $\{0, 0, 0\}$. Устойчивое состояние сохраняется до появления импульса частоты F_0 на входе $D_0 = 1$, организующего подстановку «ПЕРЕХОД» с нулевого на первый адрес, инициируемый вектором состояния $\{1, 0, 0\}$ на второй строке таблицы $2F(T)$. Третья и другие нечетные строки таблицы $2F(T)$ тиражируют при отсутствии импульса $D_0 = 0$ команду «СТОП» подтверждением на выходах Q вектора состояния тождественного адреса. Четные строки таблицы $2F(T)$ генерируют подстановку перехода с j-го на $(j + 1)$-й шаг программы со сдвигом $\Delta N = 1$. Программу и ее период заключает последняя строка, моделирующая команду «ВОЗВРАТ» из четвертого адреса $\{1, 1, 1\}$ в исходное нулевое состояние с адресом $\{0, 0, 0\}$.

По таблице состояния $2F(T)$ методами аналогии по информационной технологии синтезируют схему в матричной логике ПЛМ таймера на регистре $2F(R)$ и его семейство временных диаграмм $2F(\varepsilon)$. Анализ решений осуществляют при оценке полученного коэффициента деления N^* на выходе временных диаграмм или адресных входах. Из диаграммы по входу D_3 видно, что период T программы кратен пяти тактам T_0 частоты синхронизации F_0, т.е. $T = 5T_0$ или $N^* = 5$. Это соответствует заданному коэффициенту деления $N = N^* = 5$, откуда следует правильность синтеза структур в частности и проектирования таймера на регистре в целом. Регистр можно проектировать в любой системе счисления, включая коды Фибоначчи ($p = \overline{0, \infty}$) и Грея, непозиционные и мнемокоды, четные и нечетные коды и т.д. Множественность кодов инициирует создание программно управляемого СИС-регистра, являющегося развитием аппаратно управляемого счетчика, реализующего только бинарный код.

Таким образом, проектирование регистра аналогично счетчику в матричной логике методом программирования по эквивалентам, но кроме двоичного кода моделируют числа в различных системах счисления. Множественность кодов развивает аппаратно управляемую структуру счетчика с бинарным счислением в программно управляемую структуру регистра. Метод программирования по эквивалентам развивает архитектуру в ассоциативное адресное пространство, матрице которого ставят в соответствие структурную схему комбинаторики и программу таблицы состояния.

3.3.3 Генераторы

Программно управляемые генераторы импульсов конструируют на базе счетчиков и регистров, делителей частоты и таймеров времени. Генераторы служат для создания преобразователей сигнала и энергии при организации диалоговых, сервисных и автоматических интерфейсов ввода-вывода микропроцессорных приборов и защиты, компьютерных анализаторов и электропривода. Программу генераторов определяет реализуемая функция измерения и контроля для АЦП и ЦАП, управления и регулирования для выпрямителей и инверторов энергии. Информационную технологию проектирования функциональных преобразователей приведем на примере синтеза и анализа знакогенераторов управления частотой инверторов (ЗГИ) энергии и аналого-цифрового преобразователя (АЦП) сигнала.

Знакогенератор инвертора

Знакогенератор инвертора (ЗГИ) формирует по программе циклическую последовательность импульсов для инвертирования постоянного тока в многофазный переменный по различным законам коммутации. Последовательный, параллельный и смешанный законы позволяют программно адаптировать параметры компьютерного электропривода по моменту, мощности и скорости вращения на валу двигателя. Для определенности примера приведем проектирование знакогенератора инвертора энергии постоянного тока в трехфазный переменный по последовательному закону коммутации. Программа таблицы состояния ЗГИ синтезируется методом эквивалентов по таблице коммутации управляемого инвертора, адекватной физике преобразования энергии в электрической сети [16].

Сущность метода эквивалентов заключается в отождествлении исследуемого решения эквиваленту для нахождения условий их равенства. За эквивалент таблицы состояния ЗГИ примем таблицу коммутации инвертора по последовательному закону включения тиристоров [17], приведенную на рис. 3.16. Последовательную коммутацию организуют программированием импульсов со скважностью три по прямым $\{a, b, c\}$ и инверсным $\{\bar{a}, \bar{b}, \bar{c}\}$ амплитудам напряжения фаз A, B, C в виде кода столбца $a = \{110000\}$. Прямые фазы формируют структурным сдвигом кода a на две позиции (на 120° за период

T), что соответствует столбцам b = {001100} и c = {000011} (рис. 3.16). Аналогично синтезируют инверсные значения { $\bar{a}$, $\bar{b}$, $\bar{c}$ }, начиная программирование столбца $\bar{a}$ с середины второй части таблицы, что соответствует сдвигу по фазе на 180° (половина периода) инверсной амплитуды, т.е. $\bar{a}$ = {000110}, $\bar{b}$ = {100001}, $\bar{c}$ = {011000}. Как видно из таблицы инвертора (рис. 3.16), период коммутации трехфазной сети включает циклическую последовательность из шести тактов длительностью 60°, что может быть организовано программой из шести подстановок.

Программу знакогенератора инвертора целесообразно синтезировать по таблице коммутации (рис. 3.16) методом программирования эквивалентов. Исходным образом может служить коммутация прямых значений фаз {a, b, c} первой половины таблицы инвертора. Синхронизацию тактов программы таблицы состояния сформируем по дополнительному младшему разряду a_0, а следящую обратную связь организуем в трехадресном пространстве j_k = = {a, b, c} = {a_1, a_2, a_3} с вектором состояния j_{k+1} = {Q_1, Q_2, Q_3}, которые сведем в таблицу на рис. 3.17. Команду «СТОП» создадим по нулевому потенциалу тактовой частоты F_0 подтверждением исходного j_k адреса значением j_{k+1} вектора состояния. Подстановку «ПЕРЕХОД» генерируем по единичному потенциалу последовательности импульсов F_0 с указанием по j_k-му адресу $(j + 1)_k$-го вектора устойчивого состояния с командой «СТОП».

F_0	a_1	a_2	a_3	Q_1	Q_2	Q_3
0	1	0	0			
1	1	0	0			
0	0	1	0			
1	0	1	0			
0	0	0	1			
1	0	0	1			

Рис. 3.17 Исходная таблица

По введенным правилам и таблицу 1F(T) (табл. 3.4), копируя в 0} в (k + 1)-м состоянии j_{k+1} = {1, 0, Во второй строке таблицы 1F(T) по j_k = {1, 0, 0} создана вектором «ПЕРЕХОД» на следующий адрес (см. рис. 3.17). Третья и пятая устойчивые состояния командами 0, 1}, а на четвертой и шестой запрограммированы подстановки

a	b	c	$\bar{a}$	$\bar{b}$	$\bar{c}$
1	0	0	0	1	0
1	0	0	0	0	1
0	1	0	0	0	1
0	1	0	1	0	0
0	0	1	1	0	0
0	0	1	0	1	0

Рис. 3.16 Таблица инвертора

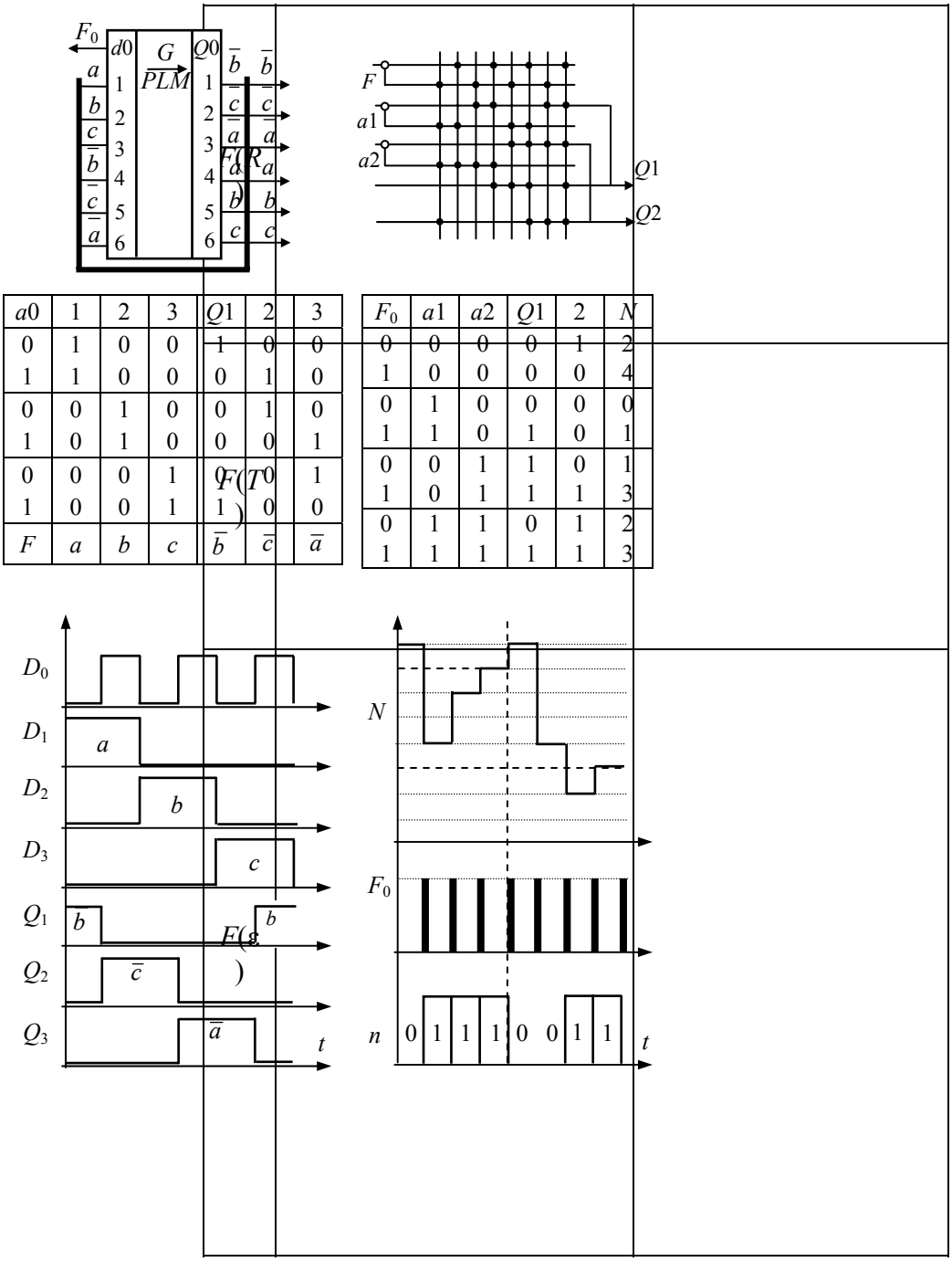
эквивалентам синтезируем исследуемую первой строке исходный адрес j_k = {1, 0, 0} для формирования команды «СТОП». импульсу F_0 на входе a_0 = 1 с адресом состояния {Q_1, Q_2, Q_3} команда j_{k+1} = {0, 1, 0} по эквивалентной таблице строки таблицы 1F(T) формируют тождественной адресации {0, 1, 0} и {0, строках таблицы 1F(T) переходов на следующие шаги {0, 0, 1} и {1, 0, 0}. Кроме того, последняя подстановка инициирует возврат из последнего адреса {0, 0, 1} в исходное состояние первого адреса {1, 0, 0}.

Сопоставим исследуемую таблицу состояния 1F(T) с эквивалентом таблицы инвертора (см. рис. 3.16) для выявления условий тождественности образов. Из анализа следует идентичность левых таблиц с адресами {a_1, a_2, a_3} = {a, b, c} прямых значений фаз за счет копирования коммутации прямых значений фаз. Очевидна также тождественность столбцов левых половин таблиц при условии переименования позиций вектора состояния {Q_1, Q_2, Q_3} в соответствующие инверсные значения фаз { $\bar{a}$, $\bar{b}$, $\bar{c}$ }, т.е. когда {Q_1, Q_2, Q_3} = { $\bar{b}$, $\bar{c}$, $\bar{a}$ }. Следовательно, при условии тождественности состояний эквивалентны синтезированная 1F(T) и образцовая (см. рис. 3.16) таблицы, что подтверждает правильность проектирования таблицы состояния ЗГИ методом программирования эквивалентов.

Семейство временных диаграмм 1F(ε) проектируют из таблицы 1F(T) методами аналогии. Диаграммы по входам a_1, a_2, a_3 иллюстрируют сдвиг по фазе на 120° коммутирующих сигналов a, b, c и их инверсий $\bar{a}$, $\bar{b}$, $\bar{c}$ на выходах Q_1, Q_2, Q_3, отстающих от прямых сигналов на 180°. Синхронно прямым и инверсным сигналам управления на нагрузке

3.4 Знакогенераторы

	1 Инвертор	2 АЦП



инвертора коммутируется трехфазное напряжение переменного тока синусоидальной формы с регулируемой частотой F_0 по заданному закону адаптации компьютерного электропривода [16]. По методам эквивалентных образов синтезирована структурная схема $1F(R)$ знакогенератора инвертора с тиражированием трехадресного пространства таблицы $1F(T)$ в шестиадресное. Удвоение состояний и адресов повышает надежность генерации знаковых импульсов за счет увеличения их скважности до двух, а также электрической развязки состояний импульсов коммутации от адресных импульсов синхронизации.

Таким образом, показано проектирование знакогенератора инвертора в матричной логике методами программирования эквивалентов по условиям тождественности таблиц вектора состояния генератора и коммутации энергии инвертора.

Знакогенератор АЦП

АЦП поразрядного уравновешивания относятся к преобразователям параллельно-последовательного действия, их отличает высокая коммуникабельность и оперативность амплитудно-дискретной обработки информации. Достоинства этих АЦП обусловлены упорядоченной матричной структурой с микропрограммным управлением по гибким алгоритмам информативного математического и универсального программного обеспечения. Создание открытой архитектуры совершенствует способы поразрядного уравновешивания до саморазвивающихся информационных технологий с высокоэффективными метрологическими средствами.

Сущность способов поразрядного уравновешивания [22, 30, 37, 67] заключается в непосредственном представлении амплитуды в код со взвешенными основаниями числоимпульсной последовательности. За период формирования последовательности количество знакомест импульсов организуют соответственно числу позиций оператора счисления, включающего оценку по операторам исчисления уровня исследуемого сигнала с интегралом эквивалентных мер для выявления значимости знакоместа. При положительной оценке формируют на адресе знакоместа импульс в виде потенциала высокого уровня, принимаемого за логическую единицу, в противном случае на адресуемом интервале инициируют потенциалом низкого уровня логический нуль [22].

Знакогенератор ЗГ формирует веса ξ_j оснований α_j кода $N_{0i} = \sum_{j=0}^{i-1} \xi_j \alpha_j$ по операторам исчисления алгебры, арифметики и логики. Например, в бинарном коде НДФ знакогенератор ЗГ синтезирует цифровой эквивалент по итерационному алгоритму

$$\Delta N_j = \Delta N_{j+1} - \xi_{j+1}\, 2^{j+1}$$

при выполнении условия:

$$\text{если} \quad \Delta N_j \begin{Bmatrix} \geq \\ < \end{Bmatrix} \xi_j\, 2^j, \quad \text{то} \quad \xi_j = \begin{cases} 1 \\ 0 \end{cases}.$$

Последовательная итерация разностных алгоритмов приводит к интегральной оценке позиционных кодов исследуемой N_i и нормируемой N_{0i} величин в НДФ:

$$\text{если} \quad N_i \begin{Bmatrix} \geq \\ < \end{Bmatrix} N_{0i}, \quad \text{то} \quad \xi_j = \begin{cases} 1 \\ 0 \end{cases},$$

где $N_{0i} = \sum_{j=0}^{i-1} \xi_j 2^j$, что очевидно из анализа нулевой разницы ΔN_0 алгоритма для i-итераций ($j = \overline{0, i-1}$) при условии $\Delta N_j \geq N_j$.

Следует особенно выделить гибкую архитектуру этих АЦП, которая обусловлена матричной структурой ЗГ с микропрограммным управлением его связями по универсальным алгоритмам исчисления.

Таблица состояния (см. табл. 3.4, $2F(T)$) иллюстрирует один из адаптивных алгоритмов исчисления в двоичном коде счисления для трех разрядов, позволяющих наглядно оценить сущность метода программирования по аналогии. Таблица соответствует ПЛМ с тремя адресными a_k входами, $k = \overline{0, 2}$ и k-ми выходами $Q = \{Q_k\}_0^2$. Нулевой вход a_0 является синхронизирующим и информационным для считывания логических состояний при сравнении исследуемой амплитуды U_i с интегралом аналоговых мер U_{0i}.

Адресные входы $\{a_1, a_2\}$ соединены поразрядно с информационными выходами $\{Q_1, Q_2\}$ ПЛМ и организуют за счет следящей обратной связи программируемый генератор. Входы a_k определяют в адресном пространстве памяти ПЛМ знакоместо подпрограммы алгоритма исчисления по заданным оператором правилам техники адресации (непосредственной или прямой, косвенной или относительной (см. табл. 4.4)). Приведенный пример реализует относительную адресацию, при которой адрес памяти операнда равен сумме адресов a_k, причем базис адреса формирует операнд с шины $\{Q_1, Q_2\}$ по условиям преобразования амплитуд во время на синхронизирующем входе a_0 и алгоритма смещения, реализуемого операндами адресации входов $\{a_1, a_2\}$.

Проектирование программного обеспечения знакогенератора осуществляют методом эквивалентов. В приведенной таблице (см. табл. 3.4, $2F(T)$) не показан выход Q_0, соответствующий знакоместу старшего разряда с основанием 2^2, так как значащая логическая единица запрограммирована во второй строке (число N_2 (4) = 001), а все другие состояния столбца Q_0 обнулены. Выходы Q_k и входы a_k для $k = 1, 2$ отражают на k-х позициях j-е состояния по основаниям 2^{k-1} бинарного кода. Вход синхронизации a_0 моделирует текущую позицию j с весом ξ_{2-j} с программируемым основанием 2^{2-j} для $j = 0, 1, 2$.

По адресным входам a_k таблица синтезирует двоичный код от 0 до 7 и соответствует таблице дешифратора, программируемой стандартным образом. Начинается в нулевом состоянии с нулей и заканчивается на седьмом состоянии единицами. Нулевая позиция с основанием a_0 заполняется по очереди (в разрядку) термами 0 и 1, по первому адресу a_1 с разрядкой в два раза (два нуля – две единицы) и с раз-

рядкой через четыре на второй позиции для адреса a_2. При расширении таблицы дешифратора разрядка удваивается по степенному полиному 2^k бинарного кода.

Выходная таблица мультиплексора по выходам Q_1, Q_2 программирует операнды в кодах таблицы дешифратора и моделирует алгоритм поразрядного уравновешивания для реализации метода половинного сечения. В двоичном коде операторы деления или умножения на два организуют циклическим сдвигом операнда на одну позицию влево или вправо. Исходные операнды половинного $N_{max}/2$ и максимального N_{max} (или минимального $N = 0$) кода размещают на нулевом или первом состояниях по выходам Q соответственно рангу. В предлагаемом примере $N_{max} = 4$, $N_{max}/2 = 2$ (001_2, 010_2) и их программируют в первой и нулевой строке таблицы мультиплексора по выходам $\{Q_1, Q_2, Q_0\}$ с основаниями {1, 2, 4}. Исходные операнды по выходам Q указывают адреса перехода при нулевом импульсе F для $j = 0$, при этом на адресе a_0 формируется минтерм $\xi_2 2^2$ второй позиции, так как $k = 2 - j$, а нулевое и первое состояние организуют старт программы.

Исходное состояние старта программы регламентирует адрес $\{a_1, a_2\}$={0, 0} за счет пересылки операнда по выходам $\{Q_1, Q_2, Q_0\}$ == {0, 0, 1} на исходный адрес $\{a_1, a_2, a_0\}$ = {0, 0, 1}, а при маскировании тактового входа подтверждается адрес старта {0, 0}. Поэтому в исходном состоянии моделируется статика по адресу {0, 0}, а на шине Q сформирован максимальный цифровой эквивалент, т.е. $N = 4$. Если на адресном входе a_0 появляется нулевой импульс $F = 1$ единичного уровня – исходный адрес {0, 0, 1} подтверждается, а адрес пересылки не изменяется. Таким образом моделируется статический режим: программа не переключается, а на выходной шине Q фиксируется код $N = 4$. При отсутствии нулевого импульса $F = 0$ генерируется нулевой адрес {0, 0, 0} по входам a_k, а на шине Q формируется адрес перехода {0, 1} и половина максимального эквивалента $N = 2$.

По нулевому импульсу $F = 0$ программа переключается с нулевого на второй адрес {0, 1}четвертого состояния. При отсутствии импульса $F = 0$ указывается адрес {1, 0} и эквивалент уменьшается в два раза $N = 1$. Программа пересылается на первый адрес и при $F = 0$ останавливается на втором состоянии, так как пересылается на нулевой адрес {0, 0} по выходам Q, который сформирован на них. Это обусловлено адресом a_0, моделирующим нулевое основание и маскирующим логическую единицу нулевым уровнем. Возврат на текущий адрес организует команду «СТОП», фиксируется второе состояние, а на выходах Q регистрируется нулевой эквивалент $N = 0$.

Если по первому адресу {1, 0} появляется единичный уровень $F = 1$ на входе a_0 синхронизации, то программа останавливается на третьем состоянии и формируется эквивалент единицы $N = 1$. Команда статики обусловлена равенством кода пересылки на шине Q входному адресу {1, 0} третьей строки. Аналогично синтезированы цифровые меры 2 и 3 на шестой и седьмой строке таблицы состояний, а пятая строка организует условия за счет пересылки со второго {0, 1} на третий {1, 1} адрес при появлении импульса $F = 1$ единичного уровня по входу a_0 синхронизации.

Приведенный пример трансформируется в многомерную матрицу по методам аналогии размерностью $2(k + 1)2^{k+1}$ с числом позиций $k = \log_2 N_2$ представления бинарного кода N_2. При этом таблица с k-м рангом базируется на матрицах $(k - 1)$ уровня и является основой таблиц состояния старшего $(k + 1)$-го базиса. Это соответствует интеграции программы из однотипных подпрограмм, синтезирующих подобные модули. Синтез и анализ программ по аналогии интеграции дифференцированных модулей из подобных морфологических (структур и связей) и функциональных (алгоритмов и моделей) признаков является методом проектирования эквивалентного программного обеспечения (блок-схем и языков программирования, таблиц кодов и состояния). Метод проектирования программ по эквивалентам отличается от комбинаторного программирования высокой метрологической и технологической, экономической и эргономической эффективностью. При этом хаотический перебор массива узкоспециализированных решений заменяется информационной технологией создания оптимального программного продукта для гибкой архитектуры АЦП адаптивных автоматических интерфейсов ввода-вывода коммуникабельных микропроцессорных систем.

Метод эквивалентных программ

Метод эквивалентных программ [22] предназначен для синтеза и анализа программного обеспечения на различных иерархических уровнях (от таблиц кодов до блок-схем программ) по принципам аналогии. Программирование таблицы состояния включает:

– синтез таблицы дешифратора в системе счисления заданных кодов (в бинарном коде по стандарту в диапазоне от минимального до максимального числа);

– анализ таблицы кодов дешифратора методами булевой алгебры или аналогии;

– использование спроектированной таблицы в качестве эквивалента для синтеза таблицы мультиплексора по операторам исчисления за счет техники адресации;

– анализ адресов пересылки операндов таблицы мультиплексора при формировании адресной последовательности в таблице дешифратора для синтеза исследуемого алгоритма функции;

– оценку исследуемой функции с образцовой и синтез программ более высокого ранга из эквивалентных модулей, сформированных из спроектированных таблиц состояния адекватных функций.

Наглядность и простота, оперативность и экономичность, прямой алгоритм проектирования оптимального по нормированным эквивалентам технического решения и открытая архитектура с адаптацией к априорной информации позволяют автоматизировать метод эквивалентных программ для создания современных информационных технологий. Благодаря этому гибкая архитектура АЦП поразрядного уравновешивания развивается и в перспективе достигнет уровня интеллектуальных кибернетических интерфейсов.

Выводы

1. Счетчики и регистры в комбинаторной логике синтезируют по их определениям, систематизирующим закономерности последовательного и параллельного соединения динамически триггеров ИС, по правилам функционирования которых анализируют (оценивают) правильность проектирования СИС.

2. Вектор развития последовательностных СИС направлен от жесткой структуры счетчика с бинарным кодом к управляемой архитектуре регистра с программируемым кодом.

3. Программно управляемые последовательностные СИС в матричной логике синтезируют методами программирования по эквивалентам с открытой архитектурой за счет их тиражирования по аналогии в ассоциативном адресном пространстве для организации информационной технологии проектирования БИС и микропроцессорных средств.

4. Показано повышение эффективности информационной технологии на примере проектирования методами программирования по эквивалентам динамических триггеров и счетчиков, регистров и генераторов, обусловленное тождественностью таблиц вектора состояния исследуемого решения и физического эквивалента.

4 КОМПОНЕНТЫ МИКРОПРОЦЕССОРНЫХ СРЕДСТВ

Проведем информационный анализ микропроцессорных средств: вычислителей с жесткой структурой и гибкой архитектурой для изучения математического обеспечения и метрологических средств микропроцессоров на уровне обобщенной архитектуры и логического устройства, регистров кода операции и признаков.

С позиций информационной концепции в процессе интеграции информационных процессов формируются аппаратные средства при развитии полупроводниковых приборов (ПП) в малые (ИС) и средние (СИС) интегральные схемы. Функция хранения инициирует в больших (БИС) интегральных схемах появление программного обеспечения, составляющего неделимую архитектуру совместно с аппаратными средствами. Математическое обеспечение определяется функцией вычисления при дальнейшей интеграции БИС в персональные компьютеры (ПК) (сверхбольшие ИС). Функция измерения организует метрологические средства микропроцессорных измерительных средств (МИС) с упорядоченным информационным обеспечением, неделимыми компонентами которого служат также аппаратные средства, программное и математическое обеспечение [12, 13, 15, 16, 22]. Проведем сопоставительный анализ компонент микропроцессорных средств на уровне вычислителей и БИС.

4.1 ВЫЧИСЛИТЕЛИ

Покажем развитие вычислителей от жесткой структуры до архитектуры персонального компьютера в процессе анализа по гибкости управления структурами и связями арифметико-логических устройств.

Венцом научно-технической революции XX века является информатизация [16], обусловленная развитием электронно-вычислительных средств от жесткой структуры с регламентированными связями [9, 13, 30, 45, 50 – 54] до гибкой архитектуры с коммуникабельным информационным обеспечением [5 – 31, 34 – 51, 60 – 72]. Основой информатизации XXI века служит персональный компьютер (ПК), орга-

низованный на микропроцессоре – цифровой программно управляемой БИС [13, 16]. ПК индивидуального пользования заменил в 90-х годах двадцатого столетия коллективные цифровые электронно-вычислительные машины (ЦВМ), выполненные на процессорах – цифровых программно управляемых вычислительных модулях [35, 46, 60 – 64, 69 – 72]. Интенсивное развитие микроэлектроники от полупроводниковых приборов до интегральных схем привело к конкуренции цифровых ЦВМ и аналоговых АВМ вычислительных машин в 70-х годах с неоспоримым лидерством цифровых вычислителей за счет гибкости и программируемости архитектуры [54]. Электронно-вычислительным машинам долгое время не уступали по метрологическим и экономическим характеристикам их прототипы – вычислители с жесткой структурой (ВЖС), созданные на заре электроники в 30-е годы [45, 46, 54, 72] и совершенствующиеся по пути технологической интеграции микроэлектронных структур в постоянно запоминающие устройства (ПЗУ).

Краткий анализ НТР по пути информатизации показывает развитие вычислителей в процессе интеграции базисных структур микроэлектроники от средств с жесткой структурой ВЖС и АВМ к гибкой архитектуре ЦВМ и ПК, систематизированных в табл. 4.1 на уровне структур $F(R)$, формул $F(\Phi)$ и мнемосхем $F(T)$ арифметико-логических устройств (АЛУ). Оценку вычислителей по гибкости управления проведем на примере реализации вычислительной функции [13, 21]

$$F = \frac{A}{x}\exp\left(\frac{B}{x}\right),$$

включающей четыре оператора $f_i \neq f_{i+1}$, $F = \{f_i\}_0^3$. Это арифметические операторы деления f_1 (÷), умножения f_2 (×), умножения-деления f_4 (YZ/T) и алгебраическое экспоненцирование f_3 (e^Y). Используя метод аналогии, синтезируем по математическому образу $F(\Phi)$ мнемосхемы $F(T)$ алгоритмов функционирования АЛУ вычислителей (см. табл. 4.1, $F(T)$).

4.1.1 Вычислители с жесткой структурой

По правилам комбинаторной логики ИС вычислители ВЖС, АВМ и ЦВМ [13, 35, 54] организуют мнемосхему АЛУ из последовательного включения со входа X на выход F блоков функций f_1, f_2, f_3, f_4, с дополнительной связью между последним f_4 и первым f_1 блоками (см. табл. 4.1, $1F(T) \cdot 3F(T)$). Первый блок организует деление входной переменной X по оператору $f_1 = 1/X$ с выдачей результата на входы блоков f_2 и f_4. Во втором блоке обратная величина $1/X$ перемножается на постоянную B и экспоненциируется в третьем блоке $f_3 = \exp f_2 = \exp(B/X)$. Множительно-делительный блок реализует перемножение функций f_1 и f_3 на параметр усиления A, а результат $f_4 = Af_1f_3 = A/X \exp(B/X)$ тождественен заданной функции, принимаемой за эквивалентный образ. Тождественность структурных формул исследуемой мнемосхемы $1F(T)$ и эквивалента $F = 1F(T)$ подтверждает правильность проектирования АЛУ вычислителя с жесткой структурой.

Анализ мнемосхемы $1F(T)$ показывает, что функция ВЖС конструируется по жесткой структуре $S = \{s_i\}_0^n$ с регламентированными связями $\{c_{i,j}\}_0^m = C$, т.е. $F(S, C)$. Множество функционально законченных структур $f_i \neq f_{i+1}$ тождественны различным функциям $s_i \equiv f_i$ с постоянными связями $c_{i,j} = \text{const}$ и неизменными структурами $s_i = \text{const}$. Функция $1F(T) = F(S, C)$ однозначно копирует регламентируемый алгоритм, инвариантный вычислительной функции $F = F(S, C) = \text{const}$. Высокие показатели метрологических характеристик «покупаются» низкой технологичностью мелкосерийных схем с уникальными структурами и связями, что приводит к необоснованным интеллектуальным, материальным и экономическим затратам. Перспективой развития ВЖС является замена комбинаторной логики ИС на упорядоченную матричную логику ПЗУ с масочным программированием на заводе-изготовителе.

4.1.2 Аналоговая вычислительная машина

Аналоговая вычислительная машина [54] (см. табл. 4.1, 2) повышает гибкость вычислительной функции F за счет введения управляемых связей $c_{i,k} = \text{var}$ с избыточной адресацией $k = \overline{0, m}$ между фиксированными функционально законченными структурами s_i инвариантных функций $s_i \equiv f_i = \text{const}$. Знакоместа k-х связей i-х структур изменяются аппаратно пользователем АВМ с помощью переключателей и штеккеров, разъемов и проводников. Мнемосхема $2F(T)$ позволяет аппаратно управлять функцией F АЛУ за счет коммутаций избыточными связями $c_i = \{c_{i,k}\}_0^n$ между фиксированными структурами $\{f_i\}_0^n$,

что повышает в $n^2/2$ раз гибкость АВМ относительно ВЖС. Однозначность функционально законченных неизменных структур $s_i \neq s_{i+1}$ ВЖС и АВМ соответствует их жесткой структурной схеме 1, $2F(R)$ с регламентированными связями. Структурная схема 1, $2F(R)$ реализует последовательное соединение устройств ввода УВ, вычисления ВУ и вывода УВыв, выполненных функционально законченными конструктивами, специализированными под конкретные вычислительные операции. По методам аналогии мнемосхема АЛУ однозначно иллюстрирует алгоритм функционирования ВУ, который совместно с устройствами ввода и вывода отражает реальную функцию ВЖС и АВМ в комбинаторном представлении.

Достоинством вычислителей с жестким алгоритмом является тождественность мнемосхем АЛУ математическому образу функции в явном виде. В АВМ характерны линейные преобразования аналоговых сигналов для организации уникальных прецизионных вычислительных функций в реальном масштабе времени, но отсутствие памяти для хранения информации не позволяет программно управлять информационными процессами. Перспективой развития АВМ является замена линейных интегральных схем и аппаратно коммутируемых связей на автоматические интерфейсы ввода-вывода, реализуемые на АЦП и ЦАП с программным управлением операторами вычисления.

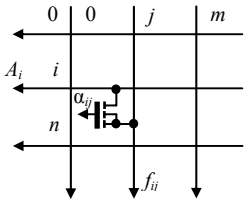
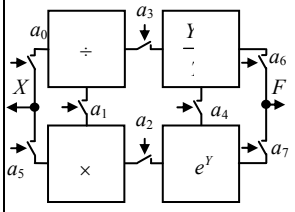
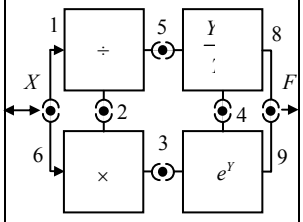
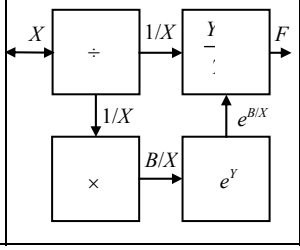
4.1.3 Цифровая вычислительная машина

Цифровую вычислительную машину [35] (табл. 4.1, 3) отличает от АВМ [54] программное обеспечение, обусловленное памятью последовательностных цифровых интегральных схем, которые замещают линейные интегральные схемы, позволяющие продолжительное время хранить информацию. Программное управление [13] заменяет электромеханические связи $c_{i,k}$ на электронные коммутаторы $A_j = \{a_{ij}\}_0^n$ адресного пространства, систематизированного по правилам программирования. Мнемосхема $3F(T)$ АЛУ ЦВМ включает функционально законченные структуры тождественных функций $f_i \neq f_{i+1}$ с фиксированными операторами $f_i = \text{const}$ и программируемыми связями c_{ij} по адресам $a_{ij} = \{0, 1\} = \text{var}$. Позиции i и j определяют соответственно номер функции f_i j-й подстановки программы. Программа организована

целенаправленной последовательностью подстановок j-х операторов $F_j = \sum_{i=0}^{n-1} a_{ij} f_i$, $j = \overline{1, m}$. Подстановки тождественны кодам и структурным формулам в нормальной дизъюнктивной $F(1)$ или конъюнктивной $F(0)$ формах. Форму представления подстановок выбирает разработчик АЛУ, а пользователь по регламентированным правилам составляет вычислительные программы для ЦВМ.

Мнемосхема $3F(T)$ по программе реализует заданную функцию F. Первая подстановка по адресу $a_{01} = 1$ коммутирует вход с переменной X с блоком деления (÷), выполняющим оператор $f_1(x) = 1/X$. На втором шаге по адресу $a_{12} = 1$ функционирует второй блок и умножает результат на постоянную B, который экспоненциирует третий блок (e^Y) по адресу $a_{23} = 1$ на третьем шаге. При этом выполняется функция $f_3[f_2(f_1)] = \exp(B/X)$, результат которой хранится в блоке экспоненцирования (e^Y). Четвертая подстановка параллельно включает адреса $a_{14} = a_{34} = 1$ и множительно-делительной устройство (YZ/T) перемножает результаты функций f_1 и f_3 на коэффициент A, что соответствует оператору $f_4 = a_{14} f_1 A \cdot a_{34} f_3$. На пятом шаге инициализируется адрес $a_{46} = 1$, коммутирующий результат $f_4 = F$ на выход $3F(T) = A/X \exp(B/X)$, что соответствует заданному эквиваленту $F = 3F(T)$.

В отличие от вычислителей с жесткой структурой ЦВМ реализует множество функций по заданным априори программам, что позволяет автоматизировать вычислительные процедуры и информационные процессы. Следует обратить внимание на изменение структурной схемы

	$F(T)$	$F(\Phi)$	$F(R)$
4 ПК		1. $F_k = \sum_{j=0}^{m} \beta_{kj} \prod_{i=0}^{n} \left(\overline{\alpha_{ij}} + A_i\right)$ 2. $f_{ijk} = f_{i+1,j,k} = \text{var}$ 3. $C = \{\alpha_{ij}, \beta_{kj}\} = \text{var}$	ИП М(П) ИВВ
3 ЦВМ		1. $F_j = \sum_{i=0}^{n-1} \alpha_{ij} f_i$ 2. $f_{i+1} \neq f_i = \text{const}$ 3. $c_{i,j} = \alpha_{ij} = \text{var}$	
2 АВМ		1. $F\left[\{s_i\}_0^n, \{c_k\}_0^m\right]$ 2. $s_i = f_i = \text{const}$ 3. $c_{i,k} = \text{var}$	УВыв ВУ УВ
1 ВЖС		1. $F(S, C)$ $S = \{s_i\}_0^n;\ F = \{f_i\}_0^n$ 2. $s_i = f_i = \text{const}$ 3. $c_{i,j} = \text{const}$	

ЦВМ за счет замены устройств с жесткой структурой на интерфейсы с гибкой архитектурой. Под архитектурой понимают неделимую совокупность аппаратных средств и программного обеспечения. Программное обеспечение ЦВМ отражает мнемосхема $3F(T)$ алгоритма программы вычисления, аппаратным средствам которой сопоставляют структуру 3, $4F(R)$. Структура ЦВМ и ПК соответствует одной из схем стандартных архитектур: кольцевой (последовательной), шинной (параллельной) и магистральной (смешанной) или их комбинаций. На приведенном примере 3, $4F(R)$ показана магистральная архитектура ЦВМ, состоящая из процессора П на базе АЛУ $3F(T)$, к которому подключены интерфейсы ввода-вывода (ИВВ) и памяти (ИП).

Программно управляемое АЛУ $3F(T)$ с функционально законченными блоками регламентирует архитектуры процессора и ЦВМ для вычислений, основанных на операторах арифметики и алгебры, тригонометрии и высшей математики, упорядоченных в адресном пространстве мнемотехники. АЛУ процессора конструируют из необходимых функционально законченных операционных устройств, коммутируемых между собой по программе заданных подстановок в адресном пространстве вычислительных функций. Следовательно, процессор – это программно управляемый цифровой преобразователь сигнала для выполнения вычислительных функций в адресном пространстве ЦВМ. Структура процессора копирует матрицу АЛУ из комбинаторных блоков, но отличается от релейной логики различными операционными преобразователями в междуузлиях матрицы, что диктует микропрограммное управление.

4.1.4 Персональный компьютер

Микропрограммирование – достижение микропроцессора ПК [16, 46 – 50] (табл. 4.1, 4), мнемосхемой АЛУ 4$F(T)$ которого служит программируемая матрица с упорядоченным адресным пространством однотипных функционально незаконченных операторов $f_{ijk} = f_{i+1,\, j+1,\, k} = \text{var}$ [13, 16]. В отличие от блочной мнемосхемы 3$F(T)$ ЦВМ вычислительная функция f_i дифференцирована в jk-м адресном пространстве, что инициирует микропрограммный уровень управления ijk-ми логическими ключами в $n \times m \times l$-мерной ассоциации униполярных функций $f_{ijk} = \{\overline{0, 1}\}$. Микропрограммирование многомерной ассоциации организовано избыточными связями α_{ij}, α^*_{ij} матриц И, НЕ-И, а также β_{jk} матрицы ИЛИ, интегрированными в код подстановки $C = = \{\alpha_{ij}, \alpha^*_{ij}, \beta_{jk}\} = \text{var}$. Целенаправленная последовательность подстановок составляет программу, формирующую заданный априори вычислительный алгоритм АЛУ 4$F(T)$ по универсальной математической модели 4$F(\Phi)$ [13]:

$$F_k = \sum_{j=0}^{m} \beta_{kj} \prod_{i=0}^{n} \left(\overline{\alpha_{ij}} + A_i\right)\left(\overline{\alpha^*_{ij}} + \overline{A_i}\right),$$

где A_i – входные сигналы информации $A = \{A_i\}$.

Гибкость архитектуры ПК в jk раз выше ЦВМ за счет развития макро- в микропрограммное управление многомерной ассоциацией коммутаторов целенаправленными подстановками по универсальной математической модели в унитарном логическом пространстве. Соответственно, микропроцессор – это программно управляемый цифровой преобразователь сигнала для выполнения подстановки и функций сравнения. Подстановки необходимы для создания ствола программы, а функции сравнения – для ветвления программы. В отличие от процессора, выполняющего вычисления на макропрограммном уровне разнообразных операторов исчисления, микропроцессор настраивает ассоциацию ключей с микропрограммным управлением на коммутацию логических функций в унитарном пространстве для организации макропрограммирования вычислительных функций.

Структурная схема ПК 3, 4$F(R)$ аналогична структуре ЦВМ, но вместо процессора П включен микропроцессор (М), диктующий микропрограммное управление интерфейсам ввода-вывода (ИВВ) и памяти (ИП). В отличие от реальной структурной схемы вычислителей ВЖС и АВМ, архитектуру ЦВМ и ПК иллюстрируют удобной для понимания топологии, подобной средствам с регламентированными связями, жесткой структурой $F(R)$, которую дополняют мнемосхемой $F(T)$ или программой. Для полного представления вычислительной функции и возможностей архитектуры описание ЦВМ и ПК организуют на уровне математического обеспечения (модели и методы, способы и алгоритмы) и метрологических средств (меры и критерии эффективности) для оценки коммуникабельности информационного обеспечения, универсальности и гибкости его компонент.

Таким образом, информатизация НТР развивает вычислители от жесткой структуры ВЖС и АВМ к гибкой архитектуре ЦВМ и ПК. В ЦВМ процессор создают по мнемосхеме АЛУ из функционально законченных модулей с макропрограммным управлением различными операторами вычисления, систематизированных в адресное пространство по правилам релейной логики. Архитектуру представляют неделимым комплексом мнемосхемы $F(T)$ программного обеспечения и структуры $F(R)$ аппаратных средств, причем структурную схему ЦВМ и ПК представляют стандартной топологией вычислителей с регламентированным алгоритмом. Микропроцессор ПК реализуют на программируемой матрице с мнемосхемой АЛУ в упорядоченном адресном пространстве логических ключей с микропрограммным управлением многомерной ассоциацией унитарных функций. Гибкость архитектуры оценивают по коммуникабельности ПК, включающего также универсальное математическое обеспечение и эффективные метрологические средства.

4.2 МИКРОПРОЦЕССОРЫ

Микропроцессор – это программно управляемый цифровой преобразователь для выполнения функций сравнения и подстановок. По информационной концепции, предполагающей интеграцию информационных процессов от обмена энергией к преобразованию сигнала, управления структурой к хранению (программированию) информации, программно управляемые цифровые преобразователи относятся к микроэлектронному базису больших интегральных схем (БИС). БИС по функции хранения или программного управления классифицируют на микропроцессоры (М), интерфейсы памяти (ИП) и ввода-вывода (ИВВ), организующие персональные компьютеры (ПК) при развитии функции программно управляемого преобразования в информационный процесс – вычисление [13, 21].

Микропроцессоры, как и другие БИС, реализуют на управляемых преобразователях сигнала – средних интегральных схемах (СИС), включающих счетчики и дешифраторы, мультиплексоры и регистры,

цифровые компараторы и логические устройства (ЛУ). Согласно информационной концепции микропроцессоры удобно систематизировать по функциям обмена и преобразования, управления и программирования, вычисления и измерения. Информационные процессы регламентируют основные признаки интегральных схем, а на уровне БИС определяют архитектуру микропроцессоров функции управления и обработки (вычисления), формирующие соответственно структурную схему аппаратных средств и алгоритм обработки программного обеспечения [21].

Топология схем зависит от соединения структур в координатах пространства R (X, Y, Z), которые включают параллельно, последовательно и смешанно. Согласно виду соединений по функции управления топологию микропроцессоров представляют шинной (параллельной), кольцевой (последовательной) и магистральной (смешанной) структурной схемой (см. табл. 4.2). Это продиктовано и программным управлением (хранением) информации в координатах пространство R – время T – функция Ф. Поэтому шина организует программное управление структурой в топологическом пространстве $F(R)$, кольцевая структура хранит информацию синхронно временным преобразованиям $F(T)$, а функциональное программирование $F(\Phi)$ – основа последовательно-параллельной, магистральной структуры.

4.2 Микропроцессоры

ПК / Функция	МиниЭВМ	МК	МикроЭВМ
Обработка	Программно управляемая	Программируемая	Микропрограммная
Хранение	Пространство	Время	Функция
Управление	Шина	Кольцо	Магистраль
Преобразование	Кодо-импульсное	Число-импульсное	Время-импульсное
Обмен	Инициируемый (программный)	Синхронный (сканируемый)	Асинхронный (приоритетный)

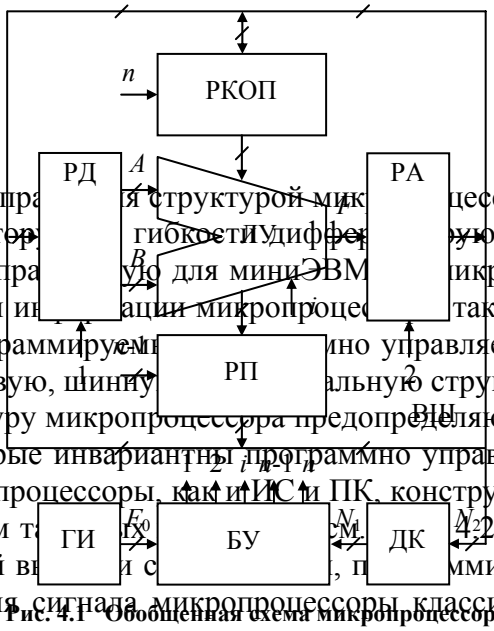


Рис. 4.1 Обобщенная схема микропроцессора

Правила управления структурой микропроцессора (рис. 4.1) инициирует алгоритм программы обработки ПК, которые по гибкости и дифференцируют на программируемую в микрокалькуляторах (МК), программно управляемую для миниЭВМ и с микропрограммным управлением в микроЭВМ. По функции обработки информации микропроцессоры также классифицируют на уровне программного обеспечения на программируемые, программно управляемые и с микропрограммным управлением, что определяет кольцевую, шинную и магистральную структуры (см. табл. 4.2).

Архитектуру микропроцессора предопределяют способы преобразования сигнала и режимы обмена энергии, которые инвариантны программно управляемой обработке информации. Относительно ритма обмена микропроцессоры, как и ИС и ПК, конструируют с синхронным, асинхронным и инициируемым сканированием та[illegible] 4.2) для организации последовательной, параллельной и ассоциативной в[illegible] с[illegible], п[illegible]раммируемым и приоритетным прерыванием. По способам преобразования сигнала микропроцессоры классифицируют [16, 49, 55] на кодо-, число- и времяимпульсные для программирования в пространственных R, временных T и функциональных Ф координатах с управлением по шинной, кольцевой и магистральной структуре (см. табл. 4.2).

Морфологическая табл. 4.2 систематизирует архитектуры микропроцессоров с согласованными способами информационных процессов для реализации рациональных архитектур персональных ком-

пьютеров и микропроцессорных измерительных средств (МИС), микропроцессорных систем (МПС) и сетей (МС). Различные способы информационных процессов дифференцированы по строкам морфологической таблицы, а рациональные архитектуры с согласованными преобразованиями интегрированы по столбцам на примере классификации ПК. Действительно, микрокалькуляторы включают программируемый числоимпульсный микропроцессор с кольцевой структурой и синхронным сканированием во временных координатах хранения информации. МиниЭВМ создают на программно управляемом кодоимпульсном микропроцессоре по шинной структуре и с инициируемым сканированием в пространственных координатах адресации. В отличие от них, микро-ЭВМ оперирует в функциональном адресном пространстве с приоритетным сканированием по магистральной структуре времяимпульсного микропроцессора с микропрограммным управлением.

Разнообразие микропроцессоров, обусловленное информационными процессами и способами их реализации, за счет систематизации последних в морфологическую таблицу позволяет выявить закономерности в виде принципов микросхемотехники для проектирования обобщенной архитектуры микропроцессора.

4.2.1 Обобщенная архитектура

Обобщенная архитектура интегрирует основные признаки различных микропроцессоров в условном векторном пространстве без конкретизации способов преобразования сигнала, с учетом которых структура трансформируется в стандартные схемы и программы с типовыми правилами адресации [16].

Структурная схема микропроцессора (см. рис. 4.1) содержит [9, 13, 42, 46, 49] логическое устройство (ЛУ), соединенное шинами *A*, *B*, *F* с регистрами данных (РД) и аккумулятора (РА) для обработки операндов, регистры кода операции (РКОП) и признаков (РП) для организации программы вычисления, внутреннюю шину (ВШ), объединяющую регистры и через дешифратор команд (ДК) блок управления (БУ) с генератором импульсов для синхронизации алгоритма обработки по программе.

Двухадресное ЛУ по каналам *A* и *B* принимает информацию из РД и загружает результаты вычислений в РА, которые по ВШ поступают для обработки в РД или для формирования подстановки программы в РКОП. РКОП дифференцирует код команды на код операции, адрес и операнд и инициирует через ДК микропрограмму БУ. ДК преобразует код N_2 операнда в начальный адрес N_1 выбранной в БУ микропрограммы, которую синхронизирует тактовой частотой F_0 генератор ГИ. БУ является микропроцессором в микропроцессоре, в адресном пространстве которого содержится банк типовых логических микроинструкций, «зашитых» в постоянное запоминающее устройство (ПЗУ). Программирование ПЗУ БУ осуществляется серийно на заводе-изготовителе или пользователем в процессе проектирования микропроцессорного средства. На выходе БУ по заданной микропрограмме генерируются импульсы синхронизации $i = \{\overline{1, n}\}$, управляющие моментом включения τ_{ij} блоков с *j*-м адресом. Следовательно, БУ по микропрограмме адресует включение блоков в координатах пространства *R* и времени *T* для выполнения логической функции Ф. Регистр признаков РП служит для ветвления программы, поступающей в РКОП, по состояниям матрицы ЛУ: обнуления и переполнения, вспомогательного переноса и знака модуля, прерывания вычислений и четности кода.

РКОП начинает следующий шаг программы последним импульсом *n* с БУ предыдущего цикла микропрограммы, заканчивающейся командой «ВОЗВРАТ». В регистр загружается код операции, по которому через ДК выбирает из ПЗУ БУ заданную микропрограмму. БУ синхронизирует микропрограммный цикл реализации логической функции по тактовым импульсам частоты F_0 генератора ГИ. На выходах *i* согласно микропрограмме появляются управляющие импульсы τ_{ij}, включающие в заданной последовательности блоки микропроцессора. Например, по коду операции логического сложения $F = A + B$ в РД последовательно загружаются по внутренней шине данные слагаемого *A* из РКОП и слагаемого *B* из РА по импульсам $\{n, 1, 2\}$ с БУ. По каналам *A* и *B* слагаемые поступают на информационные входы ЛУ, на управляющих входах которого сформирован из РКОП код операнда логического сложения. При генерации *i*-го импульса ЛУ суммирует данные *A* и *B*, а результат *F* заносится в РА после появления 2-го импульса. Цикл микропрограммы заканчивается командой «ВОЗВРАТ» *n*-м импульсом с БУ и в РКОП загружается код операции следующего шага подстановки программы. Начинается следующий шаг программы по микропрограмме БУ, выбранной через ДК по операнду РКОП.

Обобщенная схема микропроцессора (см. рис. 4.1) преобразуется в число-, время- или кодоимпульсную кольцевую, магистральную или шинную структуру за счет последовательного, смешанного или параллельного соединения регистров [13]. Например, при последовательном включении регистров РКОП и РП, РД и РА конструируется числоимпульсный микропроцессор с кольцевой структурой, а при

параллельном их объединении по координатам управления *R*, *T*, Ф создают кодоимпульсный микропроцессор с шинной структурой.

Стандартная структура регламентирует типовой формат кодов команд и операции. Для числоимпульсных микрокалькуляторных комплектов [12, 13, 55] микропроцессоров программный цикл суммируется из регистровых с дифференциацией на *RT*Ф циклы по тетраде импульсов двоично-десятичного кода. Команды формируются из двух полубайтов с определением операнда Ф, адреса *R* блока и знакоместа *T* синхронизации. Кодоимпульсный микропроцессор [5 – 13, 36 – 49] с программным управлением оперирует одно-, двух- и трехбайтными командами для внутри-, внешнеблочной и системной адресации. Байты дифференцируют по координатам управления, начиная с операнда *T*, адреса *R* блока и данных Ф. Одно- и двухбайтные команды адресуют код операции также в координатах «Что? – Ф», «Где? – *R*», «Когда? – *T*». В микропроцессорах с микропрограммным управлением [12 – 14, 38, 46] формат команд создают из четырех байт, адресующих по-байтно координаты {Ф, *R*, *T*} с использованием четвертого байта для организации циклов, вложений и прерываний.

Для уяснения алгоритмов работы ЛУ, регистров РКОП и РП ниже рассмотрим универсальную модель, технику адресации и признаки ветвления программ.

4.2.2 Логическое устройство

Программно управляемый функциональный логический преобразователь [13, 16] для выполнения логических операций назовем логическим устройством (ЛУ). Оно является центральным элементом, мозгом микропроцессора. Как функциональный преобразователь ЛУ относится к базису СИС комбинационного типа. От других комбинационных СИС логическое устройство отличает универсальная математическая модель, реализующая по программе алгоритмы от элементарных логических функций И, ИЛИ, НЕ до сложных преобразований дешифратора и кодера, мультиплексора и генератора. Матричная структура ЛУ микропроцессора принципиально отличается от релейной логики различных функционально законченных блоков арифметико-логического устройства (АЛУ) процессора. Если АЛУ, кроме логических счислений, реализует арифметические и алгебраические, дифференциальные и интегральные исчисления за счет коммутации вычислительных блоков, то ЛУ выполняет эти операторы только по программе на базе логических счислений по алгоритмам универсальной математической модели.

По числу входов ЛУ различают одно- и двухадресные. Одноадресные ЛУ принимают информацию по одному каналу, а двухадресные – по двум. Конструктивно ЛУ конструируют на логических матрицах НДФ *F*(1) или НКФ *F*(0) с программно управляемым полем (ПЛМ), обладающих функциональной полнотой и самодостаточностью за счет избыточности ассоциативных связей. Избыточность последовательных, параллельных и смешанных соединений логических ключей организует ассоциацию многомерных конъюнкторов, дизъюнкторов и инверторов для линейного преобразования сигналов с минимальным температурным, временным и параметрическим дрейфом относительно эквивалентной меры. Логическое устройство является перспективным развитием аналоговых линейных интегральных схем на базе операционного усилителя, но его параметрическая избыточность коэффициента усиления достигается в ЛУ топологической избыточностью программируемых строк и столбцов ПЛМ [13].

Функциональная полнота ЛУ обеспечена параллельным включением многомерных матриц И, НЕ-И, последовательно соединенных с матрицей ИЛИ при реализации адресного пространства в НДФ *F*(1). Структура ЛУ в единичном логическом пространстве *F*(1) приведена на рис. 4.2 [13, 21]. Преобразуемый сигнал *A* поступает на информационные входы матриц конъюнкции И-НЕ, И, код операций которых программируется векторами α*, α в $n \times m$-мерном адресном пространстве ПЛМ. На выходе матриц умножения формируется конъюнктивная функция *C*, преобразуемая матрицей ИЛИ в дизъюнкцию *F*, управляемую в $m \times l$-м поле адресов вектором β. Полный код операции ПЛМ программного управления данными $A = \{A_i\}_0^n$ для создания функции $F = \{f_k\}_0^l$ представлен многомерным вектором $\boldsymbol{N} = \{\alpha, \alpha^*, \beta\}$, где $\alpha = \{\alpha_{ij}\}_{0,0}^{n,m}$, $\alpha^* = \{\alpha_{ij}\}_{0,0}^{n,m}$, $\beta = \{\beta_{jk}\}_{0,0}^{m,l}$. Обобщенная математическая модель ПЛМ в векторной форме имеет вид

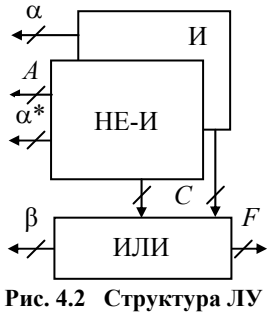


Рис. 4.2 Структура ЛУ

$$\boldsymbol{F} = \boldsymbol{N} \bullet \boldsymbol{A},$$

которая отражает взаимосвязь входных *A*, выходных *F* и управляющих *N* сигналов, но не позволяет представлять конкретные алгоритмы вычисления и адреса подстановок многомерного пространства.

Для вывода математической модели ЛУ преобразуем структуру ПЛМ к схеме замещения (рис. 4.3) в виде последовательного соединений мультиплексора НЕ ($MUX\ \overline{\&}$), конъюнктора И (&) и мультиплексора ИЛИ ($MUX1$). Это отражает физический смысл коммутаторов прямого A и инверсного $\overline{A}$ входных сигналов ij-х ячеек конъюнктивных матриц; j-х столбцов вентилей, выполняющих логическое умножение; а также коммутаторов конъюктивных сигналов $C = \{c_j\}_0^m$ суммирующей матрицей ИЛИ. Декомпозиция структуры ПЛМ сводит сложную задачу в векторной форме к итерационному решению типовых логических переключателей НЕ, И, ИЛИ, систематизированных в табл. 4.3.

Мультиплексор $1F(R)$ коммутирует сигналы A и $\overline{A}$ в вектор $a = \{a_{ij}\}$ при задании значений {0, 1} адресов α и α^*, комбинации возможных состояний которых систематизированы в векторную таблицу $1F(T)$ мультиплексора. Физика коммутации сигналов требует формирования на выходе a прямого A и инверсного $\overline{A}$ сигналов при адресации $\{\alpha, \alpha^*\}$ соответственно {1, 0} и {0, 1}, обнуления переключателя при активизации адресов логическими единицами {1, 1} и высокого потенциала единичного уровня $a_0 = 1$ при отсутствии потенциалов по адресу {0, 0}. По таблице $1F(T)$ синтезируем структурные формулы мультиплексора в единичном $F(1)$ и нулевом $F(0)$ логических пространствах.

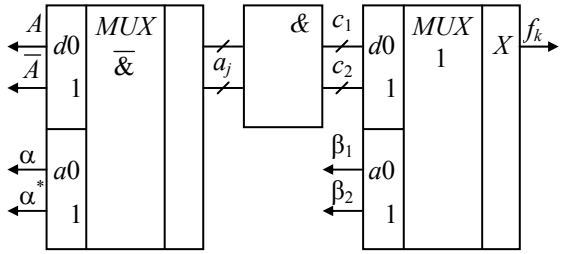


Рис. 4.3 Схема замещения

По правилам НДФ алгебры Буля единичная функция $F(1) = a$ мультиплексора НЕ состоит из суммы произведения минтермов:

$$a = \overline{\alpha}\overline{\alpha^*} + \alpha\overline{\alpha^*}A + \overline{\alpha}\alpha^*\overline{A}. \qquad (4.1)$$

Используя аксиому дизъюнкции ($A + \overline{A} = 1$), создадим четное число слагаемых для минимизации структурной формулы (4.1) при объединении подобных членов

$$a = \overline{\alpha}\overline{\alpha^*}\left(A + \overline{A}\right) + \alpha\overline{\alpha^*}A + \overline{\alpha}\alpha^*\overline{A},$$

что соответствует

$$a = \overline{\alpha}\,\overline{A}\left(\overline{\alpha^*} + \alpha^*\right) + \overline{\alpha^*}A\left(\overline{\alpha} + \alpha\right).$$

Применив аксиому дизъюнкции, находим формулу $a(1)$ в НДФ

$$a(1) = \overline{\alpha^*}A + \overline{\alpha}\,\overline{A}. \qquad (4.2)$$

Для определения функции $a(0)$ в НКФ применим дважды теорему Деморгана

$$a(0) = \overline{a(1)} = \overline{(\alpha^* + \overline{A})\,(\alpha + A)}$$

или после перемножения

$$a(0) = \overline{\alpha^*\alpha + \alpha^*A + \alpha\overline{A} + A\overline{A}}.$$

Учитывая аксиому конъюнкции ($\alpha\alpha^* = 0$ и $A\overline{A} = 0$) и после повторного преобразования по Деморгану, получим выражение

$$a(0) = (\overline{\alpha^*} + \overline{A})\,(\overline{\alpha} + A)\,. \tag{4.3}$$

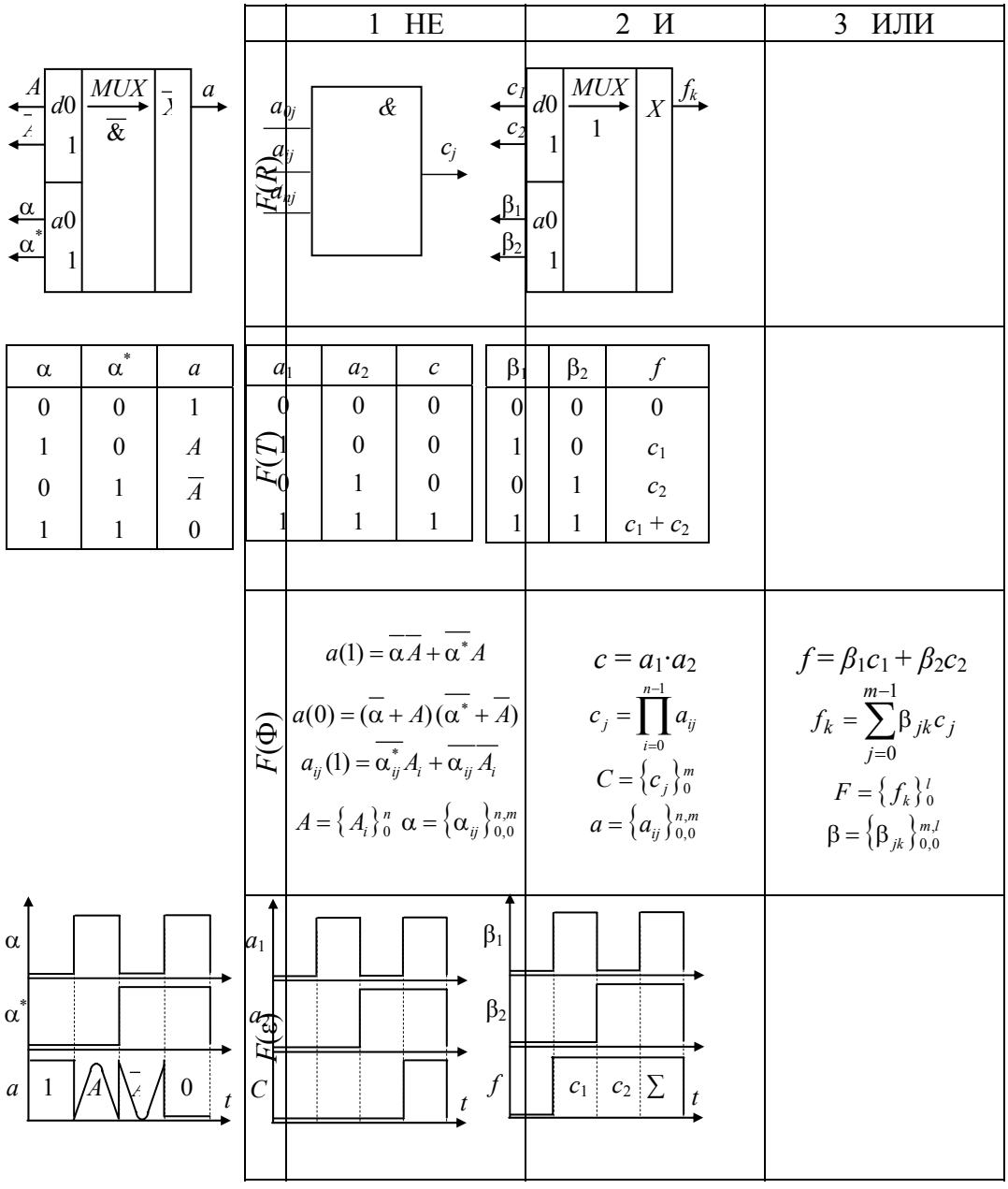
Структурные формулы (4.2) и (4.3) в векторной форме 1*F*(Ф) выразим в матричной форме для *ij*-го мультиплексора:

$$\begin{cases} a_{ij}(1) = \overline{\alpha^*_{ij}}\,A_i + \overline{\alpha_{ij}}\;\overline{A_i}\,; \\ a_{ij}(0) = \left(\overline{\alpha_{ij}} + A_i\right)\left(\overline{\alpha^*_{ij}} + \overline{A_i}\right). \end{cases} \tag{4.4}$$

Для двух ячеек a_1 и a_2 *j*-го столбца справедлива таблица 2*F*(*T*) конъюнкции, для которой истинное $C = 1$, если И a_1 И a_2 истинны $a_1 = a_2 = 1$. По таблице 2*F*(*T*) двух переменных несложно в НДФ записать структурную формулу

$$C = a_1\,a_2. \tag{4.5}$$

4.3 Логическое устройство



α	α*	a
0	0	1
1	0	A
0	1	Ā
1	1	0

a_1	a_2	c
0	0	0
1	0	0
0	1	0
1	1	1

β_1	β_2	f
0	0	0
1	0	c_1
0	1	c_2
1	1	$c_1 + c_2$

По методу индукции, заменяя двухвходовой конъюнктор функции (4.5) на многовходовой конъюнктор 2*F*(*R*), запишем в матричной форме логическое умножение 2*F*(Ф) *i*-х мультиплексоров НЕ *j*-го столбца

$$c_j = \prod_{i=0}^{n-1} a_{ij} \ . \tag{4.6}$$

После подстановки в выражение (4.6) зависимостей (4.4) находим формулы j-го столбца конъюнктора $2F(R)$ в НДФ и НКФ:

$$\begin{cases} c_j(1) = \prod_{i=0}^{n-1} \left(\overline{\alpha_{ij}^{*}} A_i + \overline{\alpha_{ij}\, A_i} \right); \\ c_j(0) = \prod_{i=0}^{n-1} \left(\overline{\alpha_{ij}} + A_i \right) \left(\overline{\alpha_{ij}^{*}} + \overline{A_i} \right). \end{cases} \tag{4.7}$$

Мультиплексор ИЛИ $3F(R)$ для двух переменных c_1 и c_2 k-й строки суммирующей матрицы управляется двумя адресами β_1 и β_2. Его таблица истинности $3F(T)$ обнулена при отключенных адресах $\beta_1 = \beta_2 = 0$, а при их активизации ($\beta_j = 1$) пропускает на выход f сигнал c_j по адресам $\{\beta_1, \beta_2\}$ равным {1, 0} и {0, 1}. Если активизированы $\beta_1 = \beta_2 = 1$, то по адресу {1, 1}на выходе f мультиплексора ИЛИ появляется суммарный сигнал $f_3 = c_1 + c_2$. Используя метод структурных формул, синтезируем в НДФ по таблице $3F(T)$ функцию мультиплексора $3F(R)$ для двух переменных:

$$f = \beta_1 \bar{\beta}_2 c_1 + \bar{\beta}_1 \beta_2 c_2 + \beta_1 \beta_2 (c_1 + c_2).$$

Объединяя подобные слагаемые

$$f = \beta_1 c_1 (\bar{\beta}_2 + \beta_2) + \beta_2 c_2 (\bar{\beta}_1 + \beta_1),$$

с учетом аксиомы дизъюнкции, находим решение для двухвходового коммутатора

$$f = \beta_1 c_1 + \beta_2 c_2. \tag{4.8}$$

При замене двух- на многовходовый мультиплексор получим по методу индукции формулу $3F(\Phi)$ m-мерного сумматора:

$$f_k = \sum_{j=0}^{m-1} \beta_{jk} c_j \ . \tag{4.9}$$

Подставляя в формулу (4.9) выражения (4.7), запишем в матричной форме математическую модель схемы замещения ПЛМ (см. рис. 4.3), адекватную структуре логического устройства:

$$\begin{cases} f_k(1) = \sum_{j=0}^{m-1} \beta_{jk} \prod_{i=0}^{n-1} \left(\overline{\alpha_{ij}^{*}} A_i + \overline{\alpha_{ij}\, A_i} \right); \\ f_k(0) = \sum_{j=0}^{m-1} \beta_{jk} \prod_{i=0}^{n-1} \left(\overline{\alpha_{ij}} + A_i \right) \left(\overline{\alpha_{ij}^{*}} + \overline{A_i} \right), \end{cases} \tag{4.10}$$

где код операции $N = \{\alpha_{ij}, \alpha^{*}_{ij}, \beta_{jk}\}$ программирования ij-го коммутатора матриц умножения по адресам α_{ij} – прямого сигнала A_i, соответственно α^{*}_{ij} – инверсного $\overline{A}_i$, а также управления jk-ми ключами матрицы ИЛИ за счет активизации β_{jk}-х адресов выходной функции $F = \{f_k\}_0^l$. Примеры реализации различных алгоритмов по модели (4.10) приведены в книге [13].

Анализ математических моделей (4.10) показывает их универсальность в $n \times m \times l$-мерном адресном пространстве, что инициирует гибкую архитектуру логического устройства с избыточной адресацией ассоциативных связей матричной структуры.

4.2.3 Адресация

Рассмотрим основные способы адресации [11, 13, 46, 49], реализуемые в микропроцессоре посредством регистра кода операции (РКОП) по коду команды. Код команды N_k определяет алгоритм функционирования блоков микропроцессора за счет программного управления ими в координатах пространство-время-функция. Для этого код команды N_k дифференцирован на код операции N_o, определяющий алгоритм микропрограммы БУ для синхронизации компонент микропроцессора, адрес A операнда, указывающий блок хранения и ячейку памяти для выборки информации (адреса или данных), и операнд в виде адреса или данных D, предназначенных для обработки, т.е. $N_k = \{N_0, A, D\}$.

Для работы с регистрами микропроцессора достаточна однобайтная команда, размещаемая в РКОП. По внутренней шине код операции N_o выбирает микропрограмму из БУ синхронизации блоков по адресу A команды для обработки данных D. Сервисные функции (обращение к портам) требуют двухбайтных команд, где первый байт регламентирует N_o регистра КОП, а адрес A или данные D размещают в регистрах или счетчике команд. При работе с интерфейсами памяти и ввода-вывода используют трехбайтные команды для адресации в них операндов по N_o регистра кода операции. Основой программирования служит рациональная техника адресации кода команды N_k, реализуемая кодом операции N_o РКОП.

Правила управления структурой микропроцессора (рис. 4.1) инициирует алгоритм программы обработки ПК, которую по гибкости дифференцируют на программируемую в микрокалькуляторах (МК), программно управляемую для миниЭВМ и с микропрограммным управлением в микроЭВМ. По функции обработки информации микропроцессоры также классифицируют на уровне программного обеспечения на программируемые, программно управляемые и с микропрограммным управлением, что определяет кольцевую, шинную и магистральную структуры (см. табл. 4.2).

Архитектуру микропроцессора предопределяют способы преобразования сигнала и режимы обмена энергии, которые инвариантны программно управляемой обработке информации. Относительно ритма обмена микропроцессоры, как и ИС и ПК, конструируют с синхронным, асинхронным и инициируемым сканированием тактовых импульсов (см. табл. 4.2) для организации последовательной, параллельной и ассоциативной выборки с аппаратным, программируемым и приоритетным прерыванием. По способам преобразования сигнала микропроцессоры классифицируют [16, 49, 55] на кодо-, число- и времяимпульсные для программирования в пространственных R, временных T и функциональных Ф координатах с управлением по шинной, кольцевой и магистральной структуре (см. табл. 4.2).

Морфологическая табл. 4.2 систематизирует архитектуры микропроцессоров с согласованными способами информационных процессов для реализации рациональных архитектур персональных компьютеров и микропроцессорных измерительных средств (МИС), микропроцессорных систем (МПС) и сетей (МС). Различные способы информационных процессов дифференцированы по строкам морфологической таблицы, а рациональные архитектуры с согласованными преобразованиями интегрированы по столбцам на примере классификации ПК. Действительно, микрокалькуляторы включают программируемый числоимпульсный микропроцессор с кольцевой структурой и синхронным сканированием во временных координатах хранения информации. МиниЭВМ создают на программно управляемом кодоимпульсном микропроцессоре по шинной структуре и с инициируемым сканированием в пространственных координатах адресации. В отличие от них, микро-ЭВМ оперирует в функциональном адресном пространстве с приоритетным сканированием по магистральной структуре времяимпульсного микропроцессора с микропрограммным управлением.

Разнообразие микропроцессоров, обусловленное информационными процессами и способами их реализации, за счет систематизации последних в морфологическую таблицу позволяет выявить закономерности в виде принципов микросхемотехники для проектирования обобщенной архитектуры микропроцессора.

4.2.1 Обобщенная архитектура

Обобщенная архитектура интегрирует основные признаки различных микропроцессоров в условном векторном пространстве без конкретизации способов преобразования сигнала, с учетом которых структура трансформируется в стандартные схемы и программы с типовыми правилами адресации [16].

Структурная схема микропроцессора (см. рис. 4.1) содержит [9, 13, 42, 46, 49] логическое устройство (ЛУ), соединенное шинами A, B, F с регистрами данных (РД) и аккумулятора (РА) для обработки операндов, регистры кода операции (РКОП) и признаков (РП) для организации программы вычисления, внутреннюю шину (ВШ), объединяющую регистры и через дешифратор команд (ДК) блок управления (БУ) с генератором импульсов для синхронизации алгоритма обработки по программе.

Двухадресное ЛУ по каналам *A* и *B* принимает информацию из РД и загружает результаты вычислений в РА, которые по ВШ поступают для обработки в РД или для формирования подстановки программы в РКОП. РКОП дифференцирует код команды на код операции, адрес и операнд и инициирует через ДК микропрограмму БУ. ДК преобразует код N_2 операнда в начальный адрес N_1 выбранной в БУ микропрограммы, которую синхронизирует тактовой частотой F_0 генератор ГИ. БУ является микропроцессором в микропроцессоре, в адресном пространстве которого содержится банк типовых логических микроинструкций, «зашитых» в постоянное запоминающее устройство (ПЗУ). Программирование ПЗУ БУ осуществляется серийно на заводе-изготовителе или пользователем в процессе проектирования микропроцессорного средства. На выходе БУ по заданной микропрограмме генерируются импульсы синхронизации $i = \overline{\{1, n\}}$, управляющие моментом включения τ_{ij} блоков с *j*-м адресом. Следовательно, БУ по микропрограмме адресует включение блоков в координатах пространства *R* и времени *T* для выполнения логической функции Ф. Регистр признаков РП служит для ветвления программы, поступающей в РКОП, по состояниям матрицы ЛУ: обнуления и переполнения, вспомогательного переноса и знака модуля, прерывания вычислений и четности кода.

РКОП начинает следующий шаг программы последним импульсом *n* с БУ предыдущего цикла микропрограммы, заканчивающейся командой «ВОЗВРАТ». В регистр загружается код операции, по которому через ДК выбирает из ПЗУ БУ заданную микропрограмму. БУ синхронизирует микропрограммный цикл реализации логической функции по тактовым импульсам частоты F_0 генератора ГИ. На выходах *i* согласно микропрограмме появляются управляющие импульсы τ_{ij}, включающие в заданной последовательности блоки микропроцессора. Например, по коду операции логического сложения $F = A + B$ в РД последовательно загружаются по внутренней шине данные слагаемого *A* из РКОП и слагаемого *B* из РА по импульсам {*n*, 1, 2} с БУ. По каналам *A* и *B* слагаемые поступают на информационные входы ЛУ, на управляющих входах которого сформирован из РКОП код операнда логического сложения. При генерации *i*-го импульса ЛУ суммирует данные *A* и *B*, а результат *F* заносится в РА после появления 2-го импульса. Цикл микропрограммы заканчивается командой «ВОЗВРАТ» *n*-м импульсом с БУ и в РКОП загружается код операции следующего шага подстановки программы. Начинается следующий шаг программы по микропрограмме БУ, выбранной через ДК по операнду РКОП.

Обобщенная схема микропроцессора (см. рис. 4.1) преобразуется в число-, время- или кодоимпульсную кольцевую, магистральную или шинную структуру за счет последовательного, смешанного или параллельного соединения регистров [13]. Например, при последовательном включении регистров РКОП и РП, РД и РА конструируется числоимпульсный микропроцессор с кольцевой структурой, а при параллельном их объединении по координатам управления *R*, *T*, Ф создают кодоимпульсный микропроцессор с шинной структурой.

Стандартная структура регламентирует типовой формат кодов команд и операции. Для числоимпульсных микрокалькуляторных комплектов [12, 13, 55] микропроцессоров программный цикл суммируется из регистровых с дифференциацией на *RT*Ф циклы по тетраде импульсов двоично-десятичного кода. Команды формируются из двух полубайтов с определением операнда Ф, адреса *R* блока и знакоместа *T* синхронизации. Кодоимпульсный микропроцессор [5 – 13, 36 – 49] с программным управлением оперирует одно-, двух- и трехбайтными командами для внутри-, внешнеблочной и системной адресации. Байты дифференцируют по координатам управления, начиная с операнда *T*, адреса *R* блока и данных Ф. Одно- и двухбайтные команды адресуют код операции также в координатах «Что? – Ф», «Где? – *R*», «Когда? – *T*». В микропроцессорах с микропрограммным управлением [12 – 14, 38, 46] формат команд создают из четырех байт, адресующих по-байтно координаты {Ф, *R*, *T*} с использованием четвертого байта для организации циклов, вложений и прерываний.

Для уяснения алгоритмов работы ЛУ, регистров РКОП и РП ниже рассмотрим универсальную модель, технику адресации и признаки ветвления программ.

4.2.2 Логическое устройство

Программно управляемый функциональный логический преобразователь [13, 16] для выполнения логических операций назовем логическим устройством (ЛУ). Оно является центральным элементом, мозгом микропроцессора. Как функциональный преобразователь ЛУ относится к базису СИС комбинационного типа. От других комбинационных СИС логическое устройство отличает универсальная математическая модель, реализующая по программе алгоритмы от элементарных логических функций И, ИЛИ, НЕ до сложных преобразований дешифратора и кодера, мультиплексора и генератора. Матричная структура ЛУ микропроцессора принципиально отличается от релейной логики различных функционально законченных блоков арифметико-логического устройства (АЛУ) процессора. Если АЛУ, кроме

логических счислений, реализует арифметические и алгебраические, дифференциальные и интегральные исчисления за счет коммутации вычислительных блоков, то ЛУ выполняет эти операторы только по программе на базе логических счислений по алгоритмам универсальной математической модели.

По числу входов ЛУ различают одно- и двухадресные. Одноадресные ЛУ принимают информацию по одному каналу, а двухадресные – по двум. Конструктивно ЛУ конструируют на логических матрицах НДФ $F(1)$ или НКФ $F(0)$ с программно управляемым полем (ПЛМ), обладающих функциональной полнотой и самодостаточностью за счет избыточности ассоциативных связей. Избыточность последовательных, параллельных и смешанных соединений логических ключей организует ассоциацию многомерных конъюнкторов, дизъюнкторов и инверторов для линейного преобразования сигналов с минимальным температурным, временным и параметрическим дрейфом относительно эквивалентной меры. Логическое устройство является перспективным развитием аналоговых линейных интегральных схем на базе операционного усилителя, но его параметрическая избыточность коэффициента усиления достигается в ЛУ топологической избыточностью программируемых строк и столбцов ПЛМ [13].

Функциональная полнота ЛУ обеспечена параллельным включением многомерных матриц И, НЕ-И, последовательно соединенных с матрицей ИЛИ при реализации адресного пространства в НДФ $F(1)$. Структура ЛУ в единичном логическом пространстве $F(1)$ приведена на рис. 4.2 [13, 21]. Преобразуемый сигнал A поступает на информационные входы матриц конъюнкции И-НЕ, И, код операций которых программируется векторами α^*, α в $n \times m$-мерном адресном пространстве ПЛМ. На выходе матриц умножения формируется конъюнктивная функция C, преобразуемая матрицей ИЛИ в дизъюнкцию F, управляемую в $m \times l$-м поле адресов вектором β. Полный код операции ПЛМ программного управления данными $A = \{A_i\}_0^n$ для создания функции $F = \{f_k\}_0^l$ представлен многомерным вектором $\boldsymbol{N} = \{\alpha, \alpha^*, \beta\}$, где $\alpha = \{\alpha_{ij}\}_{0,0}^{n,m}$, $\alpha^* = \{\alpha_{ij}\}_{0,0}^{n,m}$, $\beta = \{\beta_{jk}\}_{0,0}^{m,l}$. Обобщенная математическая модель ПЛМ в векторной форме имеет вид

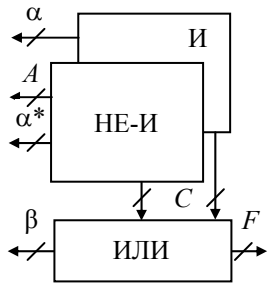


Рис. 4.2 Структура ЛУ

$$\boldsymbol{F} = \boldsymbol{N} \bullet \boldsymbol{A},$$

которая отражает взаимосвязь входных A, выходных F и управляющих N сигналов, но не позволяет представлять конкретные алгоритмы вычисления и адреса подстановок многомерного пространства.

Для вывода математической модели ЛУ преобразуем структуру ПЛМ к схеме замещения (рис. 4.3) в виде последовательного соединений мультиплексора НЕ ($MUX\ \overline{\&}$), конъюнктора И (&) и мультиплексора ИЛИ ($MUX1$). Это отражает физический смысл коммутаторов прямого A и инверсного $\overline{A}$ входных сигналов ij-х ячеек конъюнктивных матриц; j-х столбцов вентилей, выполняющих логическое умножение; а также коммутаторов конъюктивных сигналов $C = \{c_j\}_0^m$ суммирующей матрицей ИЛИ. Декомпозиция структуры ПЛМ сводит сложную задачу в векторной форме к итерационному решению типовых логических переключателей НЕ, И, ИЛИ, систематизированных в табл. 4.3.

Мультиплексор $1F(R)$ коммутирует сигналы A и $\overline{A}$ в вектор $a = \{a_{ij}\}$ при задании значений {0, 1} адресов α и α^*, комбинации возможных состояний которых систематизированы в векторную таблицу $1F(T)$ мультиплексора. Физика коммутации сигналов требует формирования на выходе a прямого A и инверсного $\overline{A}$ сигналов при адресации $\{\alpha, \alpha^*\}$ соответственно {1, 0} и {0, 1}, обнуления переключателя при активизации адресов логическими единицами {1, 1} и высокого потенциала единичного уровня $a_0 = 1$ при отсутствии потенциалов по адресу {0, 0}. По таблице $1F(T)$ синтезируем структурные формулы мультиплексора в единичном $F(1)$ и нулевом $F(0)$ логических пространствах.

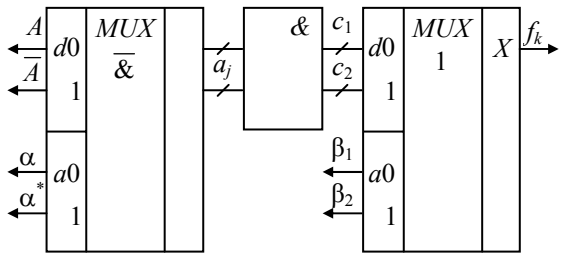


Рис. 4.3 Схема замещения

По правилам НДФ алгебры Буля единичная функция $F(1) = a$ мультиплексора НЕ состоит из суммы произведения минтермов:

$$a = \overline{\alpha}\,\overline{\alpha^*} + \alpha\overline{\alpha^*}A + \overline{\alpha}\alpha^*\overline{A}\,. \tag{4.1}$$

Используя аксиому дизъюнкции ($A + \overline{A} = 1$), создадим четное число слагаемых для минимизации структурной формулы (4.1) при объединении подобных членов

$$a = \overline{\alpha}\,\overline{\alpha^*}\left(A + \overline{A}\right) + \alpha\overline{\alpha^*}A + \overline{\alpha}\alpha^*\overline{A}\,,$$

что соответствует

$$a = \overline{\alpha}\,\overline{A}\left(\overline{\alpha^*} + \alpha^*\right) + \overline{\alpha^*}A\left(\overline{\alpha} + \alpha\right).$$

Применив аксиому дизъюнкции, находим формулу $a(1)$ в НДФ

$$a(1) = \overline{\alpha^*}A + \overline{\alpha}\,\overline{A}\,. \tag{4.2}$$

Для определения функции $a(0)$ в НКФ применим дважды теорему Деморгана

$$a(0) = \overline{a(1)} = \overline{(\alpha^* + \overline{A})\,(\alpha + A)}$$

или после перемножения

$$a(0) = \overline{\alpha^*\alpha + \alpha^*A + \alpha\overline{A} + A\overline{A}}\,.$$

Учитывая аксиому конъюнкции ($\alpha\alpha^* = 0$ и $A\overline{A} = 0$) и после повторного преобразования по Деморгану, получим выражение

$$a(0) = (\overline{\alpha^*} + \overline{A})\,(\overline{\alpha} + A)\,. \tag{4.3}$$

Структурные формулы (4.2) и (4.3) в векторной форме $1F(\Phi)$ выразим в матричной форме для ij-го мультиплексора:

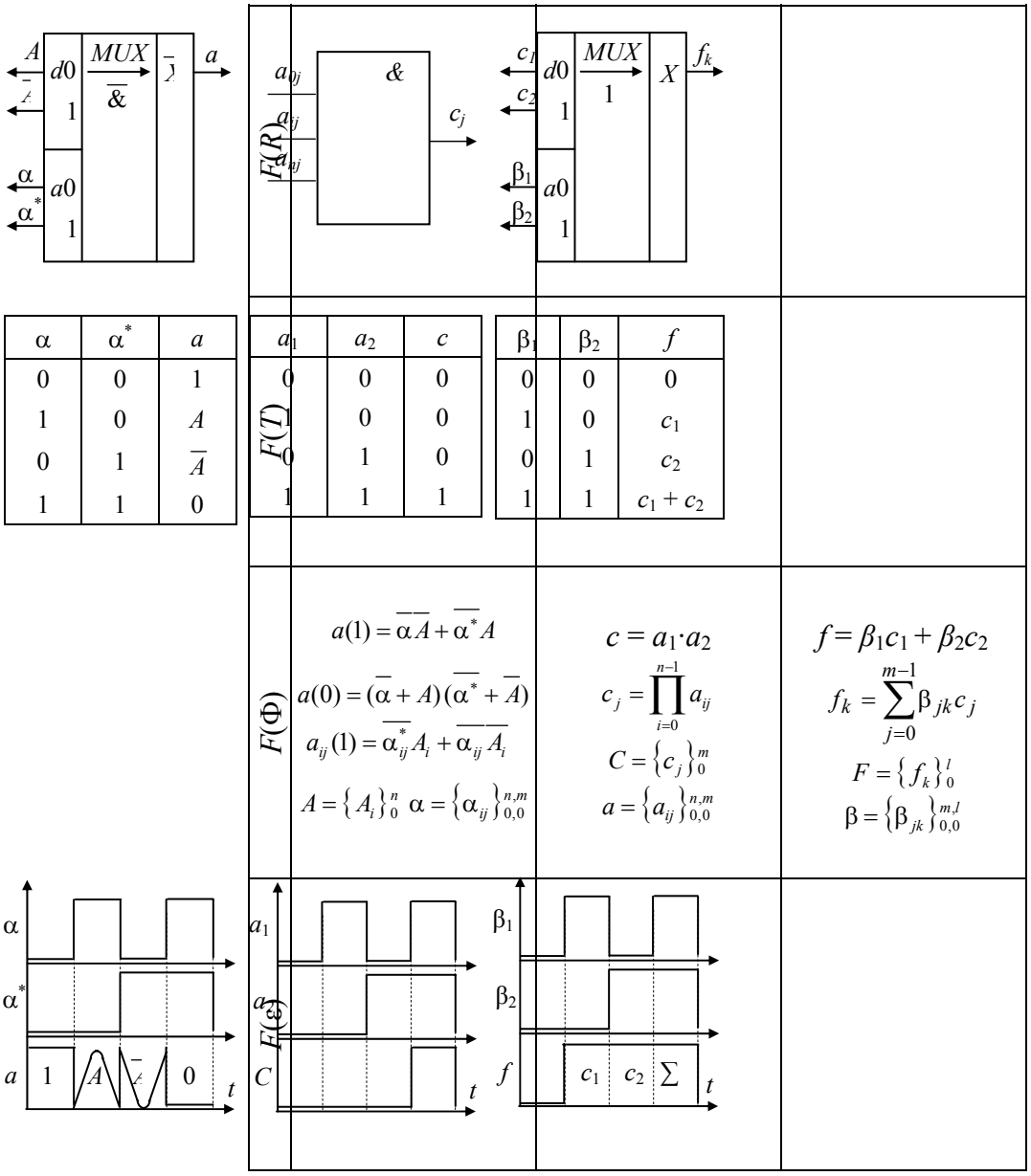
$$\begin{cases} a_{ij}(1) = \overline{\alpha_{ij}^*}A_i + \overline{\alpha_{ij}}\,\overline{A_i}\,; \\ a_{ij}(0) = \left(\overline{\alpha_{ij}} + A_i\right)\left(\overline{\alpha_{ij}^*} + \overline{A_i}\right). \end{cases} \tag{4.4}$$

Для двух ячеек a_1 и a_2 j-го столбца справедлива таблица $2F(T)$ конъюнкции, для которой истинное $C = 1$, если И a_1 И a_2 истинны $a_1 = a_2 = 1$. По таблице $2F(T)$ двух переменных несложно в НДФ записать структурную формулу

$$C = a_1\,a_2. \tag{4.5}$$

4.3 Логическое устройство

	1 НЕ	2 И	3 ИЛИ



α	α*	a
0	0	1
1	0	A
0	1	$\bar{A}$
1	1	0

a_1	a_2	c
0	0	0
1	0	0
0	1	0
1	1	1

β_1	β_2	f
0	0	0
1	0	c_1
0	1	c_2
1	1	$c_1 + c_2$

По методу индукции, заменяя двухвходовой конъюнктор функции (4.5) на многовходовой конъюнктор $2F(R)$, запишем в матричной форме логическое умножение $2F(\Phi)$ i-х мультиплексоров НЕ j-го столбца

$$c_j = \prod_{i=0}^{n-1} a_{ij} \,. \tag{4.6}$$

После подстановки в выражение (4.6) зависимостей (4.4) находим формулы j-го столбца конъюнктора $2F(R)$ в НДФ и НКФ:

$$\begin{cases} c_j(1) = \prod\limits_{i=0}^{n-1} \left(\overline{\alpha^*_{ij}} A_i + \overline{\alpha_{ij}\, \bar{A}_i} \right); \\ c_j(0) = \prod\limits_{i=0}^{n-1} \left(\overline{\alpha_{ij}} + A_i \right) \left(\overline{\alpha^*_{ij}} + \bar{A}_i \right). \end{cases} \tag{4.7}$$

Мультиплексор ИЛИ $3F(R)$ для двух переменных c_1 и c_2 k-й строки суммирующей матрицы управляется двумя адресами β_1 и β_2. Его таблица истинности $3F(T)$ обнулена при отключенных адресах $\beta_1 = \beta_2 = 0$, а при их активизации ($\beta_j = 1$) пропускает на выход f сигнал c_j по адресам $\{\beta_1, \beta_2\}$ равным $\{1, 0\}$ и $\{0, 1\}$. Если активизированы $\beta_1 = \beta_2 = 1$, то по адресу $\{1, 1\}$ на выходе f мультиплексора ИЛИ появляется суммарный сигнал $f_3 = c_1 + c_2$. Используя метод структурных формул, синтезируем в НДФ по таблице $3F(T)$ функцию мультиплексора $3F(R)$ для двух переменных:

$$f = \beta_1\bar{\beta}_2 c_1 + \bar{\beta}_1\beta_2 c_2 + \beta_1\beta_2(c_1 + c_2).$$

Объединяя подобные слагаемые

$$f = \beta_1 c_1(\bar{\beta}_2 + \beta_2) + \beta_2 c_2(\bar{\beta}_1 + \beta_1),$$

с учетом аксиомы дизъюнкции, находим решение для двухвходового коммутатора

$$f = \beta_1 c_1 + \beta_2 c_2. \tag{4.8}$$

При замене двух- на многовходовый мультиплексор получим по методу индукции формулу 3*F*(Ф) *m*-мерного сумматора:

$$f_k = \sum_{j=0}^{m-1} \beta_{jk} c_j\ . \tag{4.9}$$

Подставляя в формулу (4.9) выражения (4.7), запишем в матричной форме математическую модель схемы замещения ПЛМ (см. рис. 4.3), адекватную структуре логического устройства:

$$\begin{cases} f_k(1) = \sum_{j=0}^{m-1} \beta_{jk} \prod_{i=0}^{n-1} \left(\overline{\alpha^*_{ij}} A_i + \overline{\alpha_{ij}}\,\overline{A_i} \right); \\ f_k(0) = \sum_{j=0}^{m-1} \beta_{jk} \prod_{i=0}^{n-1} \left(\overline{\alpha_{ij}} + A_i \right) \left(\overline{\alpha^*_{ij}} + \overline{A_i} \right), \end{cases} \tag{4.10}$$

где код операции $N = \{\alpha_{ij}, \alpha^*_{ij}, \beta_{jk}\}$ программирования *ij*-го коммутатора матриц умножения по адресам α_{ij} – прямого сигнала A_i, соответственно α^*_{ij} – инверсного $\bar{A}_i$, а также управления *jk*-ми ключами матрицы ИЛИ за счет активизации β_{jk}-х адресов выходной функции $F = \{f_k\}_0^l$. Примеры реализации различных алгоритмов по модели (4.10) приведены в книге [13].

Анализ математических моделей (4.10) показывает их универсальность в $n \times m \times l$-мерном адресном пространстве, что инициирует гибкую архитектуру логического устройства с избыточной адресацией ассоциативных связей матричной структуры.

4.2.3 Адресация

Рассмотрим основные способы адресации [11, 13, 46, 49], реализуемые в микропроцессоре посредством регистра кода операции (РКОП) по коду команды. Код команды N_k определяет алгоритм функционирования блоков микропроцессора за счет программного управления ими в координатах пространство-время-функция. Для этого код команды N_k дифференцирован на код операции N_o, определяющий алгоритм микропрограммы БУ для синхронизации компонент микропроцессора, адрес *A* операнда, указывающий блок хранения и ячейку памяти для выборки информации (адреса или данных), и операнд в виде адреса или данных *D*, предназначенных для обработки, т.е. $N_k = \{N_0, A, D\}$.

Для работы с регистрами микропроцессора достаточна однобайтная команда, размещаемая в РКОП. По внутренней шине код операции N_o выбирает микропрограмму из БУ синхронизации блоков по адресу *A* команды для обработки данных *D*. Сервисные функции (обращение к портам) требуют двухбайтных команд, где первый байт регламентирует N_o регистра КОП, а адрес *A* или данные *D* размещают в регистрах или счетчике команд. При работе с интерфейсами памяти и ввода-вывода используют трехбайтные команды для адресации в них операндов по N_o регистра кода операции. Основой программирования служит рациональная техника адресации кода команды N_k, реализуемая кодом операции N_o РКОП.

Техника адресации [12, 49] включает три основных метода программного управления данными *D*, адресами *A* и функциями *F*(*A*), систематизированных в табл. 4.4 на способы адресации: непосредственной и неявной (1*D*), прямой и косвенной (2*A*), относительной и индексной (3*F*). Методы и способы

классифицированы, с методической точки зрения, от простой к сложной технике адресации по аналогии с правилами детской игры «Зарница» для школьников младшего, среднего и старшего звена обучения.

Для младших школьников элементарные правила игры требуют вознаграждения инициативы ребят при непосредственной или неявной подсказке правильного результата. По аналогии метод управления данными делят на непосредственную и неявную адресацию.

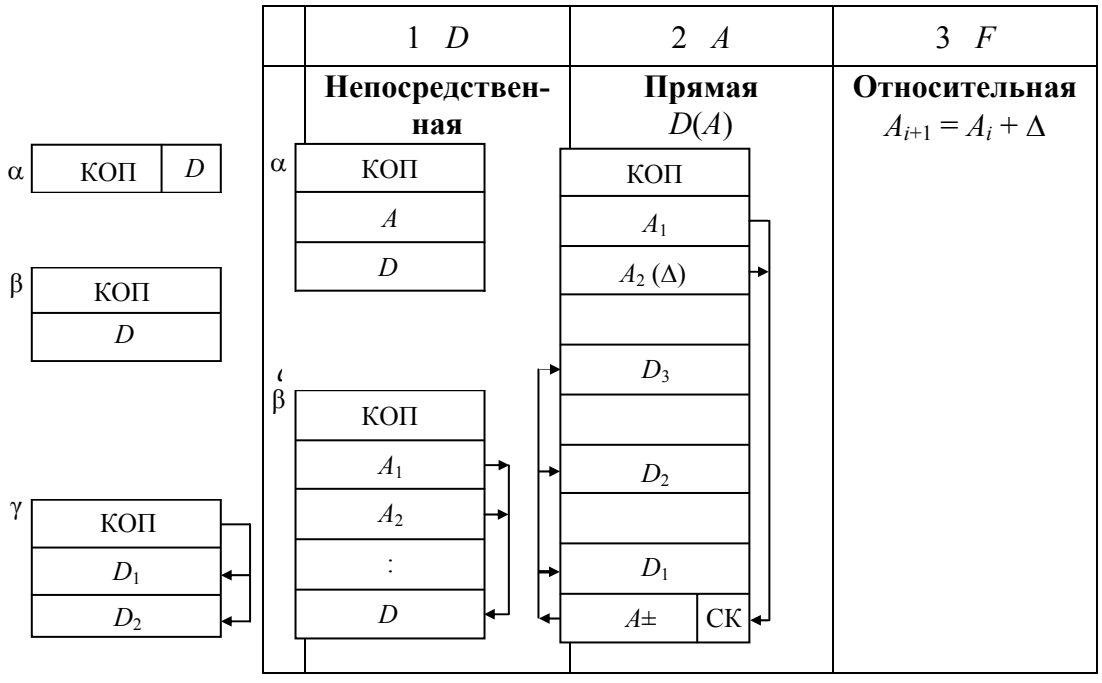
Непосредственная адресация [11, 13, 49] – это способ, в котором операнд D находится в теле команды N_k, после кода операции N_o (см. 1*D*, *a*). Она служит для логических и арифметических исчислений над константами при формировании однобайтных (α), двухбайтных (β) и трехбайтных (γ) команд при последовательном размещении байтов кода операции КОП и данных D. В этом способе операнд D следует непосредственно за кодом операции N_o и обрабатывается по микропрограмме БУ. Разновидностью непосредственной адресации является неявная.

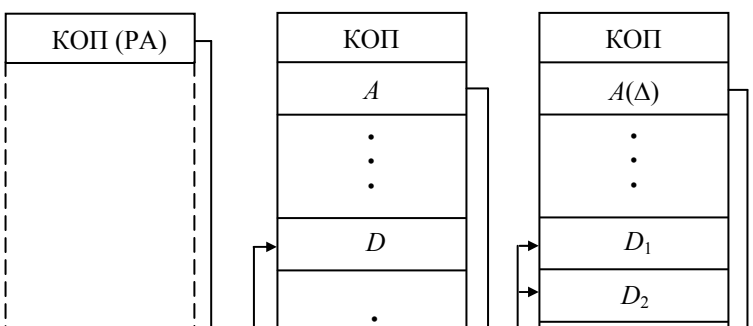
Неявная адресация [49, 55] – это способ, когда адрес операнда подразумевается КОП команды (см. 1*D*, *b*). Это обусловлено обработкой операндов регистра аккумулятора (РА), в котором хранится результат предыдущих вычислений. Неявная адресация удобна при логических операциях или обработке двух переменных, когда в РА содержится первый операнд. В данном способе код операнда с неявным адресом КОП (РА) размещен в первом байте РКОП, а операнд D хранится в РА, в заданных априори координатах адресного пространства (см. 1*D*, *b*). При непосредственной и неявной адресации операнд D является функцией кодов команды $D(N_k)$ и операции $D(N_o)$, которая характерна для элементарных вычислений.

Правила «Зарницы» для школьников среднего звена предполагают поиск клада при ориентации на местности по адресам пространственных координат. Игру усложняют последовательной адресацией, указывающей следующий признак ориентации клада. Аналогично метод управления по адресам систематизируют по их иерархии вложения на прямую и косвенную адресацию.

Прямая адресация [12, 55] – это способ, в котором адрес A операнда $D(A)$ находится в теле команды N_k после кода N_o операции КОП. Команда при этом (см. 2*A*, *a*) содержит последовательное включение кода операции КОП, адреса A операнда и данные D (см. там же, α)

4.4 Адресация





	Неявная $D(N_0)$	**Косвенная** $D[A_{12}(A)]$	**Индексная** $A_{i+1} = A + i\Delta$
b			

или за КОП старший A_1 и младший A_2 адреса, которые указывают знакоместо операнда D (см. β). Адресом операнда может служить ячейка памяти, номер порта, имя регистра. Прямая адресация служит для обработки переменных и результатов их вычислений, конкретные значения которых заранее неизвестны. Косвенная адресация развивает возможности техники адресации от прямого способа к относительному.

Косвенная адресация [13, 49, 55] – способ адресации, указывающий в команде адрес операнда $D[A_{12}(A)]$ (см. 2*A*, *b*). В этом случае код команды N_k содержит адрес памяти или регистра A, включающий значения адресов $\{A_1, A_2\} = A_{12}$ хранения операнда D. Способ использует фактически два адреса. Первый адрес A включен в команду как символическое имя регистра, указывающего не операнд, а адрес его знакоместа. В команде фиксируется старший адрес A_1, так как в КОП уже регламентировано использование пары регистров. Способ косвенной адресации предполагает многомерное вложение адресов, что удобно для выполнения подобных циклических операций, которые совершенствуются при алгоритмизации адресного исчисления при относительной и индексной адресации.

«Зарница» старшеклассников ориентирована на развитие алгоритмического мышления по логическим правилам счисления и исчисления. Ассоциативное мышление инициируют кодированные сообщения, которые в совокупности с алгоритмическими правилами указывают целенаправленную последовательность действий для решения поставленной задачи. По аналогии функциональные методы программного управления дифференцируют на относительную и индексную адресацию.

Относительная адресация [11, 16, 22, 49] формирует действительный адрес операнда из содержимого A счетчика команд (СК) и смещения Δ по младшему адресу A_2 кода команды при использовании ассоциативных образов и логических операторов. Относительный способ может использоваться для организации условного перехода. Часто адрес смещения $\Delta = A_2$ второго байта команды задают дополнительным кодом, что позволяет в программе осуществлять постраничный переход назад «–» и вперед «+» за счет де- и инкрементации адреса A счетчика команд, ячейки текущей страницы памяти или содержимого одного из регистров микропроцессора. Простейшей относительной адресацией является формирование $(i + 1)$-го адреса A_{i+1} при сложении (вычитании) i-го значения A_i со смещением Δ по алгоритму $A_{i+1} = A_i \pm \Delta$ (см. табл. 4.4, 3*F*, *a*). Сформированный адрес A_{i+1} изменяется в диапазоне от $+127_{10}$ до -128_{10}. Как видно, относительная адресация – это сводное название ряда способов, имеющих один общий принцип. Разновидностью способа относительной адресации является индексная.

Индексная адресация [11, 49] – это способ определения действительного адреса путем сложения содержимого A специального индексного регистра (ИР) с адресом $A(\Delta)$, следующим за кодом N_o операции (см. там же, 3*F*, *b*). Индексный способ удобен тогда, когда одна и та же последовательность вычислений должна быть выполнена для различных наборов данных, размещенных в последовательных ячейках памяти. Применение индексной адресации, в отличие от прямого способа, требует меньше времени при разработке программы и занимает меньше места в памяти.

Таким образом, рассмотрена техника адресации регистра кода операции по развитию методов программного управления данными, адресами и функциями, систематизированными в способы непосредственной и неявной 1*D*, прямой и косвенной 2*A*, относительной и индексной 3*F* адресации. Способы адресации определяют формат команды и код регистра кода операции, регламентирующие время программирования и оперативность вычисления, объем памяти и технологичность адресации для миними-

зации интеллектуальных, материальных и экономических затрат. Метод аналогии представления способов адресации от простого к сложному позволяет методически грамотно оценить эффективность программного управления компонентами микропроцессора для создания оптимального программного обеспечения и информационной технологии проектирования микропроцессорных средств.

4.2.4 Регистр признаков

Регистр, контролирующий состояния логического устройства по эквивалентным признакам для нормального функционирования, коррекции и защиты обработки информации за счет ветвления программы по условиям сравнения результатов, называют [11, 13, 49] регистром признаков (РП). В отличие от регистра кода операции, организующего ствол программы, РП выявляет штатные и нештатные состояния ЛУ по результатам регистра аккумулятора РА для выбора оптимальной подпрограммы посредством условных и безусловных переходов. Следовательно, РП выбирает тактику достижения цели, а стратегию алгоритма получения решения по программе регламентирует РКОП.

РП направляет выполнение какой-либо операции в зависимости от результатов предшествующих вычислений в процессе их сравнения по эквивалентным состояниям, называемым признаками [11]. Существует множество признаков, которые зависят от различных структур и разнообразия программ архитектуры микропроцессоров [13, 49]. Но общность конфигурации вычислений ЛУ и его соединений с триггерами РП, которые устанавливаются в единицу или сбрасываются в нуль в зависимости от вычислений, позволяет систематизировать основные признаки. Триггерные ячейки РП (флаги) хранят какой-то один признак на фиксированном знакоместе в адресном пространстве признаков, включающем признаки поразрядного и вспомогательного переноса, переполнения и знака мантиссы, нуля обработки и четности кодов. Признаки могут быть дифференцированы в зависимости от конфигурации архитектуры микропроцессоров, например, по тождественности обнуления (больше или меньше, равно или неравно) и правилам переноса (по тетрадам или декадам, по байтам или страницам адресации) и т.д.

В табл. 4.5 систематизированы основные признаки по строкам, дифференцированные в столбцах по формам представления логических функций в мнемотехнике $F(T)$, математике $F(\Phi)$ и метрологии $F(\varepsilon)$. Мнемоника признаков сведена в таблицы истинности сравнения результатов $F(T)$ во временных T координатах, отображение математических образов $F(\Phi)$ представлено на уровне моделей, алгоритмов и структурных формул, а также проиллюстрированы цифровые примеры логических преобразований $F(\varepsilon)$ для оценки эффективности алгоритмов тождественности состояний результатов обработки, как в десятичном N_{10} и двоичном N_2, так и в двоично-десятичном $N_{2/10}$ кодах. Для методически грамотного понимания алгоритмов сравнения, признаки классифицированы по их последовательности и преемственности применения, по вектору их развития от простого к сложному.

Признак переноса (см. табл. 4.5, 1) поразрядно сравнивает два операнда $A = \{a_i\}_0^n$ и $B = \{b_i\}_0^n$ в процессе логического сложения термов i-х значений a_i, b_i для выявления необходимости переноса единицы в следующий по старшинству $(i + 1)$-й разряд c_{i+1}. Алгоритм оценки тождественности имеет следующий вид:

$$\text{если} \quad a_i \begin{Bmatrix} = \\ \neq \end{Bmatrix} b_i , \quad \text{то} \quad c_i = \begin{cases} 0 \\ 1 \end{cases},$$

который реализует арифметическое сложение, систематизированное в таблице истинности $1F(T)$ (табл. 4.5). Признак переноса p_i соответствует логической единице тогда и только тогда, когда И a_i И b_i равны единице $a_i = b_i = 1$ согласно функции $p_i = a_i\, b_i$ конъюнкции $1F(\Phi)$, а результат получают по структурной формуле

$$c_i = a_i \overline{b_i} + \overline{a_i} b_i .$$

Поразрядный перенос иллюстрирует пример $1F(\varepsilon)$ сложения чисел 7_{10} и 5_{10} с результатом 12_{10} в десятичном коде $7 + 5 = 12$ и их двоичных аналогов $A = \{a_3, a_2, a_1, a_0\} = 0111_2$ и $B = \{b_3, b_2, b_1, b_0\} = 0101_2$. При сложении нулевых разрядов $\{a_0, b_0\} = \{1, 1\}$ результат равен нулю $c_0 = 0$, так как $c_0 = 1 \cdot \overline{1} + \overline{1} \cdot 1$, но формируется признак переноса $p_0 = 1 \cdot 1 = 1$ для суммирования в первом разряде. При этом $c_1 = 0$ из-за

	$F(T)$	$F(\Phi)$	$F(\varepsilon)$
1 Перено-	a_i b_i c_i p_i 0 0 0 0 1 0 1 0 0 1 1 0 1 1 0 1	a_i b_i c_i 0111 $-a_i$ + 0101 $-b_i$ $-\bar{a}_i b_i$ 1100 $-c_i$	$7 + 5 = 12$
2 В. переноса	a_3 b_3 c_3 p_2 0 0 0 0 1 0 1 0 0 1 1 0 1 1 0 1	$c_3 = a_3 b_3 + \bar{a}_3 b_3$ 0001 1001 + 0000 0111 p_2 0001 0000 + 0 0 0 1 1 0 1 0 0010 0110	$19 + 7 = 26$
3 Переполне-	a_n b_n c_n $p_п$ 0 0 0 0 1 0 1 0 0 1 1 0 1 1 0 1	$c_n = a_n b_n + \bar{a}_n b_n$ 0 p_n 1001 $a_n b_n$ – 0 p_n 1001 $\begin{cases}0\\1\end{cases}$, $C = \begin{cases}A+B\\EE\end{cases}$	$9 + 9 = 18 > C_m$
4 Знака	A $\bar{B}+1$ c $p_з$ 0 0 0 0 1 0 1 0 0 1 1 1 1 1 0 0	$C = A - B =$ $A + (\bar{B} + 1)$ 0 1001 $\bar{}$ $p_з = \begin{cases}0\\1\end{cases}$, $\lvert C\rvert = \begin{cases}+C\\-C\end{cases}$	$9 - 7 = 2$
5 Нуля	A $\bar{A}+1$ c p_0 0 0 0 1 1 0 A 0 0 1 $\bar{A}+1$ 0 1 1 0 1	$C = A - A =$ $A + (\bar{A} + 1)$ 0 1001 $\bar{}$ $C \begin{cases}\neq\\=\end{cases} 0$, $p_0 = \begin{cases}0\\1\end{cases}$	$9 - 9 = 0$
6 Четности	A $\bar{A}$ C $p_ч$ 0 0 0 0 1 0 $\bar{A}$ 1 0 1 A 1 1 1 1 0	$C = A + \bar{A} = 1$ 0 $p_ч$ 0 1 $A + \bar{A} = 1$ $p_ч = \begin{cases}1\\0\end{cases}$, $C = \begin{cases}\text{чет}\\\text{нечет}\end{cases}$	$7 + \bar{7} = 1$

сложения двух логических единиц $a_1 = p_0 = 1$ и также появляется признак переноса $p_1 = 1$ в следующий по старшинству разряд. Во втором разряде результат соответствует единице $c_2 = 1$ из-за суммирования трех значений $a_2 = b_2 = p_1 = 1$ с генерацией признака поразрядного переноса $p_2 = 1$. Результат вычисления третьего разряда также равен единице $c_3 = 1$ за счет сложения нулевых значений $a_3 = b_3 = 0$ с логической единицей переноса $p_2 = 1$, а в итоге вычислений формируется число $C = \{c_3, c_2, c_1, c_0\}$ в двоичном коде 1100_2. Результаты суммирования тождественны $N_{10} = N_2 = 12$, так как $N_2 = 1 \cdot 2^3 + 1 \cdot 2^2 + 0 \cdot 2^1 + 0 \cdot 2^0 = 12$.

Признак вспомогательного переноса (табл. 4.5, 2) устанавливается в единицу при переполнении тетрад, когда их старшие разряды равны логической единице $a_3 = b_3 = 1$ при исчислениях в двоично-десятичном коде $N_{2/10}$. Признак переполнения тетрады инициирует коррекцию значений младшей декады на шесть единиц из-за различия оснований шестнадцатиричного N_{16} и десятичного N_{10} кодов: $(F - A)_{16} = 6$. Коррекцию двоично-десятичного кода осуществляют по алгоритму $2F(\Phi)$:

$$\text{если} \quad p_2 = \begin{cases}0\\1\end{cases}, \quad \text{то} \quad C = \begin{cases}A+B\\A+B+6\end{cases},$$

правила которого систематизированы в таблице $2F(T)$ истинности для сравнения старших разрядов тетрад, аналогичной таблице $1F(T)$ тождественности i-х разрядов. По аналогии синтезированы структурные формулы $2F(\Phi)$ оценки старшего разряда c_3 и признака p_2 вспомогательного переноса

$$\begin{cases} c_3 = a_3\overline{b_3} + \overline{a_3}b_3; \\ p_2 = a_3 b_3. \end{cases}$$

Пример приведен для сложения операндов A и B в $N_{2/10}$, представленных нулевой и первой декадами $A = \{A_1, A_0\}$ и $B = \{B_1, B_0\}$, составленными из двоичных разрядов $A_i = \{a_{3i}, a_{2i}, a_{1i}, a_{0i}\}$ и $B_i = \{b_{3i}, b_{2i}, b_{1i}, b_{0i}\}$ (см. табл. 4.5, $2F(\varepsilon)$). Проиллюстрировано суммирование чисел $A(19) = 0001\ 1001$ и $B(7) = 0000\ 0111$. Признак переноса $p_2 = 1$ формируется при переполнении разряда c_3 нулевой декады, обусловленного признаками поразрядного переноса $p_i = 1$, $i = \overline{0, 3}$. При этом к значению $c_1(1) = 0001$ старшей декады добавляется значение признака $p_2 = 1$, а младшая декада $c_0(0) = 0000$ увеличивается на коррекцию $\Delta(6)$, за счет чего число $C(20) = 0010\ 0000$ преобразуется в искомое решение $C(26) = 0010\ 0110$. Тождественность результатов сложения в десятичном N_{10} и двоично-десятичном $N_{2/10}$ кодах подтверждает правильность вычислений за счет коррекции решения по признаку вспомогательного переноса. Разновидностью признаков переноса является переполнение разрядной сетки вычисления.

Признак переполнения (табл. 4.5, 3) вырабатывает сигнал $p_{\text{п}} = 1$ расширения длины слова операндов, а при невозможности этого – останова вычислений для защиты интерфейсов персонального компьютера. Признак переполнения аналогичен признакам переноса, но в отличие от них располагается в старшем разряде, регламентирующем длину слова. Алгоритм признака переполнения $p_{\text{п}}$ регламентирует условие $3F(\Phi)$ защиты вычислительной памяти компьютера:

$$\text{если} \quad p_{\text{п}} = \begin{cases} 0 \\ 1, \end{cases} \quad \text{то} \quad C = \begin{cases} A + B \\ EE \end{cases},$$

где код EE оповещает о превышении мощности вычислителя и переводит его в режим ожидания для функционирования в диалоге с пользователем. Признак переполнения $p_{\text{п}}$ размещают в старшем разряде $\{a_{n+1}, b_{n+1}, c_{n+1}\}$ операндов A, B, C, например, в четвертом для защиты тетрады (см. табл. 4.5, $3F(\varepsilon)$). Он соответствует нулю $p_{\text{п}} = 0$ при задании необходимой длины слова и равен единице $p_{\text{п}} = 1$, если старшие разряды слагаемых И a_n И b_n тождественны $a_n = b_n = 1$, как показано в таблице истинности $3F(T)$. Математический образ представляет система структурных формул для определения старшего c_n разряда вычислений и состояния признака $p_{\text{п}}$ переполнения:

$$\begin{cases} c_n = a_n\overline{b_n} + \overline{a_n}b_n; \\ p_{\text{п}} = a_n b_n, \end{cases}$$

которые иллюстрируют численный пример $3F(\varepsilon)$ контроля четырехразрядной мантиссы. Для наглядности длина слова ограничена мощностью тетрады и нормирована ее максимальным значением $C_m = 2^4 = 16$. Операнды слагаемых A и B выбраны одинаковыми $A = B = 9$ при их суммарном значении $C = 18$ больше заданной нормы $C > C_m$. Длина слов слагаемых A и B не превышает нормированную величину $C_m > A = B = 9$, поэтому их признак $p_{\text{п}} = 0$, а операнды представлены равными кодами 0 1001. Результат их суммирования $C(18)$ превышает заданную тетрадой норму $C_m(16) = 1\ 0000$, что индицируется признаком переполнения $p_{\text{п}} = 1$ в полученном решении $C(18) = 1\ 0010$ больше нормы C_m.

Развитием признаков переноса и переполнения служат ассоциативные условия знака, нуля и четности. В отличие от простых условий оценки скалярных величин конкретных разрядов, сложные признаки контролируют массивы данных в векторной форме.

Признак знака (табл. 4.5, 4) определяет полярность модуля операнда для представления числа в прямом или инверсном кодах при выполнении операций сложения и вычитания. Отрицательные числа представляются в микропроцессоре дополнительным кодом, а положительные – прямым, что обусловлено их суммой, равной нулю. Например, числу 7, представленному прямым кодом $N_2(7) = 0\ 0111$ соответствует дополнительный код $N_{\text{д}}(7) = 1\ 1001$, так как проводя вычитание из нуля $N_2(0) = 0\ 0000$ обычным способом, находим

$$\begin{array}{rll} & 0\ 0000 & N_2(0) \\ - & 0\ 0111 & N_2(7) \\ \hline & 1\ 1001 & N_{\text{д}}(7) \end{array}$$

.

Дополнительный код $N_д$ от прямого N_2 отличается суммой инверсного кода $\overline{N}_2$ и единицы:

$$N_д = \overline{N}_2 + 1.$$

Действительно, для числа $N_2(7) = 0\ 0111$ находим его дополнение $N_д(7)$

$$\begin{array}{ll} 0\ 0111 & N_2\,(7) \\ 1\ 1000 & \overline{N}_2\,(7) \\ 1\ 1001 & N_д\,(7) \end{array},$$

что соответствует стандартному способу вычитания. Поэтому в унитарном логическом пространстве положительное и отрицательное числа отличаются знаком старшего разряда 0 и 1, соответствующим « + » и «–» обычной арифметики.

Признак знака $p_з$ представлен алгоритмом $4F$(Ф):

$$\text{если} \quad p_з = \begin{cases} 0 \\ 1, \end{cases} \quad \text{то} \quad |C| = \begin{cases} +C \\ -C. \end{cases}$$

C является положительным числом, если признак знака $p_з = 0$, в противном случае отрицательным для $p_з = 1$. Из нулевой суммы прямого и дополнительного кодов $N_2 + N_д = 0$ следует правило вычитания в бинарном поле:

$$C = A - B = A + \left(\overline{B} + 1\right),$$

где отрицательному числу вычитаемого B соответствует его дополнительный код $N_д(B) = \overline{B} + 1$. Правила сложения прямого кода A и дополнительного $\overline{B} + 1$ систематизированы в векторной таблице логического вычитания $4F(T)$, аналогичной таблицам сравнения признаков переноса $1F(T) - 3F(T)$, с отличительной особенностью признака знака $p_з = \overline{A}(\overline{B} + 1) = 1$. В НДФ таблице $4F(T)$ соответствует система формул для результата C и признака $p_з$

$$\begin{cases} C = A\left(\overline{\overline{B} + 1}\right) + \overline{A}\left(\overline{B} + 1\right); \\ p_з = \overline{A}\left(\overline{B} + 1\right). \end{cases}$$

Признак знака иллюстрирует пример $4F(\varepsilon)$ вычитания $9 - 7 = 2$, соответствующий сложению прямого кода $A_2(9) = 0\ 1001$ с признаком $p_з = 0$ и дополнительного $B_д(7) = 1\ 1001$ с признаком $p_з = 1$, для которых получен результат $c_2(2) = 0\ 0010$ с положительным знаком ($p_з = 0$). Производным от признака знака является условие обнуления, также оперирующее с ассоциацией разрядов числа в векторной форме.

Признак нуля (табл. 4.5, 5) регистрирует обнуление матрицы логического устройства для ветвления программы по условному переходу или для организации режима ожидания. Обнуление создают параллельным или последовательным вычитанием из регистра адреса смещения Δ, соответственно равного операнду A, его части A/n или минимальному шагу $\Delta = 1$. В основу признака нуля p_0 положен алгоритм тождественности операндов $A = A$, для которого

$$\text{если} \quad C = \begin{cases} \neq \\ = \end{cases} 0, \quad \text{то} \quad p_0 = \begin{cases} 0 \\ 1. \end{cases}$$

Из алгоритма следуют модель вычитания $5F$(Ф)

$$C = A - A = A + \left(\overline{A} + 1\right) = 0$$

и векторная таблица истинности $5F(T)$, аналогичная мультиплексору и цифровому компаратору. Из таблицы следует система формул для синтеза структурных схем:

$$\begin{cases} C = A\left(\overline{\overline{A}+1}\right) + \overline{A}\left(\overline{A}+1\right); \\ p_0 = \overline{A}\left(\overline{\overline{A}+1}\right) + A\left(\overline{A}+1\right), \end{cases}$$

из которой после несложных преобразований следует математическая модель.

Признак обнуления иллюстрирует пример $5F(\varepsilon)$ вычитания из операнда $A(9) = 0\ 1001$ равного ему смещения $\Delta = A(9)$ в дополнительном коде $A_д(9) = \overline{A}(9) + 1$, т.е. $A_д(9) = 1\ 0111$. При их логическом сложении результат $C(0) = 0\ 0000$ равен нулю, а признак нуля $p_0 = 1$, устанавливается в единицу. Модификацией обнуления служит условие четности, при котором операнд сравнивается с его инверсией.

Признак четности (табл. 4.5, 6) контролирует достоверность преобразования операнда по его инверсному эквиваленту или четному числу единиц в разрядах. Признак четности $p_ч$ определяется алгоритмом

$$\text{если} \quad p_ч = \begin{cases} 1 \\ 0, \end{cases} \quad \text{то} \quad C = \begin{cases} \text{чет} \\ \text{нечет}, \end{cases}$$

где «чет» и «нечет» соответствуют четному и нечетному числу единиц в разрядах операнда C. Из алгоритма следуют математические модели тождественности единице операнда и признака $6F(Ф)$:

$$\begin{cases} C = A + \overline{A} = 1; \\ p_ч = A + \overline{A} = 1 \end{cases}$$

и правила, систематизированные в векторную таблицу $6F(T)$. Из таблицы $6F(T)$ синтезирована система структурных формул

$$\begin{cases} C = A + \overline{A} + A\overline{A}; \\ p_ч = A + \overline{A}, \end{cases}$$

которая адекватна математической модели четности.

Признак четности используют для контроля четности данных при их передаче, для этого за последним байтом (или тетрадой) посылают его инверсный код, контрольная сумма которых должна быть равна единице. Цифровой пример $6F(\varepsilon)$ показан при сложении тетрад операнда $A_2(7) = 0111$ с его инверсией $\overline{A}(7) = 1000$, сумма которых включает четное число единиц $c_2(1) = 1111$. Для приведенного примера операндам в прямом A_2 и инверсном отображении $\overline{A}_2$ с нечетным числом единиц 3 и 1 соответствует признак четности $p_ч = 0$, а результату c_2 из четного числа единиц 4 – однозначно значение единица $p_ч = 1$.

Таким образом, показано развитие признаков от простого условия переноса по разрядам до сложных оценок тождественности нуля и четности в векторной форме. С позиций преемственности и последовательности признаки проанализированы в основных формах представления мнемотехники $F(T)$, математических образов $F(Ф)$ и цифровых примеров их оценки $F(\varepsilon)$, что позволяет методически грамотно понять методы тождественности эквивалентам и алгоритмы регистра признаков. Анализ признаков и команд программного обеспечения компьютера иллюстрирует гибкость архитектуры микропроцессора за счет универсальной математической модели логического устройства из ассоциативной матрицы.

Выводы

1 Информатизация научно-технической революции развивает вычислители с жесткой структурой ВЖС и АВМ к гибкой архитектуре ЦВМ и ПК за счет интеграции аппаратных средств и программного обеспечения, математического обеспечения и метрологических средств в информационное обеспечение микропроцессорных средств.

2 Предложена классификация микропроцессоров по архитектурам, систематизированным в морфологическую таблицу оптимальных решений с согласованными компонентами информационного

обеспечения для организации информационной технологии проектирования микропроцессорных средств.

3 Показана эффективность информационной технологии при проектировании компонент микропроцессорных средств на примере обобщенной архитектуры микропроцессора, алгоритмов и моделей техники адресации и условных переходов регистров кода операции и признаков, а также синтеза универсальной математической модели логического устройства на проектируемой логической матрице с избыточной топологией БИС.

СПИСОК ЛИТЕРАТУРЫ

1 Алексенко А.Г. Основы микросхемотехники. М.: Сов. радио, 1977. 408 с.

2 Алексенко А.Г., Шагурин И.И. Микросхемотехника. М.: Радио и связь, 1990. 496 с.

3 Алексенко А.Г., Коломбет Е.А., Стародуб Г.И. Применение прецизионных аналоговых микросхем. М.:Радио и связь, 1985. 256 с.

4 Альтшуллер Г.С. Найти идею. Новосибирск: Наука, 1992. 200 с.

5 Балашов Е.П., Пузанков Д.В. Микропроцессоры и микропроцессорные системы. М.: Радио и связь, 1981. 328 с.

6 Бребрин В.Н. Программное обеспечение персональных ЭВМ. М.: Наука, 1988. 278 с.

7 Болгов В.В., Скрыль С.В., Алексеенко С.П. Основы микропроцессорной техники. Воронеж: ВШМВД, 1997. 96 с.

8 Бояринов А.Е., Глинкин Е.И., Кирьянов А.В. Схемотехника БИС. Микропроцессоры. Тамбов: Изд-во Тамб. гос. техн. ун-та, 1999. 52 с.

9 Вайда Ф., Чакань А. Микро-ЭВМ. М.: Энергия, 1980. 360 с.

10 Воскрокнутов Н.Г., Евтихиев Н.Н. Информационно-измерительная техника. М.: Высшая школа, 1977. 232 с.

11 Вуд А. Микропроцессоры в вопросах и ответах. М.: Энергоатомиздат, 1985. 184 с.

12 Герасимов Б.И., Глинкин Е.И. Микропроцессоры в приборостроении. М.: Машиностроение, 2000. 328 с.

13 Герасимов Б.И., Глинкин Е.И. Микропроцессорные аналитические приборы. М.: Машиностроение, 1989. 248 с.

14 Гилмор Ч. Введение в микропроцессорную технику. М.: Мир, 1984. 334 с.

15 Глинкин Е.И., Герасимов Б.И. Микропроцессорные средства. Тамбов: Изд-во Тамб. гос. техн. ун-та, 2005. 140 с.

16 Глинкин Е.И., Глинкин М.Е. Схемотехника МИС. Компьютерный электропривод. Тамбов: Изд-во Тамб. гос. техн. ун-та, 2002. 76 с.

17 Глинкин Е.И., Глинкин М.Е. Схемотехника БИС. Выпрямители и инверторы. Тамбов: Изд-во Тамб. гос. техн. ун-та, 1999. 72 с.

18 Глинкин Е.И., Кирьянов А.В., Бояринов А.Е. Схемотехника микропроцессорных измерительных средств. Тамбов: Изд-во Тамб. гос. техн. ун-та, 1998. 60 с.

19 Глинкин Е.И, Кирьянов А.В., Петров С.В. Схемотехника ИС. Тамбов: Изд-во Тамб. гос. техн. ун-та, 1999. 28 с.

20 Глинкин Е.И. Схемотехника АИС. Тамбов: Изд-во Тамб. гос. техн. ун-та, 2000. 120 с.

21 Глинкин Е.И. Схемотехника микропроцессорных систем. Тамбов: Изд-во Тамб. гос. техн. ун-та, 1998. 158 с.

22 Глинкин Е.И. Схемотехника АЦП. Тамбов: Изд-во Тамб. гос. техн. ун-та, 2001. 160 с.

23 Глинкин Е.И. Схемотехника БИС: автоматические интерфейсы ввода-вывода. Тамбов: Изд-во Тамб. гос. техн. ун-та, 1999. 72 с.

24 Глинкин Е.И. Схемотехника СИС. Тамбов: Изд-во Тамб. гос. техн. ун-та, 1999. 48 с.

25 Гольденберг Л.М. Импульсные устройства. М.: Радио и связь, 1981. 224 с.

26 Грановский В.А. Динамические измерения: Основы метрологического обеспечения. Л.: Энергоатомиздат, 1984. 224 с.

27 Гришин Ю.П., Казаринов Ю.М., Катиков В.М. Микропроцессоры в радиотехнических системах.

М.: Радио и связь, 1982. 280 с.

28 Гуртовцев А.А., Гудыменко С.В. Программы для микропроцессоров: Справ. пособие. М.: Высшая школа, 1989. С. 352.

29 Гусев В.В. Основы импульсной и цифровой техники. М.: Сов. радио, 1977. 440 с.

30 Гутников В.С. Интегральная электроника в измерительных устройствах. Л.: Энергоатомиздат, 1988. 304 с.

31 Дамке М. Операционные системы микроЭВМ. М.: Финансы и статистика, 1985. 150 с.

32 Ефимов И.Е., Козырь И.Я., Горбунов Ю.И. Микроэлектроника. М.: Высшая школа, 1987. 416 с.

33 Жеребцов И.П. Основы электроники. Л.: Энергоатомиздат, 1985. 352 с.

34 Зельдин Е.А. Цифровые интегральные микросхемы в информационно-измерительной аппаратуре. Л.: Энергоатомиздат, 1986. 280 с.

35 Каган Б.М. Электронные вычислительные машины и системы. М.: Энергия, 1979. 528 с.

36 Калабеков Б.А. Микропроцессоры и их применение в системах передачи и обработки сигналов. М.: Радио и связь, 1988. 368 с.

37 Калабеков Б.А., Мамзелев И.А. Цифровые устройства и микропроцессорные системы. М.: Радио и связь, 1987. 400 с.

38 Клингман Э. Проектирование специализированных микропроцессорных систем. М.: Мир, 1985. 363 с.

39 Кнут Д.Е. Искусство программирования для ЭВМ. Сортировка и поиск. М.: Мир, 1978. Т. 3. 843 с.

40 Кондаков Н.И. Логический словарь-справочник. М.: Наука, 1976. 720 с.

41 Косарев Ю.А., Виноградов С.В. Электрически изменяемые ПЗУ. Л.: Энергоатомиздат, 1985. 70 с.

42 Коффрон Д. Технические средства микропроцессорных систем. М.: Мир, 1983. 344 с.

43 Лебедев О.Н. Микросхемы памяти и их применение. М.: Радио и связь, 1990. 180 с.

44 Майоров С.А., Кириллов В.В., Приблуда А.А. Введение в микро-ЭВМ. Л.: Машиностроение, 1988. 304 с.

45 Мейзда Ф. Интегральные схемы. М.: Мир, 1981. 280 с.

46 Микропроцессоры. Архитектура и проектирование микроЭВМ: В 3 кн. Кн. 1 / Под ред. Л.Н. Преснухина. М.: Высшая школа, 1986. 495 с.

47 Микропроцессоры и микропроцессорные комплекты ИС: Справочник: В 2 т / Под ред. В.А. Шахнова. М.: Радио и связь, 1988. 368 с.

48 МикроЭВМ в информационно-измерительных системах / С.М. Переверткин и др. М.: Машиностроение, 1987. 248 с.

49 МикроЭВМ: Пер. с англ. / Под ред. А. Дирксена. М.: Энергоиздат, 1982. 328 с.

50 Микроэлектронная технология и ее влияние на общество: Сб. статей. Пер. с англ. М.: Знание, 1987. 160 с.

51 Мирский Г.Я. Микропроцессоры в измерительных приборах. М.: Радио и связь, 1984. 160 с.

52 Муромцев Ю.Л., Селиванова З.М., Чернышов В.А. Микропроцессорные системы контроля. Тамбов: Изд-во Тамб. гос. техн. ун-та, 2004. 96 с.

53 Муромцев Ю.Л., Орлова Л.П. Микропроцессорные системы энергосберегающего управления. Тамбов: Изд-во Тамб. гос. техн. ун-та, 2001. 80 с.

54 Прагер И.Л. Электронные аналоговые вычислительные машины. М.: Машиностроение, 1979. 231 с.

55 Программируемые микрокалькуляторы / Под ред. Я.К. Трохименко. М.: Радио и связь, 1990. 272 с.

56 Программное обеспечение микропроцессорных систем: Справочник. Киев: Техніка, 1989. 301 с.

57 Применение интегральных схем: В 2 кн. / Под ред. А. Уильямса. М.: Мир, 1987. 413 с.

58 Применение интегральных микросхем в электронно-вычислительной технике. М.: Радио и связь, 1987. 384 с.

59 Пухальский Г.И., Новосельцева Т.Я. Проектирование дискретных устройств на ИС: Справочник. М.: Радио и связь, 1990. 304 с.

60 Соботка З., Стары Я. Микропроцессорные системы. М.: Энергоиздат, 1981. 496 с.

61 Справочник по цифровой вычислительной технике (программное обеспечение) / Под ред. Б.Н. Малиновского. Киев: Техніка, 1981. 207 с.

62 Сташин В.В., Урусов А.В., Мологонцева О.Ф. Проектирование цифровых устройств на микроконтроллерах. М.: Энергоатомиздат, 1990. 224 с.

63 Титце У., Шенк К. Полупроводниковая схемотехника / Пер. с нем. М.: Мир, 1982. 512с.

64 Хвощ С.Т., Варлинский Н.Н., Попов Е.А. Микропроцессоры и микроЭВМ в САУ. Л.: Машиностроение, 1987. 640 с.

65 Хьюз Дж., Мичном Дж. Структурный подход к программированию: Пер. с англ. М.: Мир, 1980. 280 с.

66 Цытович Л.И., Маурер В.Г. Элементы и устройства систем управления тиристорными преобразователями. Челябинск: ЮУрГУ, 1998. 274 с.

67 Цытович Л.И. Элементы аналоговой и цифровой электроники в автоматизированном электроприводе. Челябинск: ЮУрГУ, 2001. 480 с.

68 Чернявский Е.А., Недосекин Д.Д., Алексеев В.В. Измерительно-вычислительные средства автоматизации производственных процессов. Л.: Энергоатомиздат, 1989. 272 с.

69 Шевкопляс Б.В. Микропроцессорные структуры: Справочник. М.: Радио и связь, 1990. 512 с.

70 Шилейко А.В., Шилейко Т.И. Микропроцессоры. М.: Радио и связь, 1986. 112 с.

71 Шило В.Л. Популярные цифровые микросхемы: Справочник. М.: Металлургия, 1988. 352 с.

72 Электроника: Прошлое, настоящее, будущее: Пер. с англ. / Под ред. В.И. Сифорова. М.: Мир, 1980. 196 с.