ОГЛАВЛЕНИЕ

ВВЕДЕНИЕ

Информатика и вычислительная техника – быстро развивающиеся дисциплины. Они выглядят молодыми, поскольку их современная история начинается с середины XX века, но исторические корни этих дисциплин уходят на глубину тысячелетий. Не все идеи прошлого нашли применение в современности, почти все они устарели, а большое их число оказалось тупиковыми. Но, во-первых, без их появления столь длинный путь развития никогда не был бы пройден. И, во-вторых, любая современная техника находится под влиянием традиций и идей прошлого, крепко сидящих в сознании разработчиков и пользователей. Для объективного понимания современного состояния информатики и вычислительной техникой важно представлять себе истоки современных технологий, чему и посвящен учебный курс.

Предлагаемое учебное пособие посвящено изложению исторических истоков и методологических основ информатики и вычислительной техники. В качестве вводной темы рассматривается методология научного познания, поскольку все дальнейшее изложение посвящено, по сути, истории научного познания в конкретных науках. Общие вопросы вычислительной техники и информатики, поделенные на «доэлектронную» и «электронную» эпохи, и охватывающие период от палеолита до ближайшего будущего, рассматриваются в хронологическом порядке в двух больших главах. В отдельные главы вынесено рассмотрение элементной базы, микропроцессоров, суперкомпьютеров и специализированных компьютеров, компьютерных сетей, операционных систем и языков программирования.

В основе пособия лежит курс лекций, прочитанных автором в 2007–2009 гг. студентам 5-го курса факультета Компьютерных технологий и управления СПбГУ ИТМО. В конце пособия приводится список литературы, использовавшийся при подготовке курса лекций, а также перечень литературы, рекомендуемой к самостоятельному изучению.

ГЛАВА 1. МЕТОДОЛОГИЯ НАУЧНОГО ПОЗНАНИЯ [1.1, 1.2]

Термины и определения.

История – процесс развития (от греч. *ιστορία* – рассказ о прошедшем).

Методология – учение о наиболее общих принципах, положениях и методах, составляющих основу той или иной науки.

Наука – форма деятельности людей, направленная на производство знаний о природе, обществе и о самом познании, имеющая непосредственной целью постижения истины и открытие объективных законов на основе обобщения реальных фактов в их взаимосвязи, для того чтобы предвидеть тенденции развития действительности и способствовать её изменению.

Знание – объективная реальность, данная в сознании человека, который в своей деятельности отражает, идеально воспроизводит объективные закономерные связи реального мира.

Познание – процесс приобретения и развития знания, его постоянное углубление, расширение, совершенствование и воспроизводство.

Вычислительная техника – область техники, объединяющая средства автоматизации математических вычислений и обработки информации, а также наука о принципах построения, действия и проектирования этих средств.

Информатика – отрасль науки (либо отрасль производства, либо прикладная дисциплина), изучающая структуру и общие свойства информации, а также вопросы, связанные с ее сбором, хранением, поиском, переработкой, преобразованием, распространением и использованием в различных сферах деятельности.

«Западным» аналогом информатики и вычислительной техники является дисциплина «компьютерные науки» (computer science), охватывающая широкий круг вопросов от теоретических исследований алгоритмов до практического внедрения аппаратных и программных продуктов.

§1.1. Философия науки

Прежде чем приступить к рассмотрению частных вопросов методологии информатики и вычислительной техники, рассмотрим методологию науки в общем. Методология науки, в традиционном понимании, – это учение о методах и процедурах научной деятельности. Задачей методологии является осмысление формализованного аппарата конкретных наук, изучение теоретических оснований науки и конкурирующих моделей роста научного знания. Первоначально проблемы методологии разрабатывались в рамках философии, и до сих

пор связаны с ней в рамках раздела общей теории познания (гносеологии), в особенности теории научного познания и философии науки.

Философия науки как дисциплина возникла в ответ на потребность осмыслить социокультурные функции науки в условиях научно-технической революции. Предметом философии науки являются общие закономерности научного познания как особой деятельности по производству научных знаний, взятых в их историческом развитии и в социокультурном контексте.

Современная философия науки пытается понять место науки в современной цивилизации в её многообразных отношениях к этике, политике, религии. Призывает обращать внимание на философский план любой проблемы, на отношение научной мысли к действительности во всей её полноте. Но центральной проблемой философии науки по-прежнему является проблема роста (развития) научного знания.

Философские категории

Философия оперирует наиболее обобщенными понятиями, или категориями, которые позволяют анализировать мир с точки зрения его всеобщих и необходимых свойств и качеств. Примерами философских категорий являются: материя, пространство, время, количество, качество, общее и частное, причина и следствие, случайность и закономерность, возможность и действительность, сущность и явление, содержание и форма и т.д.

Каждая наука имеет систему своих категорий, и в каждой науке они находятся во взаимосвязи. Например, случайность и закономерность имеют причину. Без ликвидации причины случайность переходит в закономерность, и если бы существовала только закономерность – то все было бы неизбежно. Философские категории являются самыми широкими и служат методологической основой научного познания во всех областях человеческой деятельности.

Научное познание и обыденно-практическое знание

Процесс получения объективного, истинного знания, направленного на отражение реальных закономерностей называется научным познанием. Задачами научного познания является описание, объяснение и предсказание процессов и явлений действительности.

На ранних этапах человеческой истории существовало обыденно-практическое знание, которое доставляет элементарные сведения о природе и окружающей среде. Оно включает в себя здравый смысл, приметы, назидания, рецепты, личный опыт, традиции, используется практически неосознанно и не требует предварительной системы доказательств. Также существовало мифологическое познание,

представляющее собой фантастическое отражение реальности, но позволяющее фиксировать и передавать опыт поколений.

Виды вненаучного знания

Познание не ограничено сферой науки, знание также существует и за её пределами. Помимо обыденно-практического выделим следующие формы вненаучного знания:

– паранаучное (от греч. para – около, возле, при). Обозначает многообразие сопутствующих науке учений, существующих за ее пределами, но связанных с нею определенной общностью проблематики или методологии. Паранаучные учения несовместимы с имеющимся гносеологическим стандартом, описывают реальные явления и могут содержать в себе как существенно ошибочные, так и истинные положения. Это могут быть как новые, не получившие подтверждения, так и устаревшие теории, а также «оккультные науки» (алхимия, астрология, хиромантия, толкование сновидений). Паранаучное знание способствует диалогу общества с другими социокультурными системами [1.3].

– лженаучное (или псевдонаучное, от греч. «псевдо», т.е. «ложный»). Обозначает деятельность, претендующую на научную или имитирующую её, но не соблюдающую научную методологию. Отличается широким использованием ошибочного и непроверенного знания, а также умелой «обработкой» фактов. Характеризуется домыслами и предрассудками, малограмотным пафосом, нетерпимостью и претенциозностью. Утопичное и сознательное искажение представлений о действительности часто называют антинаукой. К положительной стороне лженаучного знания можно отнести привлечение внимания к спорным положениям официальной науки.

Философия и наука

Наука существует как процесс выдвижения и опровержения гипотез, роль философии при этом заключается в исследовании критериев научности и рациональности. Вместе с тем, философия осмысливает научные открытия, включая их в контекст сформированного знания и, тем самым, определяя их значение. Отметим основные отличия философии и науки в контексте истории и методологии:

– Наука обычно не задается вопросом, как возникла её дисциплина, а философия стремится выяснить исходные предпосылки всякого знания

– В философии важен не только достигнутый результат, но и путь к этому результату.

– Научные дисциплины могут развиваться, не учитывая опыт

других форм общественного сознания, а за философией стоит весь опыт познания человечества.

– Наука не содержит внутри себя критериев социальной значимости своих результатов, в то время как первый философский тезис – «все связано со всем».

Методологические философские предпосылки науки [1.4]

Появившаяся в 1931 г. теорема Геделя утверждает, что всякая достаточно сильная формальная логическая теория содержит такие утверждения, которые нельзя ни доказать, ни опровергнуть внутренними средствами этой теории. Утверждение данной формальной теории, которое не может быть проверено внутри нее, вполне может быть проверено средствами более мощной логической теории, однако это далеко не всегда возможно. Кроме того, очень легко допустить ошибку в логическом выводе, предположении или эксперименте. Поэтому для научного мышления характерно полное отсутствие уверенности в своей непогрешимости, стремление к критической проверке выводов любой теории. Вместе с тем существуют положения, которые ученый принимает заранее как методологические философские предпосылки своей деятельности, или «догмы», поскольку они не доказаны и принимаются на веру:

- Объективность существования мира и закономерностей, которым этот мир подчиняется (логически крайний субъективизм неопровергаем)
- «Добывается» объективная истина о мире (надежность объективной истины определяется объективными критериями)
- Мир признается «логичным» (добываемые сведения могут быть уложены в логически стройную систему)

Виды мышления

В основе познания лежит мышление, или совокупность умственных процессов. Поскольку во многом достигаемый результат зависит от субъекта, познающего мир, то важен и учет особенностей мышления субъекта (т.е. человека). В философии при описании мышления также выделяются парные категории. Мышление может быть конкретным/абстрактным, гибким/жестким, определенным или неопределенным. Но больше принято выделять три следующих типа мышления:

Наглядно-образное – способность припоминать и осуществлять при помощи представления различные манипуляции в сознании.

Абстрактно-логическое – мышление абстракциями, т.е. категориями, которых нет в природе. Формируется в возрасте 4–5 лет. Считается, что у животных нет абстрактного мышления.

Наглядно-действенное – задачи решаются с помощью существующего, реального объекта.

Законы развития Гегеля

В XIX веке немецкий философ Гегель (1733–1799), исследуя законы сознания, сформулировал общие законы развития духа, которые впоследствии были распространены и на окружающую материальную действительность. Законы первыми указали на принципиально нелинейный характер развития сложных систем и предвосхитили современные концепции роста научного знания. В советской философии законы развития излагались так [1.5]:

– «Борьба двух противоположностей в одном единстве». Борьба порождает изменение. Отвечает на вопрос «почему происходит развитие?».

– «Накопление количественных изменений приводит к качественным изменениям». Отвечает на вопрос «как происходит развитие?».

– «Отрицание отрицания». Каждый последующий период отрицает предыдущий. У двух периодов (через период) признаки повторяются. Отвечает на вопрос «в каком направлении происходит развитие?».

§1.2. Концепции роста научного знания

•Кумулятивная (монотонное накопление знания);
•Пессимистические схемы (скептицизм и агностицизм);
•Пантеоретизм (постоянная переинтерпретация результатов);
•Позитивизм (проверенная теория неопровержима);
•Фальсификационизм Поппера (знание всегда предположительно);
•Теория научных революций Куна;
•Синергетика (знание развивается по законам нелинейных систем).

Научное знание постоянно изменяется не только по объему, но и по содержанию: обнаруживаются новые факты, рождаются новые гипотезы, на смену старым теориям приходят новые. По мере накопления знания выдвигались разные концепции, описывающие его рост. Хотя, справедливости ради, следует сказать и о существовании пессимистичных схем познания, считающих создание таких концепций бессмысленным. Так, скептицизм призывает ограничить познание одними фактами, а агностицизм считает любое познание невозможным.

Кумулятивная – одна из первых моделей, характеризует науку как постепенное накопление твердо установленных и доказанных истин. Наука представлялась как своего рода склад абсолютных истин, рост знаний виделся непрерывным и монотонным.

Пантеоретизм гипертрофированно понимает роль теории в науке и

говорит, что достаточно изобретательный ученый всегда будет в состоянии так перестроить научную теорию или таким образом переинтерпретировать противоречащие ей результаты экспериментов, что последние окажутся ее подтверждением. Поэтому фундаментальные теории принципиально нефальсифицируемы ни экспериментальными данными, ни отдельными изолированными гипотезами.

Позитивизм говорит, что научная теория, успешно преодолевшая достаточное число эмпирических проверок, приобретает высокую степень подтверждения относительно своей первоначальной области применения и оказывается застрахованной от опасности дальнейшего опровержения. Если эта теория окажется не в состоянии предсказать какие-то новые виды явлений, то их открытие потребует новых технических средств для ее проверки, которые следует ввести как дополнительные правила соответствия. Это означает вытеснение исходной теории тесно связанной с ней более исчерпывающей теорией. Такой научный прогресс состоит в выдвижении последовательности теорий, между которыми имеет место отношение редукции.

К середине XX века философия подошла с убеждением, что крупнейшие научные теории – фикция, а научное знание – результат соглашения. Реальная наука упорно не помещалась в рамки позитивизма или пантеоретизма. Первую попытку пересмотреть традицию верификации знания предпринял Поппер

Концепция фальсификационизма Поппера (1934)

Карл Поппер (1902–1994) задался вопросом, что отличает научные теории (типа эйнштейновской) от псевдонаучных доктрин, к которым он относил труды Маркса, Фрейда и Адлера, и пришел к выводу, что научной теорию делает не подтверждение и не доказательство ее положений, а способность исключать возможность некоторых событий. Поппер решительно отверг мнение о бессмысленности метафизических теорий, а также концепцию, согласно которой теория приобретает значение и становится научной только в том случае, если возможна ее индуктивная верификация с помощью эмпирических наблюдений.

Научное знание, согласно Попперу, внутренне несовершенно и всегда предположительно. Его рост происходит не за счет оправдания теорий, а в ходе критики спекулятивных гипотез, которые предлагаются в качестве решений стоящих перед нами проблем. Истинность научных теорий не может быть доказана, их не следует считать имеющими какое-то оправдание или подтверждение. Однако эта неспособность оправдать знание вовсе не обязательно приводит к иррационализму, поскольку мы всегда можем критиковать наши теории, проверяя их предсказания на опыте.

Концепция научных революций Куна (1947)

В 60-е годы XX в. стала популярной концепция развития науки, предложенная американским философом Томасом Куном (1922–1996). Основными элементами куновской модели являются понятия «научная парадигма», «научное сообщество», «нормальная наука» и «научная революция» (рис. 1.1). Взаимоотношение этих понятий, образующих систему, составляет ядро куновской модели функционирования и развития науки. Парадигма, центральное понятие модели, есть совокупность признанных всеми научных достижений и образец создания новых теорий в соответствии с уже имеющимися в данное время. Содержание парадигм попадает в учебники и проникает в массовое сознание. Парадигмы обуславливают постановку новых опытов, выяснение и уточнение значений конкретных величин, установление конкретных законов. Вокруг парадигмы объединяется научное сообщество.

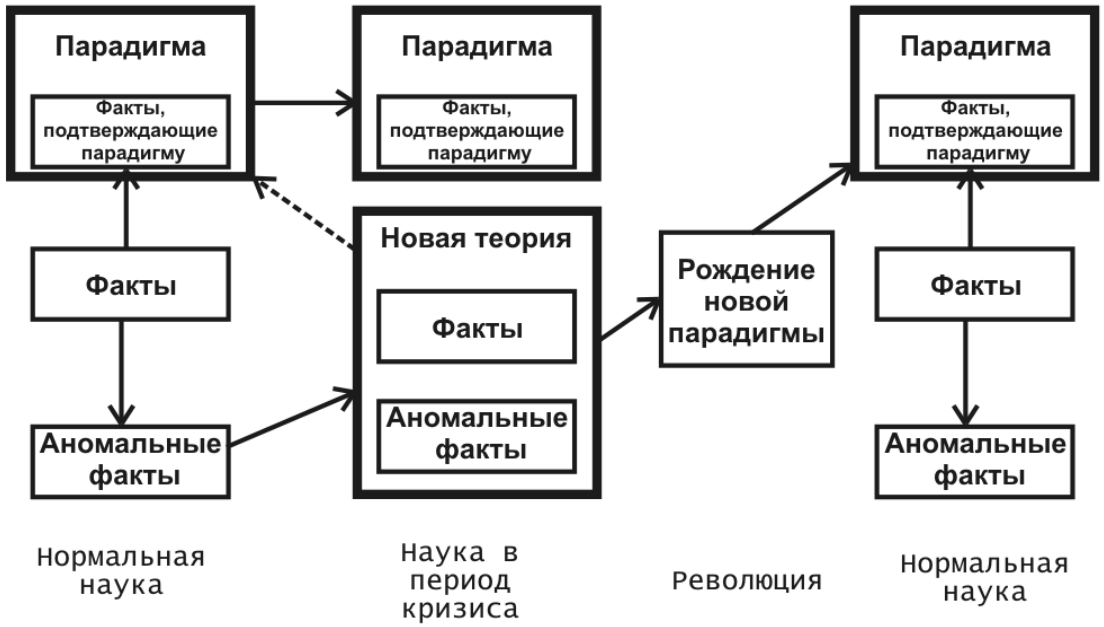


Рис. 1.1. Схема научных революций Т. Куна

Приращение знания в рамках одной парадигмы Кун называет «нормальной наукой», а смену парадигмы и, соответственно, переход от одной «нормальной науки» к другой – научной революцией. Согласно куновской модели в периоды революций возникает конкурентная борьба пар «парадигма – сообщество», которая разворачивается между сообществами. Поэтому победа в этой борьбе определяется, в первую очередь, социально-психологическими, а не содержательно-научными факторами (это связано с «несоизмеримостью» теорий, порожденных разными парадигмами).

Синергетика [1.6]

Синергетика – наука, занимающаяся изучением процессов самоорганизации открытых нелинейных систем самой различной природы, возникла из трудов И. Пригожина по термодинамике (1947 г.), А. Тьюринга по морфогенезу (1952 г.) и Г. Хакена по лазерной оптике (1960 г.). В нелинейных системах, благодаря нелинейности внутренних процессов и притоку энергии извне, возникают явления самоупорядочивания, и энтропия системы (мера беспорядка) начинает убывать. Синергетика предоставила методологическую основу и математический аппарат для исследования неустойчивых ситуаций и переходных процессов в самых разных науках. На её основе может быть рассмотрена и такая модель роста научного знания, как теория Куна. Терминам «нормальная наука» и «научная революция» в синергетике соответствуют «русло» и «точка бифуркации». В зоне русла сложные системы описываются просто, а в зоне бифуркации (ветвления) дальнейший ход процесса могут определить даже малые изменения или шумы (рис. 1.2).

Из других терминов синергетики, подходящих для общего описания эволюционных процессов, упомянем следующие:

– аттрактор (множество точек в фазовом пространстве динамической системы, к которым стремятся траектории системы),

– фрактал (нерегулярное самоподобное множество),

– турбулентность (нерегулярное изменение параметров системы во времени и от точки к точке).

– катастрофа (скачкообразные изменения при монотонном изменении базовых параметров).

– джокер (области фазового пространства, в которых система может повести себя непредсказуемым образом и произвести резкое перемещение).

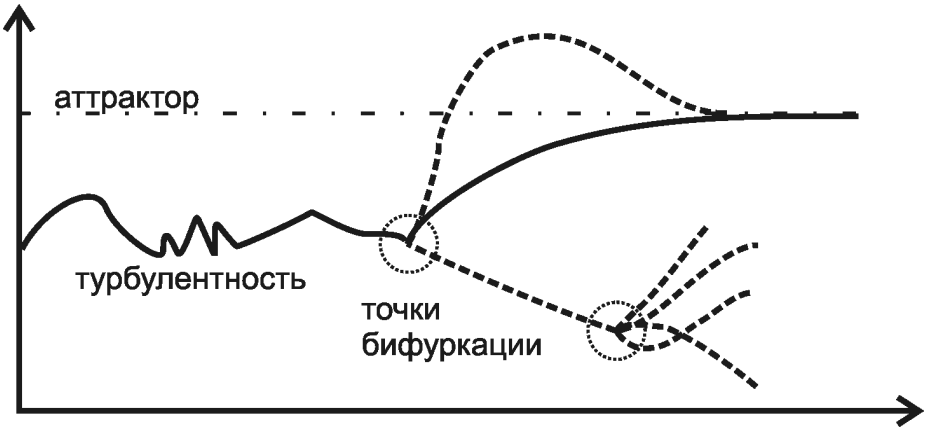


Рис. 1.2. Поведение нелинейной системы с точки зрения синергетики

Любопытно, как философия Гегеля подтверждается математикой

синергетики. Например, закон о переходе количественных изменений в качественные на языке синергетики можно выразить через катастрофу либо джокера. Теория катастроф утверждает, что монотонное изменение параметров нелинейной системы не исключает возможности резкого скачка состояния [1.7], а теория джокеров говорит, что со временем вероятность появления джокера только растет, т.е. длительное монотонное развитие системы маловероятно.

Таковы общие закономерности развития как знания, так и сложных систем вообще. В качестве заключения по главе приведем следующую цитату, равно подходящую и как введение к последующим главам:

«Изучение истории любой технологии предполагает определенный отход от технических достижений к конкретным изобретениям и людям, стоящим за ними, ходу их мысли во время разработки, рассмотрению достижений и успехов, неудач, стоящих за успехом, изделий, стоящих за успехами и неудачами, людей, стоящими за изделиями и ход мысли этих людей. Изучение поучительных случаев из истории обращает внимание на роль ошибок и неудач человека при проектировании, на отличия инженерной и научной деятельности. История помогает нам не только понять, но и придать образ миру, в котором мы живем. Она отображает стремление человеческих стараний и невероятное развитие с помощью человеческого воображения. Она определяет наше место во времени. Понимая историю, мы можем учиться на прошлых ошибках и успехах, использовать уже полученные знания, а также изучить намеренные и случайные последствия использования технологий для общества» [1.8].

ГЛАВА 2. ИСТОРИЧЕСКОЕ РАЗВИТИЕ ВЫЧИСЛИТЕЛЬНОЙ ТЕХНИКИ И ИНФОРМАТИКИ В ДОЭЛЕКТРОННУЮ ЭПОХУ [2.1]

§2.1. Возникновение счета.

Первый след человеческой мысли, связанной с вычислениями, был оставлен около 30 тысяч лет назад, в верхнем палеолите, в период максимума последней ледниковой эпохи, погубившей неандертальцев. При раскопках стоянки охотников на мамонтов в 1937 г., в Дольни-Вестонице (Чехия), среди разных предметов, была найдена испещренная зарубками кость, позволившая предположить, что уже тогда предки человека производили какие-то вычисления.

Переход в палеолите от простого собирания пищи к ее производству, от охоты и рыболовства к земледелию, мало продвинул людей в понимании числовых величин и пространственных отношений. Лишь с неолитом, когда пассивное отношение человека к природе сменилось активным, абстрактные числовые термины стали медленно входить в употребление. Впервые они появляются скорее как качественные, чем количественные термины, выражая различие лишь между одним (или «каким-то») и двумя и многими (или «какими-то»). То, что первобытные люди сначала знали только «один», «два» и «много», подтверждается тем, что в некоторых языках, например в греческом, существуют три грамматические формы: единственного числа, двойственного числа и множественного числа. Такое наглядное, интуитивное представление о числе возникло прежде возникновения счета и обозначения чисел. Позднее человек научился делать различия между двумя и тремя деревьями и между тремя и четырьмя людьми. С расширением понятия числа большие числа сначала образовывались с помощью сложения: 3 путем сложения 2 и 1, 4 путем сложения 2 и 2, 5 путем сложения 2 и 3.

Счет изначально был связан с вполне конкретным набором объектов, и самые первые названия чисел были прилагательными. Об этом свидетельствует тот факт, что слова «один» и «первый», равно как «два» и «второй» («другой»), во многих языках не имеют между собой ничего общего. В то время слова «три» и «третий», «четыре» и «четвертый», лежащие за пределами первобытного счета, указывают на появление взаимосвязи между количественными и порядковыми числительными.

Бирки, зарубки, узелки [2.2]

Бирка – это деревянная палочка (очень редко встречаются бирки из кости или камня), на которую наносят различной формы насечки (зарубки). Длинная прямая зарубка означала единицу, косая – пять,

крестообразная – десять (рис. 2.1). Является простейшим и первым искусственным счётным прибором. Возникновение бирок теряется в глубине веков, они были в употреблении очень долго. Ещё в первой четверти XX столетия ими пользовались многие народы. Вначале бирки служили для фиксирования на память тех или иных чисел. При их помощи вели счёт дням, количеству поголовья скота, записывали величину долга и т.п.

Найденная в Дольни-Вестонице лучевая кость волка была длиной около 17 сантиметров с 55 глубокими зарубками. Первые двадцать пять зарубок размещены группами по пять, за ними идет зарубка двойной длины, заканчивающая этот ряд, а затем с новой зарубки двойной длины начинается второй ряд. Этот древнейший пример пользования бирок.

Узелки – один из видов старинного счёта с помощи верёвок, на которых числа отмечались завязыванием различных узелков (рис. 2.1). Счётные узелки у разных народов считались неприкосновенными и священными. Тот, кто завязал или развязал на подобном документе, не имея на то полномочий, узел, заслуживал самой жестокой кары. В Европе вплоть до средних веков сохранились следы того, что завязанные узлы играли роль судебного доказательства.

У Геродота (V в. до н. э.) есть рассказ о том, как персидский царь Дарий, отправляясь в поход на скифов, приказал ионийцам остаться для охраны моста через реку Истер и, завязав на ремне 60 узлов, приказал развязывать по одному в день, а по их окончании отправляться домой.

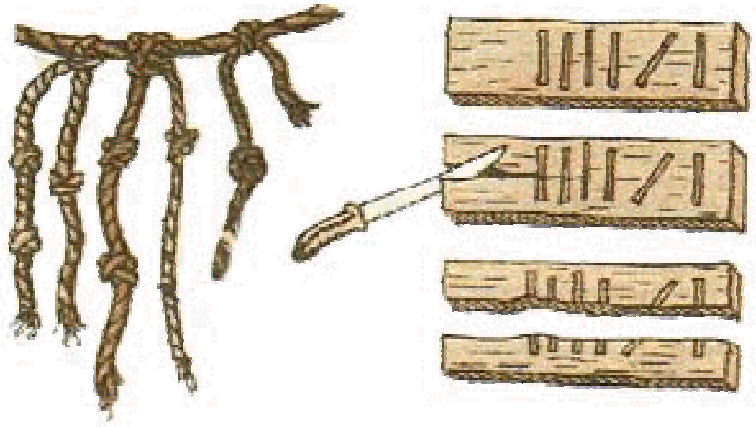


Рис. 2.1. Узелки и бирки

Ещё одним интересным методом счисления были квипу – перуанские счётные верёвки, самая древняя из которых датируется 3 тыс. лет до н.э. Кипу делались из листьев агавы или шерсти. Считали на них с помощью узелков. Маленький узелок мог означать единицу, большой пятёрку или семёрку. Узелки могли быть двойными, тройными или даже четверными, обозначая разные степени числа 10. Квипу окрашивали в разные цвета, чтобы знать, что именно на них считали. Например, на

красных считали мешки с зерном, а на синих – овец. С их помощью, фактически, проходило управление всей империей инков.

Пальцевый счет [2.3]

Пальцевый счет, то есть счет пятками и десятками, возник только на известной ступени общественного развития, развития ремесла и торговли. Так появилась возможность выражать числа в системе счисления, что позволяло образовывать большие числа, и возникла примитивная разновидность арифметики.

Числа группировали и объединяли в большие единицы, обычно пользуясь пальцами одной руки или обеих рук – обычный в торговле прием. Это вело к счету сначала с основанием пять, потом с основанием десять, который дополнялся сложением, а иногда вычитанием, так что двенадцать воспринималось как 10+2, а девять – как 10–1. Иногда за основу принимали 20 – число пальцев на руках и ногах. Умножение зародилось тогда, когда 20 выразили не как 10+10, а как 2×10. Подобные двоичные действия выполнялись в течение тысячелетий, представляя собой нечто среднее между сложением и умножением, в частности в Египте и в доарийской культуре Мохенджо-Даро на Инде. Деление началось с того, что 10 стали выражать как «половину тела», хотя сознательное применение дробей оставалось крайне редким явлением.

Пальцевый счет был широко распространен как в Древней Греции и Риме, так и в средневековье. В поэме Гомера «Одиссея» часто встречается слово «пятерить» в значении «считать». На главной площади Рима Форуме была воздвигнута гигантская фигура двуликого бога Януса. Пальцами правой руки он изображал число 300, пальцами левой – 55. Вместе это составляло 355 – количество дней в году по римскому календарю. Однако в Древнем Риме поняли бесперспективность пальцевого счета и перешли на палочки с зарубками.

Полное описание пальцевого счета составил англосаксонский монах Бе́да Венерабилис Достопочтенный (673–735). Он подробно изложил способы представления на пальцах различных чисел вплоть до миллиона. Его слова – «В мире есть много трудных вещей, но нет ничего труднее, чем четыре действия арифметики».

В настоящее время пальцевый счет находит применение на некоторых фондовых биржах, а также в играх, с помощью которых родители обучают своих детей счету.

Счетные доски [2.3]

- Счет на камнях (пирамидки).
- Древнегреческий абак («саламинская доска»).
- Китайские счеты суан-пан.
- Русский «дощаный счет».

Чтобы сделать процесс счета более удобным, первобытный человек начал использовать вместо пальцев небольшие камни. Он складывал из камней пирамиду и определял, сколько в ней камней, но если число велико, то подсчитать количество камней на глаз трудно. Поэтому он стал складывать из камней более мелкие пирамиды одинаковой величины, а из-за того, что на руках десять пальцев, то пирамиду составляли именно десять камней.

Абак – это греческое слово *αβαξ*, т.е. «счетная доска» или «саламинская доска» по имени острова Саламин в Эгейском море (рис. 2.2). Древнегреческий абак представлял собой посыпанную морским песком дощечку. На песке проходились бороздки, на которых камешками обозначались числа. Абак придумали финикийцы, а потом этот способ счёта переняли другие народы. В V в. до н.э. абак получил широкое распространение в Греции и Египте.

В древнем Риме употреблялся такой же абак, как в Греции, а также усовершенствованный абак. Такой римский абак представлял собой металлическую пластинку с вертикально прорезанными девятью желобками, вдоль которых передвигались камешки. Внизу помещались камушки для счёта до пяти, а в верхней части было специальное отделение для камушков, обозначающих «пять», то есть римляне считали пятёрками. От латинского слова «calculus», что означает «галька», «камушек», и произошло латинское слово «calculator», что означает «счетчик».

Суан-пан и соробан – это китайский и японский варианты абака (рис. 2.2). Китайцы догадались заменить камушки бусинками на прутиках. Прутики крепились на деревянной раме. Это и был суан-пан. Кроме параллельных горизонтальных прутиков с бусинками, есть ещё перпендикулярная им линейка, которая делит всё устройство на две неравные части. В большом отделении было нанизано по пять шариков – сколько пальцев на руках, а в маленьком по два – сколько рук. А соробан – это то же самое, только в большом отделении четыре бусины вместо пяти.

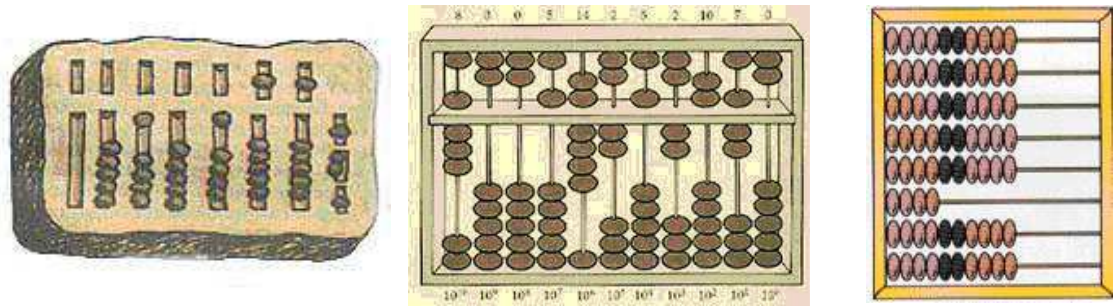


Рис. 2.2. Абак, суан-пан и русские счеты

На Руси долгое время считали по косточкам, раскладываемым в

кучки. Примерно с XV века получил распространение «дощаный счет», завезенный, видимо, западными купцами. Дощаный счет почти не отличался от обычных счетов и представлял собой рамку с укрепленными горизонтальными веревочками, на которые были нанизаны просверленные сливовые или вишневые косточки. В отличие от абака была убрана горизонтальная линейка и добавлены бусины, поэтому счёты раньше назывались «русские щоты» (рис 2.2).

§2.2. Возникновение систем счисления [2.4]

- Египет (ок. 2850 до н.э.)
- Вавилон (2 тыс. лет до н.э.)
- Греческая (5 в. до н.э.)
- Римская (500 лет до н.э.)
- Китай (ок. 4 тыс. лет до н.э.)
- Америка

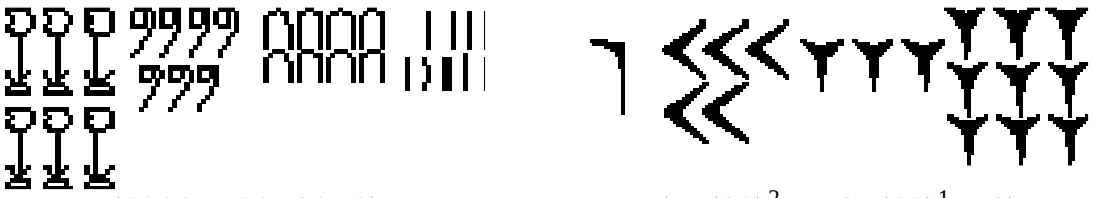
Системы счисления – это различные записи чисел с помощью специальных знаков, называемых цифрами. Число цифр, с помощью которых записываются числа в данной системе счисления, называется основанием системы счисления. Системы счисления появились одновременно с появлением пальцевого счета, поэтому неудивительно, что в качестве основания систем выбирались числа 5, 10, 20, реже – 12. Так, из 307 исследованных систем счисления первобытных американских народов 146 были десятичными, 106 – пятеричными и пятерично-десятичными, остальные – двадцатеричными и пятерично-двадцатеричными.

Системы счисления можно разделить на три части – непозиционные, позиционные и смешанные системы счисления. Непозиционные системы счисления появились исторически первыми, в них значение каждого цифрового символа постоянно и не зависит от положения. Простейшим случаем такой системы является единичная, использующая единственный символ (черта, точка), который всегда ставится в количестве, соответствующем обозначаемому числу. Модификацией единичной системы является система с основанием, в которой есть символы как для обозначения единицы, так и для степеней основания. Примером такой системы с основанием является древнеегипетская, возникшая в третьем тысячелетии до новой эры.

Египет (ок. 2850 до н.э.)

В древнеегипетской системе счисления используются несколько иероглифов: шест (единицы), дуга или подкова (десятки), пальмовый лист или силок (сотни) и цветок лотоса (тысячи) (рис. 2.3). Продолжая в

том же духе, египтяне обозначили десять лотосов согнутым пальцем, десять согнутых пальцев – волнистой линией, и десять волнистых линий – фигуркой удивленного человека. В итоге древние египтяне могли представлять числа до миллиона. Числа в египетской системе, как и в любой непозиционной, получаются сложением, порядок следования любой, но для узнавания цифры группируются. Дроби использовались аликвотные (вида **1**/n).



(6000+700+80+9) $(1x(60)^2 + 53x(60)^1 + 9)$

Рис. 2.3. Пример записи числа 6789 в древнеегипетской и вавилонской системах счисления

В позиционных системах счисления важную роль играет порядок следования цифр. Каждая цифра имеет свою позицию, которая определяет её численное значение. Для системы выбирается основание – положительное натуральное число, тогда любое число представляется как сумма степеней n с целыми коэффициентами a_k от 0 до n–1, называемыми цифрами:

$$x = \sum_{k=-m}^{k=+n} a_k \cdot n^k ,$$

где каждая степень b^k в такой записи называется разрядом (позицией). Например, в десятичной системе $103=1\times10^2+0\times10^1+3\times10^0$. Изобретение позиционного счисления приписывается шумерам и вавилонянам.

Вавилон (2 тыс. лет до н.э.)

Вавилония, возникшая в начале 2 тыс. лет до н.э., унаследовала культуру предыдущих царств Междуречья – Шумера и Аккада, разделивших двумя тысячелетиями раннее год на 12 частей, сутки – на 24, и использовавших 60-ричную систему счисления. Эти же царства ввели клинопись, доминирующую в междуречье до 1 тыс. до н.э. Принято считать, что шестидесятеричная система была выбрана из метрологических соображений: число 60 имеет много делителей. Помимо 60, в вавилонской системе счисления присутствовали основание 10 и 12.

Для малых чисел вавилонская система счисления в основных чертах напоминала египетскую. Одна вертикальная клинообразная черта означала единицу, а для обозначения числа 10 использовался знак в виде

угловой скобки, направленной влево (рис. 2.3). Повторенные определенное число раз, они обозначали соответствующее количество единиц и десятков. Различие наблюдалось только во внешнем виде цифр, поскольку вавилоняне для их записи использовали палочки и глиняные дощечки, а не мягкий папирус.

Но для записи чисел больше 59 древние вавилоняне впервые использовали принцип позиционности. В качестве коллективных символов более высокого порядка стали применяться ранее использованные символы, занимающие в записи числа новое положение левее предыдущих символов низкого порядка. Так, один клиновидный знак мог использоваться для обозначения и 1, и 60, и 60^2, и 60^3, в зависимости от занимаемого им в записи числа положения.

В исключительных случаях вавилоняне применяли сокращенные формы записи, иногда – с новыми символами для обозначения чисел 100 и 1000, или использовали принципы умножения или вычитания.

Превосходство разработанной в Месопотамии системы счисления отчетливо видно в обозначении дробей. Здесь не требовалось вводить новые символы. Как и в современной десятичной позиционной системе, в древневавилонской системе подразумевалось, что на первом месте справа от единиц стоят величины, кратные 1/60, на втором месте – величины кратные $1/60^2$ и т.д. Отсюда берет начало привычное нам деление часа и углового или дугового градуса на 60 минут, а одной минуты – на 60 секунд.

Греческая (5 в. до н.э.)

В Древней Греции имели хождение две основных системы счисления – аттическая и ионическая (александрийская или алфавитная) (рис. 2.4).

Аттическая система счисления использовалась греками, по-видимому, уже к 5 в. до н.э. По существу это была десятичная система. Используя число 5 как промежуточное подоснование системы счисления, греки на основе принципа умножения комбинировали пятерку с символами степеней числа 10. Единица в аттической системе обозначалась вертикальной палочкой, остальными цифрами были пента (П, 5), дека (D, 10), гекатон (H, 100), хилиои (X, 1000), мириои или мириада (M, 10000). Числа больше 50000 обычно описывались словами.

Ионическая система счисления получила широкое распространение III веке до н.э., в начале Александрийской эпохи, хотя возникнуть она могла уже у пифагорейцев. Эта система счисления была чисто десятичной. Используя двадцать четыре буквы греческого алфавита и еще три архаических знака, ионическая система сопоставила девять букв первым девяти числам (α–θ); другие девять букв – первым девяти целым кратным числа десять (ι–ϟ); и последние девять символов

– первым девяти целым кратным числа 100 (ρ–ϡ). Для обозначения первых девяти целых кратных числа 1000 греки частично воспользовались древневавилонским принципом позиционности, снова использовав первые девять букв греческого алфавита, снабдив их штрихами слева. По этому же принципу, но по собственному алфавиту, действовала и древнерусская система счисления.

(5+1)*1000 + (5+2)*100 + (5+3)*10 + (5+4) 6*1000 + 700 + 80 + 9

Рис. 2.4. Пример записи числа 6789 в аттической и ионической системах счисления

Греческое алфавитное обозначение целых чисел можно было бы легко приспособить для обозначения десятичных дробей, но они игнорировались по нескольким причинам. Например, само слово «число» греки понимали как набор целых единиц. Кроме того, десятичные представления обыкновенных дробей в большинстве случаев бесконечны, а бесконечность была исключена из строгих рассуждений.

С другой стороны, областью, в которой практические вычисления испытывали величайшую потребность в точных дробях, была астрономия, а здесь вавилонская традиция была настолько сильна, что шестидесятеричная система обозначений угловых, дуговых и временных величин сохраняется и поныне. Поэтому в ходу были аликвотные и шестидесятеричные дроби, в которых греки заменили клинопись буквенными обозначениями.

Римская (500 лет до н.э.)

Широкая известность римского обозначения чисел даже среди современных систем объясняется огромным влиянием, которым пользовалась Римская империя в сравнительно недавнем прошлом. Этруски, завоевавшие Рим в 7 в. до н.э., испытали на себе влияние восточно-средиземноморских культур. Этим отчасти объясняется сходство основных принципов римской и аттической систем счисления – это десятичные системы, в которых особую роль играет число пять. Обе системы использовали при записи чисел повторяющиеся символы. Разумеется, в деталях они отличались. Старыми римскими символами для обозначения чисел 1, 5, 10, 100 и 1000 были, соответственно, символы I, V, X, Q и Ф. Согласно одной из распространенных теорий, римская цифра V изображает раскрытую руку с четырьмя прижатыми друг к другу пальцами и отставленным большим пальцем; а символ X изображает две скрещенные руки или сдвоенную цифру V. Символы чисел 100 и 1000, возможно, берут начало от греческих букв Q и Ф.

Более поздние обозначения C и M либо произошли от старых римских символов, либо они акрофонически связаны с начальными буквами латинских слов, означавших 100 (центум) и 1000 (милле). Полагают, что римский символ числа 500, буква D, возник из половинки старого символа, обозначавшего 1000.

Римляне часто использовали принцип вычитания, поэтому иногда вместо VIIII использовали IX и XC вместо LXXXX; позднее IV вместо IIII (рис. 2.5). Поначалу древние римляне избегали записывать число IV вместо IIII, возможно потому, что символ IV совпадает с первыми двумя буквами старолатинского написания имени Юпитер.

MMMMMMDCCLXXXIX

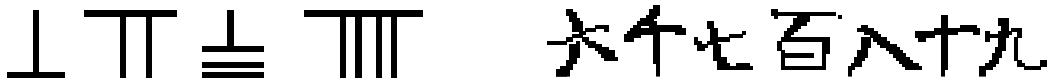
Рис. 2.5. Пример записи числа 6789 в древнеримской системе счисления

Непозиционная система счисления неудобна для ведения расчетов, поэтому неудивительно, что в абаке, существовавшем параллельно, использовалось десятичное позиционное счисление. В целом римляне не были склонны заниматься математикой, поэтому не испытывали особой потребности в больших числах. Тем не менее для обозначения 10000 и 100000 они эпизодически использовали отдельные символы, а для обозначения чисел 5000 и 50000 – их половинки.

Дробей римляне избегали так же упорно, как и больших чисел. В практических задачах, связанных с измерениями, они не использовали дроби, подразделяя единицу измерения обычно на 12 частей, с тем чтобы результат измерения представить в виде составного числа, суммы кратных различных единиц, как это делается сегодня, когда длину выражают в ярдах, футах и дюймах. Английские слова «ounce» (унция) и «inch» (дюйм) происходят от латинского слова uncia (унция), обозначавшего одну двенадцатую основной единицы длины.

Китай

Одни из древнейших систем счисления были созданы в Японии и Китае (рис. 2.6). Из китайских систем самая первая возникла как результат оперирования с палочками, выкладываемыми для счета на стол или доску. При превышении 99 система превращалась в позиционную. Вторая китайская система счисления, возникшая около 4 тыс. лет назад, для обозначения первых девяти целых чисел или символов использовала девять различных знаков и одиннадцать дополнительных символов для обозначения первых одиннадцати степеней числа 10. В сочетании с умножением и вычитанием это позволяло записывать любое число меньше триллиона.



(60 + 7)*100 + 80 + 90 6*1000 + 7*100 + 8*10 + 9

Рис.2.6. Пример записи числа 6789 в двух китайских системах счисления

Америка

Системы счисления американского континента интересны тем, что на их развитие не оказывали влияние цивилизации Европы и Азии. Исследователи, путешествовавшие в XVI веке по Центральной Америке, обнаружили цивилизации с высокоразвитыми системами счисления, отличными от тех, которые были известны в Европе. У индейцев майя самыми важными элементами в системе счисления были позиционный принцип и символ нуля. Если отвлечься от того, что принятая у майя система счисления была не шестидесятеричной, а двадцатеричной и вместо 10 использовала вспомогательное основание 5, то в остальном принципы были аналогичны вавилонским. В схеме майя точка означала единицу, а горизонтальная черта – пятерку. Для обозначения числа двадцать майя воспользовались позиционным принципом, используя точку, помещенную над символом нуля.

Система счисления у ацтеков была более последовательно двадцатеричной, чем у майя, но в остальном менее тонкой, так как не использовала ни позиционный принцип, ни специальный символ для нуля. Точка означала у ацтеков единицу, а для обозначения степеней числа 20 были введены новые знаки: флаг для 20, дерево для 400 и кошелек для 8000. При необходимости другие числа представлялись с помощью повторения этих символов, а от их чрезмерного повторения они избавлялись, вводя специальные промежуточные коллективные знаки: ромбовидный знак для 10 и фрагменты дерева для 100, 200 или 300.

Североамериканские индейцы не имели письменности, а используемые названия чисел были в основном прилагательными и лишь в отдельных случаях достигали уровня абстракции, когда они становились существительными. Тем не менее, с помощью рисунков или устно индейцы могли выразить число вплоть до миллиона. Системы составления чисел были самыми различными, но примерно половина из них по существу была десятичной.

Как видно, десятичная позиционная система в разных несовершенных формах появлялась у многих цивилизаций. Но начало современной десятичной системе дала еще одна вариация десятичной системы, возникшая у индусов как счет на пальцах.

§2.3. Возникновение современной десятичной системы счисления

Индия

Индийские системы счисления проходили в своем развитии те же этапы, что и во всех прочих цивилизациях. На древних надписях из Мохенджо-Даро (ок. 2 тыс. лет до н.э.) вертикальная черточка в записи чисел повторяется до тринадцати раз, а группировка символов схожа с египетской. Затем имела хождение система счисления «кхарошти», очень напоминающая аттическую, в которой для обозначения чисел 4, 10, 20 и 100 использовались повторения коллективных символов. Эта система постепенно уступила место системе «брахми», где буквами алфавита обозначались единицы (начиная с четырех), десятки, сотни и тысячи. Переход от кхарошти к брахми происходил в те годы, когда в Греции, вскоре после вторжения в Индию Александра Македонского, ионическая система счисления вытеснила аттическую. Вполне возможно, что переход от кхарошти к брахми происходил под влиянием греков. Надписи, найденные в Нана-Гат и Насике, относящиеся к первым векам до нашей эры и первым векам нашей эры, по-видимому, содержат обозначения чисел, которые были прямыми предшественниками современной индо-арабской системы. К 8–9 вв. эта система получает символ нуля и обретает элементы современной системы счисления: индийская система была десятичной, цифровой и позиционной. При желании можно даже усмотреть некоторое сходство с начертанием современных цифр (рис. 2.7).

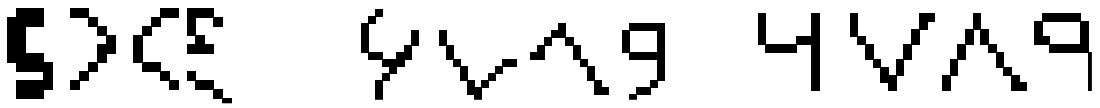


Рис. 2.7. Примеры записи числа 6789 в индийской, арабской «индийской» (8 в.) и арабской «испанской» (10 в.) системах счисления

Принцип позиционности, вероятнее всего, проник в Индию из древнего Вавилона. Поскольку индийские астрономы использовали шестидесятеричные дроби, вполне возможно, что это навело их на мысль перенести позиционный принцип с шестидесятеричных дробей на целые числа, записанные в десятичной системе.

Аравия

Современную систему обозначения чисел часто называют арабской, но она берет начало не из Аравии. До хиджры (переселения пророка Мухаммеда из Мекки в Медину) арабы записывали числа словами, но затем, как это делали ранее греки, они стали обозначать числа буквами своего алфавита. В 772 индийский трактат «Сидданта»

был привезен в Багдад и переведен на арабский, после чего после чего в торговых расчетах арабские купцы стали применять систему, заимствованную из Индии. Старая, алфавитная система, по-прежнему употреблялась астрономами. Среди тех, кто пользовался индийской системой, начертания цифр, как и в Индии, сильно варьировали, и зачастую арабские цифры из разных частей халифата казались никак не связанными (рис. 2.7). Тем не менее, удобство записи привело к тому, что индийские цифры были позаимствованы у арабов итальянскими купцами. Так они попали в средневековую Европу.

§2.4. Недесятичные системы счисления

- двенадцатеричная
- шестидесятеричная
- восьмеричная и шестнадцатеричная
- троичная
- двоичная

Возникшие в древности у разных народов 5-, 10- и 20-ричная системы имеют анатомическое происхождение – они связаны с количеством пальцев на одной, двух руках и руках вместе с ногами соответственно. 12-ричную систему можно связать с количеством фаланг пальцев на руке, если считать их большим пальцем. Но, скорее всего, её появление у шумеров связано с делимостью, делавшим число «правильным». С делимостью вероятно связано и появление 60-ричной системы. Из всего разнообразия систем древности до нашего времени дожила и осталась популярной десятичная европейская система счисления, также сохранились в ходу 12-ричная и 60-ричная системы.

Двенадцатеричная и шестидесятеричная системы удобны из-за наличия множество делителей, используются для обозначения времени, углов. 12-ричная система до сих пор применяется у некоторых народов Нигерии и Тибета, деление шиллинга на 12 пенсов отменено лишь в 1971г., а деление фута на 12 дюймов и употребление термина «дюжина» применяется и в настоящем.

Восьмеричная и шестнадцатеричная системы удобны для записи двоичных чисел, поскольку группируют цифры по 3 или 4 разряда.

Периодически поднимается вопрос об оптимальности самой популярной системы – десятичной, и предлагается переход на восьмеричную или двенадцатеричную систему, главные преимущества которых связаны с делимостью их оснований. Но такой переход требует полного пересмотра таблиц сложения и умножения и прочие сложности, что сводит ожидаемые преимущества на нет.

Целочисленной позиционной системой, самой экономичной по числу знаков, считается троичная система (наиболее близка к числу e),

которая иногда рассматривается как возможная альтернатива двоичной системе. Разработана троичная логика, а одно время даже выпускалась троичная ЭВМ «Сетунь» (1959). В ней вместо одного троичного элемента использовалось два двоичных, но все равно машина обладала преимуществами перед двоичной.

Двоичная система счисления

Двоичное счисление не имеет таких корней в обыденно-практическом опыте, какие есть у пятеричной или десятичной систем, и встречается на протяжении тысячелетий в единичных случаях, но в современной вычислительной технике играет исключительную роль.

Древнейшим использованием двоичного счисления, которое применялось уже в I–II тысячелетии до н.э., возможно, является кипу – узелковая запись инков и их предшественников, состоявшая как из числовых десятичных записей, так и нечисловых двоичных.

Первое известное науке описание двоичной системы счисления дал древнеиндийский математик Пингала (IV или II век до н.э.), описавший ведическую систему стихосложения с короткими и длинными слогами. В его системе четыре коротких слога (двоичные 0000) означали единицу, поскольку ноль в позиционной системе еще не использовался.

Китайская Книга Перемен династии Чжоу (1122–771 гг. до н.э.) состоит из 64 символов – гексаграмм, каждый из которых выражает ту или иную жизненную ситуацию на разных стадиях развития. Символы состоят из шести черт, которые обозначают последовательные ступени развития. Гексаграмма делилась на две триграммы – верхнюю и нижнюю (рис. 2.8). Порядок расположения гексаграмм в соответствии со значениями соответствующих двоичных цифр (от 0 до 63) разработал китайский философ Шао Юн (1011–1077), но неизвестно, располагал ли он их по правилам двоичной арифметики, или же просто по порядку возрастания двухсимвольных буквенных обозначений гексаграмм [2.5].

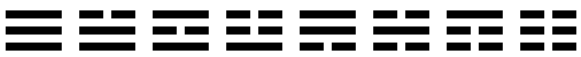


Рис. 2.8. Триграммы Книги Перемен – Цянь (Небо), Дуй (Водоем), Ли (Огонь), Чжень (Гром), Сунь (Ветер), Кань (Вода), Гень (Гора) и Кунь (Земля)

Современная двоичная система описана Лейбницем в работе «Explication de l’Arithmétique Binaire» (1703), в которой он использует современные обозначения 0 и 1. Лейбниц не рекомендовал использовать двоичное счисление для практических вычислений, но отмечал, что при переходе к нему появляется «чудесный порядок». Лейбницем, увлекавшимся китайской культурой, была замечена и Книга Перемен. Он

заметил, что гексаграммы соответствуют числам от 0 до 111111, и восхищался тем, что это свидетельствует о достижениях китайской философии того времени.

В 1854 г. Дж. Буль сформулировал основы математической логики и подробно описал двоичную систему счисления. В 1936–1938 годах американский инженер и математик Клод Шеннон нашёл применения двоичной системы при конструировании схем из реле и переключателей. Вскоре двоичная арифметика и двоичная логика были повсеместно внедрены в компьютерную технику, построенную на бистабильных элементах.

§2.5. Средства автоматизации счета в раннее Новое время

Счетные палочки Непера

Переход из Средневековья в Новое время сопровождался бурным ростом производства и торговли, великими научными и географическими открытиями, существенными изменениями в социальной структуре общества. В это же время, в XV–XVII века, активно развиваться мореплавание и астрономия. Чтобы составить астрономическую или мореходную таблицу привлекались десятки, а то и сотни человек, которые складывали, умножали, делили и извлекали корни. Джон Непер (1550–1617) придумал специальные таблицы для вычисления, которые были названы «палочками Непера», и позволяли быстро выполнять операции умножения и деления. Непер писал, что «…нет ничего более хлопотного в математической практике, что более досаждало бы вычислителю, чем выполнение над большими числами умножения, деления, извлечения квадратных и кубических корней, которые сопряжены обычно с массой трудно обнаруживаемых ошибок».

Алгоритм счета на палочках Непера восходит к одному из древнейших способов умножения – «gelosia», который был введен в использование в Европе в 1202 г. В нем сомножители записывались вдоль верхней и левой соседних граней решетки, поразрядно умножались, а при суммировании «по косой» вдоль нижней и правой граней решетки получался результат (рис. 2.9).

Собственная идея Непера – разрезать таблицу на столбцы и выполнять действия, подбирая нужные палочки в соответствии с составом числа. Естественно, что для «ввода» числа в наборе должно быть больше палочек, цифры могут повторяться. Таким образом, умножение становится тривиальной задачей, но этим потенциал палочек не исчерпывается, с ними можно выполнять и деление, и возведение в степень, и извлечение корня, опираясь на сложение и вычитание логарифмов.

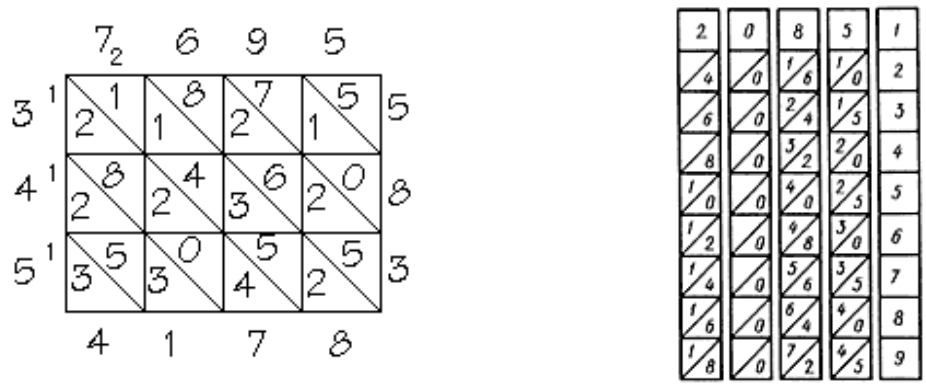


Рис. 2.9. Алгоритм «gelosia» (543×7695 = 4178358) и палочки Непера

Наряду с палочками Непер предложил счетную доску для выполнения четырех арифметических действий, а также возведения в квадрат и извлечения в двоичной системе счисления квадратного корня, предвосхитив тем самым преимущество двоичной системы для автоматизации вычислений. Также он – один из изобретателей логарифмов, открытие которых послужило основой создания логарифмической линейки, более 300 лет отслужившей инженерно-техническим работникам всего мира. Таблицами логарифмов Непера широко пользовались при вычислениях астрономы, математики, штурманы дальнего плавания.

Логарифмическая линейка (1654)

В 1654 г. Роберт Биссакар, а в 1657 г. независимо С. Патридж, разработали прямоугольную логарифмическую линейку, заменившую операции умножения и деления операциями сложения и вычитания логарифмов. Линейка состояла из двух шкал в логарифмическом масштабе, способных передвигаться относительно друг друга. Первая линейка, пригодная для выполнения любых инженерных расчетов, была сконструирована в 1779 году Дж. Уаттом. Логарифмические линейки широко использовались до 1980-х гг. (в СССР), когда были вытеснены микрокалькуляторами.

Конструкция линейки за века практически не изменилась, хотя более сложные линейки содержат дополнительные шкалы и прозрачный бегунок с несколькими рисками. На обратной стороне линейки могут находиться какие-либо справочные таблицы. С помощью логарифмической линейки находят лишь мантиссу числа, его порядок вычисляют в уме. Вычисления с помощью логарифмической линейки производятся просто, быстро, но приближенно – обычно это два-три десятичных знака. Поэтому она не годится для точных расчетов, например финансовых.

§2.6. Арифметические машины

Прообразы арифметических машин

Арифметические машины предназначались для выполнения простейших математических операций. Рисунок предположительно первого десятичного автоматического суммирующего устройства был найден в работах Леонардо да Винчи (1452–1519) в 1967 г. (рис. 2.10). Устройство состояло из 13 колес с десятью зубцами, числа представлялись в виде углового положения оси или колеса, а поворот одного колеса вызывал поворот соседнего в пропорции 1:10 [2.6]. Неизвестно, задумывалась ли машина для суммирования, но по подобной схеме в дальнейшем и начали разрабатываться счетные машины.

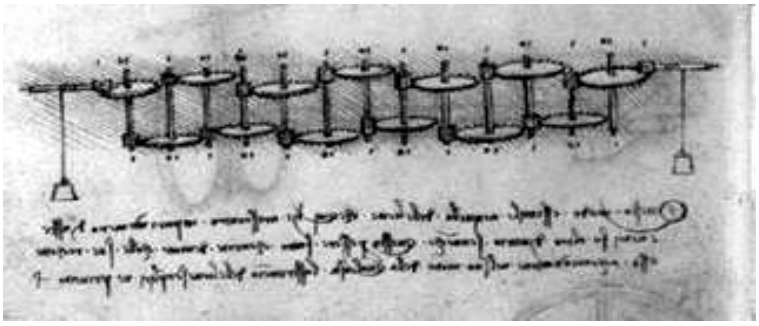


Рис. 2.10. Суммирующее устройство да Винчи

Автором второй вычислительной машины считается Вильгельм Шиккард (1592–1635), в переписке которого с астрономом Кеплером в 1957 г. был найден эскиз машины. Судя по переписке, Шиккард конструировал счётную машину для сложения, вычитания и умножения. Машина разделялась на три блока – суммирования, умножения и промежуточного хранения, состояла из 11 десятизубых, 6 однозубых колес и оперировала с 6-разрядными числами. В 1624 г. она сгорела в пожаре. Упоминания о работах да Винчи и Шиккарда были найдены только в XX веке, поэтому их идеи не повлияли на развитие вычислительной техники.

Вычислитель Паскаля (1642)

Блез Паскаль (1623–1662) в 1645 году представил счётную машину, выполняющую сложение и вычитание и предназначенную для облегчения труда отца – сборщика налогов. В свой машине Паскаль использовал пронумерованные шестерёнки. Нужную цифру устанавливали, поворачивая колесо на горизонтальной оси до тех пор, пока соответствующий цифре зубец не появлялся в окошке. Для прибавления колесо поворачивалось на соответствующее число зубьев,

при полном повороте десятки переходили на следующее колесо.

Известие о счетной машине воспринималось как чудо. В 1649 г. Паскаль получил королевскую привилегию, которая устанавливала его приоритет в изобретении и закрепляла за ним право производить и продавать машины, но производство продолжалось всего 3 года, т. к. машина могла только складывать и вычитать, и то с трудом. Она так осталась придворной игрушкой и широкого признания не получила. Было выпущено несколько десятков экземпляров машины – из дерева, из меди, из слоновой кости и пр.

Машина Лейбница (1673)

Лейбниц (1646–1716) в 1673 г. усовершенствовал машину Паскаля, разделив его на подвижную и неподвижную часть, а также введя в конструкцию ступенчатый цилиндр («колесо Лейбница»). Это позволило реализовать операции умножения и деления, так как подвижная часть играла роль аккумулятора, который накапливал промежуточные результаты, образующиеся при поразрядном умножении. Из-за ступенчатого механизма умножения и деления машина получила название «Ступенчатый вычислитель». Она позволяла также вычислять корень. Для приведения машины в действие оператор должен был вращать ручку, связанную с осью машины. Поводом, побудившим Лейбница к созданию машины для вычислений, стало знакомство с астрономом Гюйгенсом. Астрономам приходилось выполнять огромный объем вычислений, и Лейбниц говорил, что «недостойно одарённому человеку тратить, подобно рабу, часы на вычисления, которые можно было бы доверить любому лицу, если при этом применить машину».

Изделие Лейбница также не стало массовым – во-первых, время спроса на подобные механизмы еще не пришло, во-вторых, механизм работы был далек от оптимального, сам Лейбниц потратил почти 40 лет на его усовершенствование. Однако многие математики начали строить машины по подобию машины Паскаля или Лейбница, хотя они так же не были массовыми. В России такая машина была создана в 1770 году Евно Якобсоном.

Арифмометры (1818)

Арифмометр – это настольный цифровой механический вычислитель, выпускаемый, в отличие от предшественников, серийно. Первые коммерчески успешные счётные машины – 2, 4, 11 и 14-разрядные арифмометры – создал в 1774 г. Филипп-Малтус Хан, продавший небольшое количество таких моделей. К XIX веку технология точной обработки металлов достигла значительных успехов, стал возможен массовый выпуск арифмометров. Это сделал 1818 г. К. Томас, модернизировавший машину Лейбница и начавший серийный

выпуск. Было выпущено большое количество моделей разных фирм и конструкций, расцвет приходится на 1950-е годы, в СССР арифмометры применялись до 1980-х. Помимо полностью «ручных» арифмометров выпускались и «автоматические», самостоятельно выполнявшие сложные операции, такие как умножение и деление.

Ввод чисел в арифмометрах применялся последовательный (ввод цифр числа по порядку слева направо, как в микрокалькуляторах) и параллельный (цифры вводятся независимо друг от друга, как в машине Паскаля). Устройством вывода был счётчик (нескольких колёс с цифрами – вертикальными и горизонтальными), в арифмометре их было два – счетчик суммирования и счетчик прокруток. Счётчик прокруток показывал количество сделанных поворотов ручки арифмометра (количество условных сложений/вычитаний).

§2.7. XIX век. Предвестники цифровой вычислительной техники

XIX век отмечен несколькими событиями, ставшими первыми шагами на пути создания цифровой электронной вычислительной техники. Это начало программирования с помощью перфокарт, появление средств обработки перфокарт, разработка универсальной цифровой вычислительной машины и создание двоичной логики.

Ткацкий станок Жаккарда (1801)

Жозеф Мари Жаккард (1752–1834), сын лионского ткача, изобретал ткацкий станок с 1790 года и представил его в 1801 г. Тремя годами позже он повез свою машину в Париж, где автоматы Жака Де Вокансона навели его на окончательную конструкцию станка, осуществленную в 1808 году. Наполеон Бонапарт предоставил Жаккарду право взимания премии в 50 франков с каждого действующего во Франции станка его конструкции. В 1812 году во Франции работало 18 тысяч станков Жаккарда.

В этих станках для задания узора на ткани использовались отверстия в металлических пластинах. Крючки проходили через отверстия в пластинах и протягивали вниз нити основы, в результате чего челнок проходил над определенным образом выбранными нитями. Это позволяло наносить рисунок на ткань без участия человека. Пластины считаются прообразом перфокарт, а сам станок – первой не счетной программируемой машиной.

Станок Жаккарда относится к тем изобретениям, которые перевернули жизнь многих людей, оставив их без работы и средств к существованию, что и привело к восстанию лионских ткачей (1831, 1834 гг.).

Аналитическая машина Ч. Бэббиджа (1830-е) [2.7]

Начало эры компьютеров в том виде, в котором они существуют сейчас, связано с Чарльзом Бэббиджем, который в 30-х годах XIX века предложил идею вычислительной машины, осуществленную лишь в середине XX века. На мысль о построении технологии вычислений Бэббиджа натолкнули работы Гаспара де Прони, нашедшего алгоритмические и методологические подходы для сведения сложных вычислений к рутинным операциям.

Бэббидж обнаружил погрешности в таблицах логарифмов и поначалу обратил внимание на то, что машина может без ошибок выполнять вычисление больших математических таблиц посредством простого повторения шагов. Работая над этой проблемой, в 1822 году Бэббидж предложил проект «разностной машины» для вычислений путём аппроксимации функций многочленами, позволявшей вычислять значения многочленов до шестой степени с точностью до 18-го знака. Для повторения операций в машине предполагалось использование энергия пара. Вместо запланированных 3-х лет, из-за сложности реализации, работа заняла 9 лет и была завершена лишь частично.

Незавершенность первой работы также связана с тем, что Бэббиджа заинтересовала новая идея – создание универсальной «аналитической машины», способной выполнять широкий круг задач. Эта машина должна была делать одно сложение в секунду и работать без вмешательства человека. Работала машина по принципу мельницы: на «складе» хранятся числа – «зёрна», которые через «воронку» поступают в «мельницу», делающую арифметические операции. А из «желоба» высыпается «мука» – результат. Для хранения чисел на «складе» использовались колёса Паскаля, объединенные в регистры, по десять в каждом. Для передачи чисел со «склада» на «мельницу» использовались зубчатые рейки. Предполагалось хранение в памяти 1000 слов, каждое слово – 50 десятичных разрядов.

Но «склад» и «мельница» – это ещё не автоматическая машина. Было еще управляющее устройство, работающее по набору инструкций, записанных в виде определенной последовательности дырочек на перфокартах. Таким образом машина Бэббиджа была первой программируемой счётной машиной. Перфокарты представляли собой прямоугольные карточки из картона и были двух типов – управляющие маленькие карты размером 13,0×5,5 см и большие для хранения чисел, размером 18,5×7,0 см.

Первое подробное описание изобретения сделано в 1842 году итальянским военным инженером Луиджи Федерико Менабреа в статье «Очерк Аналитической машины, изобретенной Чарльзом Бэббиджем», опубликованной на французском языке. Статья была переведена на английский Адой Августой Лавлейс и издана с обширными

комментариями в 1843 г. (объем комментариев составил 50 страниц при объеме статьи 20 страниц). В комментариях, подготовленных совместно с Бэббиджем, Ада Лавлейс поясняет принцип работы машины и рассматривает несколько примеров – нахождение корней уравнения и расчет линейной и тригонометрической функций [2.8].

Аналитическая машина, состоящая более чем из 50000 компонентов, так и не была построена, так как у Бэббиджа не хватило денег на её строительство, а люди не верили в эту затею, называя её «чудачеством Бэббиджа». Тем не менее идеи, заложенные Бэббиджем, оказали огромное влияние на развитие вычислительной техники. Это автоматизация вычислений, универсальность вычислительной машины, набор внутренних инструкций, общая конструктивная схема, организация ввода и вывода информации, составление программ, циклы и переменные… Создатель электромеханического компьютера «Mark I» (1941 г.) Говард Айкен рассматривал свою машину как современный вариант машины Бэббиджа, в которой пар и шестеренки заменены электричеством и реле. Составительница комментариев Ада Лавлейс, создававшая своими разъяснениями основы программирования цифровых ЭВМ, названа «первым программистом», в честь неё назван язык программирования вооруженных сил НАТО – «Ада» (1975 г.).

Логика Буля (1854)

В 1854 году Джордж Буль опубликовал работу «Исследование законов мышления, базирующихся на математической логике и теории вероятностей», которая положила начало алгебре логики, или булевой алгебре. Буль первым показал, что существует аналогия между алгебраическими и логическими действиями, так как и те, и другие предполагают лишь два варианта ответов – истина или ложь, нуль или единица. Он придумал систему обозначений и правил, пользуясь которыми можно было закодировать любые высказывания, а затем манипулировать ими как обычными числами. Булева алгебра располагала тремя основными операциями – И, ИЛИ, НЕ, которые позволяли производить сложение, вычитание, умножение, деление и сравнение символов и чисел. Таким образом, Булю удалось окончательно сформулировать основы математической логики и подробно описать двоичную систему счисления. Он также попытался сформулировать общий метод вероятностей, с помощью которого из заданной системы вероятных событий можно было бы определить вероятность последующего события, логически связанного с ними.

Первоначально алгебра была разработана Булем как обычная для того времени алгебра, а не как дедуктивная система в позднейшем смысле. Отсюда и сохранение всех арифметических операций, в том числе вычитания и деления, которые было трудно истолковать

логически. Алгебра логики Буля была значительно упрощена и усовершенствована Джевонсом, отказавшимся от использования операций вычитания и деления. Такая алгебраическая система впоследствии и получила название «булевой алгебры».

В 1937 году Клод Шеннон показал, что существует соответствие между концепциями булевой логики и некоторыми электронными схемами, которые получили название «логические вентили» и стали основой современной цифровой техники.

Статистическая машина Холлерита (1890)

В XIX веке заметной стала необходимость решения новых социально-экономических задач, связанных с обработкой больших объемов информации (прежде всего в сферах учета и статистики). Существовала шутка, почему переписи населения называют десятилетними – потому что обработка результатов занимает 10 лет.

Американец Герман Холлерит построил статистический табулятор (от лат. tabula – доска, таблица) с целью ускорить обработку результатов переписи населения. Возможно он был знаком с идеями Бэббиджа и Жаккарда, но к этой идее его подтолкнула работа кондукторов, отмечающих компостером в специально отведенных местах на билете пол, цвет волос и глаз пассажиров. Табулятор суммировал данные, нанесенные на карту, по 40 сложным комбинациям одновременно и не требовал пересортировки, неизбежной при ручном подсчете.

После успешного использования при переписи 1890 г., когда данные обработались втрое быстрее, сфера применения табуляторов стала расширяться. Машина Холлерита имела большой успех, на ее основе было создано предприятие, которое в 1924 году превратилась в фирму IBM – крупнейшего производителя вычислительной техники.

Машина Холлерита признана первой электромеханической счетной машиной с программным управлением (счет и сортировка осуществлялась под управлением электрических импульсов, возникающих в зависимости от отверстий в перфокартах), хотя часть работы выполнялась вручную (заполнение перфокарт и их подача).

Разумеется, при разработке машины для составления таблиц переписи Холлерит не думал об информатике и глубоких проблемах обработки данных. В то же время его машина не только считала, но и выполняла выборочную сортировку (это уже элемент информационного поиска). А прикладным вкладом Холлерита в вычислительную технику стало создание и усовершенствование целого семейства устройств ввода/вывода. Кроме того, у Холлерита есть чему учиться – он использовал новейшую технику, но подходил к решению проблем с максимальной простотой и легкостью для понимания.

ГЛАВА 3. РАЗВИТИЕ ВЫЧИСЛИТЕЛЬНОЙ ТЕХНИКИ ОТ СПЕЦИАЛИЗИРОВАННЫХ МАШИН К УНИВЕРСАЛЬНЫМ КОМПЬЮТЕРАМ [2.1, 3.1]

§3.1. Основные вычислительные задачи начала XX в.

Астрономические расчеты и навигация

Астрономия и навигация ставили вычислительные задачи с давних времен. Со временем интенсивность и значимость морских сообщений существенно возросла. Тот же Бэббидж, создавая «разностный вычислитель», получил финансирование именно под составление мореходных таблиц (обычно это таблицы умножения, логарифмов, синусов, косинусов, а также всевозможные таблицы результатов астрономических и навигационных измерений и наблюдений). Расширился и круг интересов астрономии, одним из вычислительных достижений которой стало предсказание карликовой планеты Плутона по возмущениям орбиты Нептуна и Урана (конец XIX в.).

Кораблестроение

Наряду с астрономией и навигацией, начиная с XVIII в. особенно нуждались в точных расчетах корабельные науки. Они требовали расчета устойчивости и усилий, возникающих в конструкции корабля при качке, определения «ходкости» и устойчивости корабля под парусами, расчета нагрузок на конструкции корабля от отдачи орудий и т. д. Еще в конце XVIII в. было создано много алгоритмов для решения практических задач, но вычисления по этим алгоритмам были чрезвычайно трудоемки. Рубеж же XIX и XX веков дополнительно ознаменовался «гонкой морских вооружений» (приостановленной лишь Вашингтонскими соглашениями в 1922 г.), когда новейшие броненосцы в момент спуска на воду уже оказывались морально устаревшими. Именно для расчета корпусов судов А.Н. Крылов в 1904 г. создал один из первых аналоговых компьютеров.

Статистика, экономика и бухучет

Первоначально статистика вращалась вокруг потребности государства знать демографические и экономические параметры общества. К XX веку объем дисциплины стал расширяться, и в середине века статистические расчеты уже использовались, например, при создании ядерной бомбы (метод Монте-Карло).

На начало XX века приходятся и попытки математического описания экономики – труды основоположников (Смита, Рикардо) анализируются математическими методами, создаются новые экономические теории (Кейнс). Быстрый рост монополий и увеличение

вмешательства государств в экономику ставят задачу ведения бухгалтерского и налогового учета в огромных масштабах.

Ядерная физика

Создание США ядерной бомбы потребовало проведения значительного объема математических расчетов из-за неясности относительно оптимального способа подрыва бомбы и характера её поражающего фактора (взрывной волны), осложненной нехваткой радиоактивных материалов и отсутствием времени для многократных испытаний. После успешного применения бомбы США задача СССР по созданию собственного оружия существенно упростилась, поскольку остались преимущественно инженерные задачи. Но последующая разработка водородной бомбы вновь потребовала полноценного математического моделирования.

Баллистические расчеты

Попытки описания модели полета снаряда предпринимались со времен изобретения пушек, но из-за неполноты моделей и сложности расчетов баллистические таблицы, позволяющие выбрать угол вертикальной наводки орудия при заданных условиях, долгое время составлялись исключительно путем испытания орудий на полигонах.

К концу XIX века развитие аэродинамики позволило найти достаточно точное математическое описание сил, действующих на тело, движущееся с большой скоростью в воздухе, а также был разработан численный метод интегрирования дифференциальных уравнений, позволявший с заданной точностью решать баллистические уравнения. Оставалась только проблема объема самих вычислений. Для расчета одной траектории было необходимо выполнить минимум 750 операций умножения, на что квалифицированный специалист с арифмометром затрачивал около 3 дней, а для каждой комбинации орудия и снаряда требовалось 2–4 тыс. таких расчетов.

Ярче всего недостатки систем наведения при их применении в новейшем вооружении показала ракета «Фау-2». Оснащенная автономной гироскопической системой управления, приборами для измерения скорости и первым в мире бортовым компьютером, она оказалась неэффективной в применении и производила скорее психологический эффект. Так, в круг диаметром 10 км попала только половина запущенных ракет.

В середине века проблема большого количества вычислений особенно остро встала перед США, которые двигались по пути «пассивного» наведения исходя из определенных входных данных (скорость и направление ветра, масса снаряда и т. д.). В этом способе играет роль скорость и точность расчета, и почти все первые

компьютеры работали над составлением баллистических таблиц. СССР на тот момент пошел по пути «активного» способа наведения, когда в снаряде расположена аналоговая система наведения, корректирующая траекторию уже в воздухе. Поэтому некоторое время использование вычислительных машин считалось совершенно излишним.

Криптография [3.2]

В XIX и начале XX веков несколько факторов способствовали развитию криптографии. Первым фактором были детективные истории, такие как «Золотой жук» Эдгара По или «Пляшущие человечки» Конан Дойля, в которых фигурировали закодированные сообщения и которые волновали воображение многих читателей. Вторым фактором явилось изобретение телеграфа и азбуки Морзе. Азбука Морзе была первым двоичным представлением алфавита, которое получило широкое распространение. Однако в то время сложные шифры применялись не часто, так как требовали много времени и сил для кодирования и декодирования. Также большое влияние оказывали быстрорастущие фондовые биржи, банки и монополии, создававшие спрос на способы конфиденциальной передачи сведений в больших объемах.

В первую мировую войну в ряде стран были разработаны роторные шифровальные машины, которые позволяют легко кодировать и декодировать текст, используя сложный шифр. Сама по себе идея роторного шифрования известна со времен античности, когда был изобретен шифр Цезаря, основанный на замене одной буквы другой буквой по определенному правилу. Одной из первых практически используемых машин, стала немецкая Enigma, разработанная в 1917 году. В модернизированном варианте она была принята на вооружение в Германии (общий тираж около 100 тыс. экз.), а в разных модификациях была доступна для коммерческого применения.

У американцев во время Второй мировой войны была собственная очень мощная версия 15-дискового шифратора Sigaba. Всего было выпущено до 10 тыс. устройств Sigaba, они продержались на вооружении до конца 50-х годов. В Великобритании производился шифратор Typex, собственный аналог Enigma. Работы по созданию механизированных шифраторов велись и в СССР, созданная аппаратура обеспечивала шифрование, считавшееся абсолютно надежным, но выпускалась несравненно малыми тиражами (на момент начала ВОВ на вооружении стояло около 250 комплектов).

Роторные системы позволяли реализовывать очень стойкие шифры. Из-за этого во время второй мировой войны главный метод дешифровки кодов основывался на краже неприятельской шифровальной машины. По-настоящему математического решения, требующего больших объемов вычисления, потребовала расшифровка

системы связи высшего немецкого командования «Лоренц», приведшая к созданию вычислительной машины «Колосс». К 50-м годам стало ясно, что ни один из механических шифраторов не выдерживает атаки, организованной с помощью высокопроизводительного компьютера.

Дальние линии электропередач

В дальних линиях электропередач начинают сказываться индуктивность и емкость провода, ограничивая пропускную способность линий и вызывая параметрическую неустойчивость – самовозбуждение генераторов электростанций. Постройка линий стала требовать предварительного расчета режимов работы сети, особенно при включении их в единую энергосистему или подключении масштабных регионов – потребителей энергии.

§3.2. Аналоговые вычислительные машины

Аналоговая вычислительная машина (АВМ) – устройство, заменяющее значения вычисляемых переменных физическими величинами, ведущими себя аналогично исходным величинам, при этом итоговая физическая величина будет являться ответом и может быть измерена. Обычно АВМ каждому мгновенному значению исходной величины ставит в соответствие мгновенное значение другой величины, часто отличающееся от исходной величины физической природой и масштабным коэффициентом. Каждой элементарной математической операции над машинными величинами, как правило, соответствует некоторый физический закон, устанавливающий математические зависимости между физическими величинами на выходе и входе решающего элемента (например, законы Ома и Кирхгофа для электрических цепей, выражение для эффекта Холла, силы Лоренца и т. д.). В зависимости от используемого физического принципа могут быть механическими, пневматическими, гидравлическими, электромеханическими, электронными.

Несмотря на простоту программирования АВМ и сравнительно большую скорость работы, область применения очень узка из-за ограниченной точности и малой универсальности, поскольку для решения другого класса задач требуется изменять структуру машины. Но исторически аналоговые машины появились раньше цифровых.

К первому аналоговому вычислительному устройству относят обычно логарифмическую линейку. Следующая разновидность аналоговых вычислительных устройств – графики и номограммы. Они впервые встречаются в руководствах по навигации в 1791 г. В 1814 г. был изобретен планиметр, предназначенный для определения площади, ограниченной замкнутой кривой на плоскости, позднее Дж. Томсон изобрел интегрирующий вычислитель.

Дальнейшее развитие механических интегрирующих машин связано с работами В. Буша, под руководством которого была создана механическая интегрирующая машина (1931), а затем её электромеханический вариант (1942). В 1935 г. советский инженер Н. Минорский предложил идею электродинамического аналога. Базовыми устройствами АВМ являются сумматор, интегратор, дифференциатор и усилитель.

АВМ применяются до настоящего времени, причем не только в виде газовых и гидравлических вентилей – их второе рождение связано с изобретением в 1940-е операционных («решающих») усилителей, использующих в качестве аналоговой величины напряжение.

Интегратор Кельвина (1876)

Начало работ над аналоговыми вычислительными машинами можно отнести к концу XIX века, когда Джеймс Томсон разработал планиметр, в котором использовался интегратор с шаром и диском, а его брат, Уильям Томсон (лорд Кельвин), применил этот интегратор в анализаторе гармоник и предсказателе морских приливов, раскладывающем изменения уровня воды по гармоникам. Так интегратор стал первой механической аналоговой вычислительной машиной.

Интегратор Кельвина состоит из стеклянного вращающегося диска и маленького стального колеса, которое ребром скользит и вращается по поверхности стеклянного. Угловая скорость колеса зависит от угловой скорости диска и расстояния x от центра диска до точки касания (рис. 3.1). Если расстояние x меняется как функция времени $x(t)$, то угол колеса в данный момент представляет интеграл функции $x(t)$, который далее может быть передан на стрелочный индикатор, самописец, лучевую трубку или на следующий каскад интеграторов.

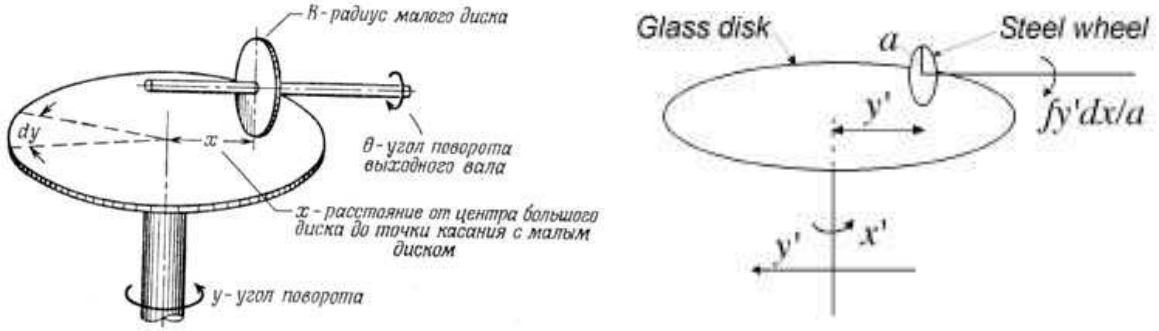


Рис. 3.1. Интегратор Кельвина и его математическая модель

Дифференциальный анализатор Буша (1931) [3.3]

Объединив в одном комплексе несколько интеграторов Кельвина, в 1931 г. Вэнивар Буш создал «решатель дифференциальных уравнений»,

или дифференциальный анализатор, выдававший результаты вычислений в графическом виде и способный решать уравнения вплоть до шестого порядка. Дифференциальный анализатор Буша более десяти лет широко применялся в различных областях, в том числе военной. С его помощью, например, определялись точки, куда нужно навести ствол орудия, чтобы снаряд сбил самолет.

С точки зрения техники дифференциальный анализатор не представлял собой ничего принципиального нового и напоминал большой радиотехнический конструктор. И когда англичане решили пойти по стопам Буша и построить свой анализатор, они воспользовались деталями из обычных детских конструкторов и добились вполне удовлетворительных результатов.

В 1942 г. Буш создал 100-тонный монстр «Дифференциальный анализатор Рокфеллера» с передовым для своего времени вводом информации с перфоленты, проработавший с полной нагрузкой всю войну.

Наблюдение за работой анализатора позволило Винеру, в то время разрабатывающему основы кибернетики, предложить для управления уже цифровую, а не аналоговую машину. Также стоит отметить, что за 4 года до создания анализатора Буш попытался создать машину, «способную самостоятельно думать», но уперся в технологические проблемы. Несмотря на административную занятость, идею не бросил и в 1945 г. опубликовал работу «Как мы можем мыслить», в которой предложил прообраз гипертекстового устройства Memex.

Гельмут Хельцер, «Mischgerät» (1943) [3.4]

Аналоговый интегратор послужил основой и первого бортового компьютера. В 1935 году немецкий инженер Гельмут Хельцер (Helmut Hoelzer, 1912–1996) придумал электронный дифференциатор, аналог интегратора Кельвина. В 1939 г. по приглашению главного конструктора немецкой ракетной техники Вернера фон Брауна он был откомандирован на секретную базу в Пенемюнде, где занимался созданием бортового компьютера «Mischgerät» (нем. «смешанный вычислитель») для ракеты «Фау-2» (V-2, от нем. Vergeltungswaffe – «оружие возмездия»). Это был электронный, но не цифровой, а аналоговый компьютер, предназначенный для решения уравнений баллистики. Помимо адаптации дифференциатора к управлению полетом, Хельцер применил его и для создания симулятора полета.

§3.3. Теоретические основы электронных вычислительных машин [3.5]

Основы построения электронных вычислительных машин в их современном понимании были заложены в 30–40-е годы XX века. Так,

понятие «алгоритма» независимо было разработано математиками А. Тьюрингом (Великобритания), Э. Постом (США) и А.А. Марковым (СССР). Для уточнения этого термина применительно к вычислениям Тьюринг и Пост предложили «абстрактные вычислительные машины». Обе эти работы велись также независимо, машины были предложены в 1936 году (в мае и октябре соответственно), они эквивалентны, но машина Поста отличается большей простотой. Нормальный алгоритм Маркова был предложен в 1940-х и лег в основу логического программирования.

Машина Поста (1936)

Состоит из каретки (считывающая и записывающая головка) и разбитой на секции бесконечно ленты, каждая секция ленты либо пустая (0), либо помечена меткой (1). За шаг каретка может сдвинуться на одну позицию в сторону, считать, поставить или удалить метку в том месте, где она стоит. Работа машины задается программой, команд всего 6 (сдвиг вправо, сдвиг влево, запись метки, удаление метки, условный переход по метке, остановка).

Машина Тьюринга (1936) [3.6]

Состоит из бесконечной ленты, разбитой на ячейки, и управляющего устройства, перемещающегося по ленте, читающего и записывающего в ячейки символы некого алфавита (рис. 3.2). Управляющее устройство способно находиться в одном из множества состояний (число состояний конечно и точно задано), и работает согласно правилам перехода, которые и представляют алгоритм, реализуемый данной машиной.

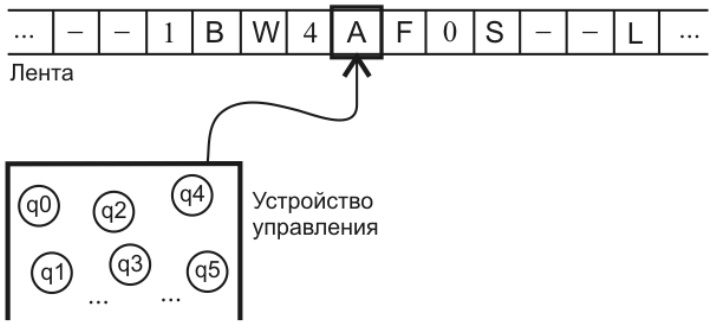


Рис. 3.2. Машина Тьюринга

Каждое правило перехода предписывает машине, в зависимости от текущего состояния q_i и наблюдаемого в текущей клетке символа a_j, записать в эту клетку новый символ a_{j1}, перейти в новое состояние q_{i1} и переместиться на одну клетку влево или вправо. Некоторые состояния машины Тьюринга могут быть помечены как терминальные, и переход в

любое из них означает конец работы, остановку алгоритма. Для начала работы необходимо указать конечное и начальное состояния, начальную конфигурацию на ленте и расположение головки машины. Можно сказать, что машина Тьюринга представляет собой простейшую вычислительную машину с линейной памятью.

Машина Тьюринга является расширением конечного автомата, описывающего пути изменения состояния объекта в зависимости от его текущего состояния и входных данных, при условии, что общее возможное количество состояний конечно. Машина Тьюринга способна имитировать другие исполнители (с помощью задания правил перехода), каким-либо образом реализующие процесс пошагового вычисления, в котором каждый шаг вычисления достаточно элементарен.

От машины Тьюринга происходит и понятие «полноты по Тьюрингу», означающей возможность реализации в вычислителе любой вычислимой функции.

Тьюринг показал, что не существует «чудесной машины», способной решать все математические задачи. Но, продемонстрировав ограниченность возможностей, он на бумаге построил то, что позволяет решать очень многое, и что теперь называется словом «компьютер». Хотя машина Тьюринга не стала реально действующим устройством, она до настоящего времени постоянно используется в качестве основной модели для выяснения сущности таких понятий, как «вычислительный процесс», «алгоритм», а также для выяснения связи между алгоритмом и вычислительными машинами.

Клеточный автомат Неймана (1948) [3.7]

Фон Нейман, привлеченный в 1947 г. к созданию вычислительных машин, стал разрабатывать теорию самовоспроизводящихся автоматов и универсальных вычислительных машин. Такие машины на уровне блок-схемы близки к машине Тьюринга, а их функционирование во многом аналогично процессу воспроизведения живой клетки. Нейман довел эту общую схему до детальной логической конструкции, введя представление о клеточном автомате, который состоит из неограниченного числа повторяющихся конечных автоматов-переключателей, каждый из которых взаимодействует со своими соседями. Такие идеализированные переключатели с задержками были получены из идеализированных нейронов, имели несколько возбуждающих и тормозящих входов, пороговое число и единичную задержку.

В упрощенном виде каждый конечный элементарный автомат на каждом такте пребывает в одном из конечного числа состояний r_i и имеет два входных канала: левый и правый (рис. 3.3). По каждому из них на такте t поступает по одному состоянию из r. Состояния элементов в

момент времени *t* определяют конфигурацию всего автомата. Функционирование автомата Неймана – это переход от состояния *k(t)* к состояниям *k(t+1), k(t+2)* и т.д.

Применение для описания вычислительных машин идеализированных переключающих элементов позволило при конструировании отделить этап логического синтеза от синтеза соответствующих электрических цепей.

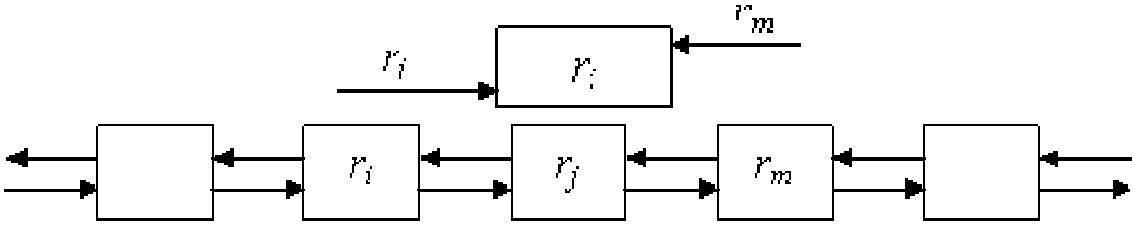


Рис. 3.3. Элементарный конечный автомат (вверху) и автомата Неймана

Также фон Нейман разрабатывал методы автоматического программирования, а совместно с Германом Голдстейном (Herman Goldstine) ввел блок-схемы, облегчающие переход от математического описания вычислений к программе, составленной в машинных кодах.

§3.4. Электромеханические вычислительные машины

В начале XX века на смену зубчатым колёсам приходят электрические элементы и первый из них – реле. В 30-е годы XX века на базе реле происходит развитие и совершенствование счетно-аналитической техники. Наряду с табуляторами фирма IBM начинает серийный выпуск множительных перфораторов (для сложения, вычитания и умножения) и вычислительных перфораторов (для выполнения четырех арифметических действий). Разрабатываются ленточные перфораторы, вводные устройства для автоматической записи показаний различных приборов, итоговые перфораторы и т. п. Во второй половине 30-х годов в Германии и США начинается работа над проектами универсальных вычислительных машин с программным управлением для выполнения сложных расчетов. Первая такая машина, Z3, была создана германским инженером Конрадом Цузе в 1941 г. (работа над проектами автоматических вычислительных машин велась с 1935 г.). В 1939 г. Г. Айкен (Гарвардский университет, США) возглавил работу над проектом Mark-1 и в 1944 г. завершил разработку машины. С 1938 г. работу над автоматическими цифровыми машинами на контактных реле ведет Дж. Стибиц (фирма «Bell», США). Результатом работ явилось создание в 1939–1944 гг. нескольких специализированных машин и мощной универсальной машины Bell-V, которая была закончена в 1946 г., уже после постройки первого компьютера.

С одной стороны, релейные машины имели малую надёжность из-за самих реле, в которых были механические трущиеся и поворачивающиеся части, из-за чего они быстро ломались. С другой стороны, несмотря на ненадежность и большой шум, они работали и очень даже неплохо и были вытеснены лишь к 60-м. Например, РВМ-1, сконструированная в СССР в середине пятидесятых годов, могла выполнять 1250 умножений в минуту и состояла из пяти с половиной тысяч реле.

С началом второй мировой войны правительства разных стран начали разрабатывать вычислительные машины, осознавая их стратегическую роль в ведении войны. Увеличение финансирования в значительной степени стимулировало развитие вычислительной техники. В 1941 году Цузе разработал вычислительную машину Z2, выполнявшую расчеты, необходимые при проектировании самолетов и баллистических снарядов. В 1943 году английские инженеры завершили создание вычислительной машины для дешифровки сообщений немецкой армии, названной «Колосс». Однако эти устройства не были универсальными вычислительными машинами, они предназначались для решения конкретных задач.

Конрад Цузе, Z1-Z3 (1938–1941) [3.8, 3.9]

Конрад Цузе (Konrad Zuse, 1910–1995) работал на самолетостроительном заводе Хеншеля (Henschel) и каждый раз, встречаясь с друзьями, жаловался, говоря что работает со статистикой и делает в день десятки тысяч однообразных вычислений. Он задумывался о машине, которую нужно было бы только запрограммировать на определенную проблему, а дальше она работала бы сама.

Размышляя о такой машине, Цузе достаточно четко и раньше других сформулировал саму идею хранимой в памяти программ. Вот что он пишет в заявке на патент от января 1936 г., которого так и не получил: «В представленном изобретении механические реле комбинируются с последовательной логической системой, где могут храниться произвольные спецификации, например числа. Такие возможности, особенно в области вычислительных машин (Rechenmaschinen), играют определенную роль. Они также могут быть использованы для хранения других спецификаций, в частности инструкций работающих машин (релейная память), комбинаций букв (телеграфная память), алфавитного кодирования (шифровальные машины) или подобных вещей».

В 1938 г. Цузе создал механическую вычислительную машину Z1, которая явилась первым программируемым компьютером с булевой логикой и двоичной арифметикой с плавающей запятой (рис. 3.4). В 1940 году он получил поддержку Исследовательского института аэродинамики, который использовал его работу для создания

управляемых ракет. Благодаря ей в 1941 г. Цузе построил доработанную версию вычислителя – Z2, на основе телефонных реле. В том же 1941 году Цузе создал еще более совершенную модель Z3 (рис. 3.4), которую сегодня многие считают первым реально действовавшим программируемым компьютером. Впрочем, программируемость этого двоичного вычислителя, собранного, как и предыдущая модель, на основе телефонных реле, также была ограниченной. Несмотря на то, что порядок вычислений теперь можно было определять заранее, условные переходы и циклы отсутствовали. Тем не менее, Z3 первым среди вычислительных машин Цузе получил практическое применение и использовался для проектирования крыла самолёта.

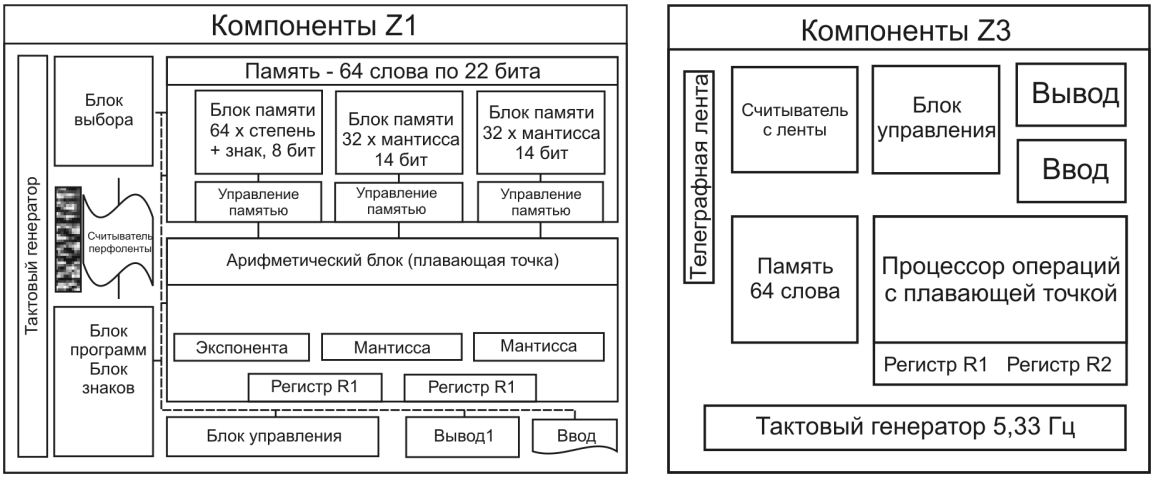


Рис. 3.4. Компоненты компьютеров Z1 и Z3

Основными параметрами Z3 были – тактовая частота 5,3 Гц, возможность выполнения четырех арифметических действий и извлечения корня, оперирование со словами длиной 22 бита. Сложение и умножение выполнялось за 0,8 и 3 секунды. Машина состояла из 2600 реле (из которых 600 в арифметическом блоке и 2000 в памяти), потребляла 4 кВт и весила около тонны. Программа вводилась с помощью телеграфной ленты. Сама машина была Тьюринг-полной, хотя и не полностью электронной.

О работах Цузе в предвоенные годы стало известно только в 1972 г., когда вышла его работа «Der Plankalkül», посвященная первому языку программирования Plankalkül (1945 г.).

Harvard Mark I (1941) [3.10]

В 1937 году гарвардский математик Говард Айкен (Howard Aiken), впечатленный разностной машиной Бэббиджа, предложил проект создания большой счетной машины на электромеханических реле. Спонсировал работу президент компании IBM Томас Уотсон (Tomas

Watson), который вложил в нее 500 тыс. долларов. В то время компания IBM начала делать первые шаги к созданию компьютеров и уже изобрела полностью электронный множитель на вакуумных лампах. Но Эйкен в качестве элементной базы выбрал реле, проигнорировав громадное увеличение скорости вычислений, достижимое с помощью электронных ламп. Ошибочность этого решения позднее стала очевидна, в частности благодаря самому Эйкену. По мнению Н. Винера, «тогда у него была странная причуда, заставлявшая его считать работу с механическим реле нравственной и разумной, а использование электронных реле – делом, никому не нужным и морально нечистоплотным» [3.11].

Проектирование Mark I началось в 1939 г., постройка завершилась в 1941 г., а в 1944 машина была официально передана Гарвардскому университету. Компьютер содержал около 750 тыс. деталей, 3300 реле, более 800 км проводов, весил около 4,5 тонн, а синхронизация модулей осуществлялась механически 15-метровым валом.

Из-за использования электромеханических реле Mark I был довольно медленной машиной, выполняя сложение за 3 секунды, умножение – за 6, и деление – за 15 секунд. Фактически он представлял собой большой арифмометр, заменявший труд примерно 20 операторов. Однако он управлялся с помощью программы, которая вводилась с перфоленты, что отражалось в названии, данном IBM – «Automatic Sequence Controlled Calculator». Это дало возможность, меняя вводимую программу, решать довольно широкий класс математических задач.

Mark I имел 60 наборов 24-рычажных переключателей для ручного ввода данных и мог хранить 72 слова, каждый в виде 23-значных десятичных чисел. Формат команд следующий – 24 канала входной ленты делились на 3 поля по 8 каналов. Все аккумуляторы, наборы переключателей, регистры ввода/вывода и арифметических устройств имели свой уникальный номер. Первое поле было кодом приемника данных, второе поле – источника данных. Третье поле было кодом операции. По итогам испытания Mark I было построено еще несколько экземпляров Mark II, производительность которых была выше в пять раз. В 1952 г. была выпущена полностью электронная модель Mark IV.

Машина применялась в военных целях для расчета артиллерийских таблиц, также благодаря ей были расшифрованы секретные коды, использовавшиеся в радиопередачах немецкой армии. Примененное в машине раздельное размещение программы (на перфоленте) и данных (на релейных счетчиках) дало название «гарвардской» архитектуре процессоров.

§3.5. Электронные вычислительные машины

Атанасов и Берри, ABC (1941) [3.12]

Первый проект электронной цифровой вычислительной машины был разработан Джоном Атанасовым (США), который в 1937 году сформулировал, а в 1939 году опубликовал окончательный вариант своей концепции современной машины. Машина предназначалась для решения систем линейных уравнений.

Вместо механических устройств Атанасов решил применять электронные переключатели, с помощью которых должны были выполняться функции управления и арифметические операции. В этом смысле ему принадлежит первенство. До этого ни одна машина, предназначенная для решения сложных математических задач, не была основана на электронике. Он пришел также к убеждению, что его цифровая машина должна оперировать двоичными числами и что операции над этими числами будут осуществляться в соответствии с правилами логики, а не прямым подсчетом. Атанасов решил и важную частную проблему, касающуюся применения в качестве запоминающих элементов компьютера конденсаторов. Проблема заключалась в том, что конденсаторы постепенно теряют свой заряд, и Атанасов обошел это затруднение, применив периодическую регенерацию заряда.

Постройка машины велась Атанасовым и его единственным помощником, аспирантом Клифордом Берри в 1939–1941 гг., машина получила название ABC (Atanasoff Berry Computer). На завершающем этапе разработчики столкнулись с проблемой в системе ввода-вывода на перфокартах, которая стала ошибаться приблизительно раз на каждые 10 тыс. операций чтения или перфорирования. Они пытались решить эту относительно несложную техническую проблему, но начавшаяся вторая мировая война заставила их бросить работу над компьютером. После войны Атанасов остался на флоте и к компьютерам не вернулся.

Поначалу вклад Атанасова в развитие вычислительной техники не ценился, хотя Мочли, будущий создатель ENIAC, в 1941 г. был в гостях у Атанасова и был ознакомлен с ABC. Эта история всплыла 30 лет спустя, когда в 1971 г. одна фирма опротестовала патент на ENIAC. Истцам удалось доказать, что идеи ENIAC были заимствованы от ранее созданного ABC.

Colossus (1943–1944) [3.13]

Предшественником проекта Colossus в дешифрации является разработанная при участии Тьюринга «Бомба» (the Bombe) которая позволила с середины 1940 года расшифровывать все кодированные сообщения Люфтваффе.

Тогда же, в 1940 г., впервые были перехвачены необычные

сигналы, передающиеся в виде шифрованных кодов Бодо. Позднее выяснилось, что это сообщения системы связи высшего командования Вермахта «Lorenz», зашифрованные ключом сложением по модулю 2. После перехвата в 1941 г. ошибочного повторного сообщения была разработана математическая теория декодирования и начата работа по созданию механических средств декодирования. Механический перебор отличался ненадежностью, а реализовать синхронизацию перфолент, необходимую для сравнения двух сообщений, на скорости 1000 символов в секунду было невозможно. Инженер Томми Флауэрс предложил использовать электронные лампы и возглавил проект по созданию машины на их основе , названный Colossus.

Проектирование новой машины началось летом 1943 года, а уже в январе 1944-го Mk 1 Colossus был запущен в эксплуатацию. Он состоял из 1500 ламповых триодов, тиратронов и фотоумножителей, размещенных в пяти стойках общей длиной 5,5 метров и более 2 метров высотой. В стойках были смонтированы тиратронные цепи, эмулировавшие вращение дисков Lorenz. Архитектура Colossus позволяла обрабатывать перфоленту с шифрованным сообщением со скоростью 5 тыс. символов в секунду, и на один цикл обработки среднего сообщения уходило не более секунды. В нем впервые появились регистры сдвига и систолические матрицы, обеспечивающие параллельную обработку 5 каналов ленты. Также машины допускали параллельный режим работы. Это позволило оперативно получать сведения о решениях верховного командования Вермахта, хотя поначалу мало кто верил, что при таком количестве ламп удастся сохранить надежность устройства.

Colossus был построен на цифровых электронных схемах, но возможность перепрограммирования была частичной и ограничивалась имитацией механических дисков Lorenz. Т.е. машина была специализированной, и не обладала полнотой по Тьюрингу. Кроме того, всего было создано 10 таких машин, которые после войны были уничтожены, а сам проект засекречен. Информация о Colossus стала доступна лишь в 1970-х гг. и не оказала существенного влияния на ход разработки первых компьютеров.

ENIAC (1945) [3.17, 3.14]

В 1940 г. Джон Мочли (John W. Mauchly) знакомится с работой Атанасова над ABC, публикует небольшую пятистраничную записку под названием «Использование устройств на вакуумных лампах для вычислений», а в 1942 г. совместно с Джоном Эккертом (John P. Eckert) предлагают свой проект ENIAC (Electronical Numerical Integrator and Calculator). В 1943 г. возможностями выполнения расчетов на ENIAC заинтересовалась Баллистическая исследовательская лаборатория Армии

США (Ballistic Research Laboratory, BRL), и в том же году под руководством Мочли и Эккерта началась постройка машины. В 1945 г. машина ENIAC была введена в действие, а в феврале 1946 г. проект был рассекречен, и состоялась первая публичная демонстрация. В ENIAC'е электромеханические реле были заменены на электронные вакуумные лампы, что дало прирост в скорости примерно в 1000 раз.

ENIAC весил 30 тонн, состоял из 18 тысяч электронных ламп, 1500 реле и десятков тысяч прочих радиоэлементов. Количество ламп было достаточно, чтобы скептики могли предсказать, что она не будет нормально функционировать. Производительность была 5000 операций сложения или 385 операций умножения в секунду. Общая стоимость машины составила $750 тыс., потребляемая мощность около 200 кВт, вес 27 тонн, а занимаемое пространство – 63 м^2. Для сравнения, Mark 1 весил 4,5 тонны, стоил $500 тыс., потреблял 4 кВт, и занимал 10 м^2.

ENIAC оперировала 10-разрядными десятичными числами, хранящимися в 20 аккумуляторных модулях. Для хранения одного десятичного разряда требовалось 36 ламп, 10 из которых образовывали кольцевой регистр триггеров (двойных триодов). Во время вычислений подсчитывались импульсы, генерируемые при «прокручивании» регистра, эмулировавшего поворот колеса арифмометра. Машинный цикл состоял из 20 машинных тактов, поэтому на частоте 100 кГц производительность составляла 5000 простых операций в секунду.

Ввод чисел в машину производился с помощью перфокарт, а программное управление последовательностью выполнения операций осуществлялось с помощью 6000 переключателей и набора соединительных кабелей. Такой способ программирования занимал до двух дней, но позволял реализовывать счетные способности ENIAC'а и тем выгодно отличался от способа программной перфоленты, характерного для релейных машин.

ENIAC стал работающим прообразом современного компьютера. Во-первых, ENIAC был полностью электронным, во-вторых, полностью был основан цифровой обработке информации, в-третьих, ENIAC стал действительно универсальной вычислительной машиной. Первоначально задуманный для статистического предсказания погоды, он создавался для расчета баллистических таблиц, а после создания решал также задачи, связанные с разработкой ядерной и водородной бомбы, использовался в космических исследованиях и других областях. Его машинное время часто предоставлялось сторонним университетам и исследовательским центрам для тех или иных научных расчетов.

Основной период эксплуатации пришелся на период 1948–1955 гг. (в 1946 и 1947 годах ENIAC работал крайне нерегулярно, постоянно выходил из строя), затем ENIAC был демонтирован, т.к. был очень дорог в эксплуатации, гораздо дороже, чем новые компьютеры ORDVAK или

EDVAC. По воспоминаниям Эккерта, перегорание ламп случалось примерно раз в два дня, а самый длительный бессбойный период работы не превысил 5 дней.

ENIAC радикально отличался не только от предшествовавших ему вычислительных машин, но также сильно отличался и от машин, созданных после него. От последних он отличался использованием нескольких полуавтономных вычислительных узлов, работающих одновременно и полунезависимо, и исключительным применением электронных ламп для оперативной памяти. Последующие модели строились по иным архитектурным принципам, а вместо электронных ламп для оперативной памяти стали применять ртутные линии задержки и электронно-лучевые трубки, что позволило существенно увеличить объем памяти, а также уменьшить число используемых в машинах ламп.

Сравнение производительности

Для сравнения производительности первых вычислительных машин приведем следующие данные. При расчете баллистических таблиц расчет траектории, занимающий у инженера 20 часов, дифференциальный анализатор Буша выполнял за 15 минут, а ENIAC – за 30 секунд. При этом дифференциальный анализатор давал приблизительные результаты, для уточнения которых привлекались десятки людей, работавших с обычными арифмометрами.

Отечественные компьютеры [3.15]

Появление компьютеров в СССР связано с академиком С.А. Лебедевым. Разрабатывая в годы войны аналоговые логические элементы для систем стабилизации стволов и торпед, в 1945 г. Лебедев создал первую в СССР аналоговую электронную вычислительную машину для решения систем дифференциальных уравнений. В 1950 г. был создан МЭСМ – Макет Электронно-Счетной Сашины (или «Малая ЭСМ»), ставшей первым в континентальной Европе компьютером с поддержкой концепции хранимой программы. МЭСМ применялась для решения задач из области термоядерных процессов, космических полетов, ракетной техники и дальних линий электропередачи. В 1953 г. была выпущена БЭСМ – Быстродействующая (или Большая) Электронно-Счетная Машина, которая в год выпуска оказалась самой быстродействующей в Европе.

По некоторым свидетельствам, двоичные вычисления интересовали Лебедева еще до войны, и если бы не она – то работа над созданием вычислительной машины с использованием двоичной системы счисления началась бы раньше.

§3.6. Предварительный доклад по EDVAC (1944) [3.16]

С ENIAC связано имя фон Неймана. В конце 1944 г. Голдстейн (заместитель руководителя BRL, обративший внимание на проект Мочли-Эккерта) случайно встретил на вокзале своего коллегу Неймана. Тот консультировал программу по созданию атомной бомбы и искал пути повышения счетных мощностей для решения дифференциальных уравнений. Обнаружив схожесть задач, Голдстейн пригласил Неймана в рабочую группу ENIAC, которая по заказу Пентагона разрабатывала проект вычислительной машины EDVAC (Electronic Discrete Variable Automatic Computer). Машина задумывалась исключительно на электронных схемах, а программы, вводимые с помощью перфокарт, предполагалось кодировать в виде пригодных для обработки символов и сохранять в централизованной памяти.

В июле 1945 года Нейман в 100-страничном «Предварительном докладе о машине EDVAC» единолично опубликовал идеи разработчиков, ранее изложенные ими в закрытом докладе. Несмотря на споры об авторстве, вклад фон Неймана не стоит недооценивать, т.к. его известность и авторитет, ясная систематизация, обобщение и анализ сделали открытие достоянием разработчиков из разных стран. Именно с момента публикации доклада компьютер был признан объектом, представлявшим научный интерес, а принцип совместного хранения в памяти инструкций и данных, совместно с другими рекомендациями, получил название «архитектура фон Неймана».

Из рекомендаций доклада, опубликованного фон Нейманом, обычно выделяют несколько следующих положений:

- Машины на электронных элементах должны работать не в десятичной, а в двоичной системе счисления.
- программа, исходные данные и промежуточные константы должны иметь одинаковое представление и размещаться в одном устройстве памяти;
- трудности физической реализации устройства памяти, быстродействие которого соответствовало бы скорости работы логических схем, требует иерархической организации памяти;
- арифметические устройства машины конструируются на основе схем, выполняющих операцию сложения;
- операции над числами производятся одновременно по всем разрядам;
- компьютер состоит из процессора, памяти и внешних устройств;
- процессор исполняет программу команда за командой в соответствии с изменением содержимого счетчика команд;
- обработка информации происходит только в регистрах процессора, информация в который поступает из памяти или от внешнего устройства.

Изложенные фон Нейманом основы архитектуры вычислительных устройств оказались настолько фундаментальными, что подавляющее большинство вычислительных машин на сегодняшний день – фон-неймановские машины, а первые поколения компьютеров выпускались практически в полном соответствии с рекомендациями доклада.

В следующие несколько лет в рамках «архитектуры фон Неймана» появился ряд новых компьютеров. Так, в 1949 г. был создан и запущен в производство EDVAC, который в отличие от ENIAC оперировал с двоичной арифметикой и хранил программу в машинной памяти. Эккертом и Моучли в 1949 г. был выпущен BINAC (Binary Automatic Computer), первый коммерческий цифровой компьютер с хранимой в памяти программой, а в 1951 г. ими же был выпущен и поставлен в бюро переписи США десятичный UNIVAC (Universal Automatic Computer). UNIVAC завершил перечень эксклюзивных счетных машин, которые строились в незначительном количестве по заказу отдельных организаций. Производство компьютеров становилось массовым.

§3.7. «Первый» компьютер [3.17]

Единого мнения относительно того, что считать «первым» компьютером, нет. «Первыми» могут считаться больше десятка моделей:

- Ткацкий станок Жаккарда (1801) – первая не счетная программируемая машина;
- Analytical Engine Бэббиджа (1834) – первый программируемый компьютер (спроектированный);
- Машина Тьюринга (1936) – первая теоретическая модель вычислительной машины, работающей по алгоритму;
- Z3 Цузе (1941) – первый программируемый цифровой компьютер (работающий);
- ABC Атанасова и Берри (1942) – первый специализированный электронный цифровой компьютер;
- Mischgerät Хельцера (1943) – первый бортовой компьютер;
- Colossus Флауэрса (1943) – первый электронный компьютер (в Великобритании);
- Harvard Mark I Айкена (1944) – первый программируемый компьютер (в США);
- ENIAC Эккерта и Мочли (1946) – первый электронный программируемый компьютер (США);
- Manchester SSEM Baby Килбурна (1948) – первый компьютер с хранимой программой (Великобритания);
- BINAC Эккерта и Мочли (1949) – первый компьютер с хранимой программой (в США);
- МЭСМ Лебедева (1950) – первый компьютер с хранимой программой (в СССР и континентальной Европе).

Причин путаницы в приоритете и датах создания компьютеров сразу несколько. Во-первых, какой момент считать датой создания компьютера: когда возникла идея, когда она была опубликована, когда что-то из задуманного начало работать, или когда компьютер был сдан в опытную эксплуатацию? Во-вторых, о каком компьютере идет речь: аналоговом или цифровом, (электро)механическом или электронном, специализированном или универсальном? В-третьих, что же вообще понимать под компьютером? Должен ли он отвечать определенным критериям и где проходит грань между изощренным калькулятором, вычислительной машиной и компьютером?

Эта путаница на уровне названий заметна во многих странах. В СССР долгое время употребляли термины «электронная счетная машина», «электронная вычислительная машина» (ЭВМ), а с начала 1980-х годов их вытесняет термин «компьютер». В 1930–1940-е годы слово «computer» обычно означало человека, производящего вычисления, а также любой тип машины, механизирующей вычисления. В фундаментальной работе Алана Тьюринга «О вычислимых числах» (1936) слово «компьютер» используется исключительно для обозначения человека, а в 1950 г. Тьюринг уже употребляет для ясности термины human computer и digital computer. После 1945 г. термин «компьютер» стал употребляться все чаще, причем практически всегда в смысле автоматического электронного цифрового компьютера с внутренней памятью для программ (automatic electronic digital computer with internal program storage).

«Первым» компьютером было решено считать ENIAC, а вычислительные машины его эпохи позднее были отнесены к первому поколению компьютеров.

ГЛАВА 4. РАЗВИТИЕ ЭЛЕМЕНТНОЙ БАЗЫ, АРХИТЕКТУРЫ И СТРУКТУРЫ КОМПЬЮТЕРОВ

§4.1. Реле, лампы, транзисторы

Выше уже упоминались реле и лампы, на основе которых строились первые вычислительные машины. Для лучшего понимания принципов функционирования цифровой вычислительной техники рассмотрим её элементную базу подробнее, охватив период до интегральных схем включительно.

Реле – это электромагнитный переключатель, в простейшем случае состоящий из пластинки-якоря, замыкающего электрическую цепь и переключаемого электромагнитом. Реле может быть ключом, двоичной ячейкой памяти, или использоваться для счета в виде позиционного шагового двигателя.

Электронные лампы

В 1883 году Эдисон пытался увеличить срок службы осветительной лампы с угольной нитью накаливания. Он ввёл в баллон лампы металлический электрод, и открыл явление термоэлектронной эмиссии. В 1905 году лампа со вторым электродом была запатентована как диод, а в 1906 году американский инженер Ли де Форест добавил в лампу третий электрод – управляющую сетку и, таким образом, создал триод (рис. 4.1).

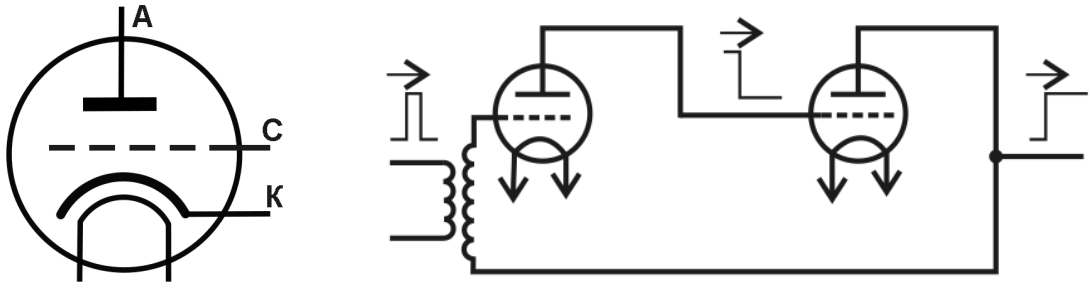


Рис. 4.1. Схема лампового триода и лампового триггера

Триггер на электронных лампах был изобретен независимо в 1918 г. М.А. Бонч-Бруевичем и в 1919 г. У. Икклсом и Ф. Джорданом (США), он состоял из двух последовательно включенных ламп и положительной обратной связи (рис. 4.1). Благодаря этому он находился только в двух стабильных состояниях: включенном и выключенном. Триггер похож этим на реле, но переключается гораздо быстрее. Настолько, что первая вычислительная машина на триггерах (ENIAC) работала в сотни раз быстрее релейной. С триггеров и начинается история развития электронных компьютеров, в которых не было громоздких колёс, и которые перешли на двоичную систему счисления.

В 20–30-е годы улучшаются характеристики электронных ламп, создаются новые типы ламп (тетроды, пентоды, комбинированные лампы и т.п.), развивается теория электронных цепей. В 30-е годы зарождаются телевидение и радиолокация, развивается электронная контрольно-измерительная техника. Электронные лампы впервые начинают применяться для выполнения счетных операций (в приборах ядерной физики для счета заряженных частиц). Первые электронные счетчики для данных целей были разработаны в 1930–1931 гг. Уинн-Уильямсом (Великобритания).

Пик расцвета ламповой схемотехники пришёлся на 1935–1950 гг. Создание действующей электронной счетной машины стало возможным только после того, как была проделана большая работа по повышению надежности ламп и получен опыт проектирования устройств с большим числом ламп.

Ламповые триггеры работали на частотах до десятков мегагерц, на больших частотах выгоднее стало применять транзисторы. Теоретически границы для ламп находятся около гигагерца, и основные ограничения связаны с временем пролета электрона между электродами, паразитными индуктивностями и емкостями.

Транзисторы

В 1947 году сотрудники компании «Белл Телефон» Бардин, Браттейн и Шокли представили действующий образец «полупроводникового триода», в котором три металлических «усика» контактировали с бруском из поликристаллического германия. Устройство получило название «транзистор» (в пер. с английского – «преобразователь сопротивления»).

В 1951 г. Шокли продемонстрировал миру первый надежный биполярный транзистор, представлявший собой трехслойный германиевый «сэндвич» толщиной около 1 см, заключенный в металлический корпус. В этой модели тонкий слой полупроводника p-типа (база) зажат между двумя слоями полупроводника n-типа (эмиттер и коллектор). Прикладывая к базе невысокое управляющее напряжение, можно было переключать или усиливать ток в основной цепи.

В середине 50-х годов стоимость производства транзисторов резко снизилась, в том числе потому, что их стали изготавливать из более дешевого кремния вместо германия. В отличие от германия кремний – самый распространенный на Земле (после кислорода) химический элемент.

Помимо описанных выше биполярных транзисторов существуют и полевые транзисторы, они были запатентованы в США еще в 1920-е гг. Несмотря на более простой физический принцип работы, практически реализовать эти идеи не удавалось до 1947 г., когда та же группа во главе

с Шокли получила работающий прибор. По технологическим причинам полевые транзисторы попали в производство лишь в 1970-х гг., а востребованными оказались к 1980-м. В полевых транзисторах электрическое поле, возникающее под затвором при подаче входного напряжения, влияет на концентрацию носителей заряда в канале, меняя его сопротивление, а значит и ток между истоком и стоком. Самый распространенный вид полевого транзистора – с металлическим затвором и изолятором из оксида кремния между затвором и каналом, или МОП-транзистор (металл-окисел-полупроводник, MOS).

Частотные пределы для транзисторов общего назначения составляют сотни мегагерц, для высокочастотных – около 10 ГГц. Изготовлении транзистора по более совершенным технологиям, например при размещении элементов по вертикали, и с применением других материалов (например арсенида галлия), граница частоты сдвигается вверх. Так, в 2002 г. компания IBM сообщила о достижении частоты 110 ГГц германий-кремниевым транзистором. А в 2008 году эта же компания сообщила о создании графенового транзистора, частота переключения которого составила 26 ГГц, и теоретически может достичь терагерца.

§4.2. Интегральные схемы

Интегральная схема (ИС) это электронная схема произвольной сложности, изготовленная на полупроводниковом кристалле (или плёнке) и помещённая в неразборный корпус. Первые интегральные схемы были продемонстрированные Дж. Килби в 1958 г. и Р. Нойсом в 1959 г. Патент Килби описывал ИС как «полупроводниковый материал, в который интегрированы элементы и связи между ними», т.е. то, что и принято называть микросхемами, но поначалу стали выпускаться микросборки – блоки дискретных элементов в миниатюрном исполнении. В частности IBM, назвавшая серию System/360 компьютерами нового «поколения микросхем», выбрала в качестве элементной базы именно микросборки, как более проверенную технологию.

Интегральные схемы, в зависимости от рассматриваемого признака, можно разделить на группы:

- по способу изготовления – гибридные, пленочные, полупроводниковые;
- по используемой логике – в настоящее время это КМОП и ТТЛ;
- по степени интеграции – от десятков до миллиардов элементов;
- по виду обрабатываемого сигнала – аналоговые, цифровые.

Появление пленочной технологии позволило в едином технологическом цикле выполнять элементы ИС (проводники,

резисторы, конденсаторы) на поверхности подложки в виде пленок из проводящих, резистивных и диэлектрических материалов. В случае, когда определенные элементы выполнялись в виде навесного монтажа, схемы назывались гибридными. В частности, гибридными были все пленочные ИС с транзисторами.

Полупроводниковая технология изготовления ИС позволила изготавливать все элементы в толще полупроводника (кремний, германий, арсенид галлия), выполняя поочередное напылением слоев и внедрение примесей в приповерхностные слои (рис. 4.2). Основными технологиями изготовления полупроводниковых ИС являются литография, диффузия, эпитаксия, ионная имплантация, «кремний на диэлектрике»/«кремний в диэлектрике», окисление кремния и металлизация [4.1].

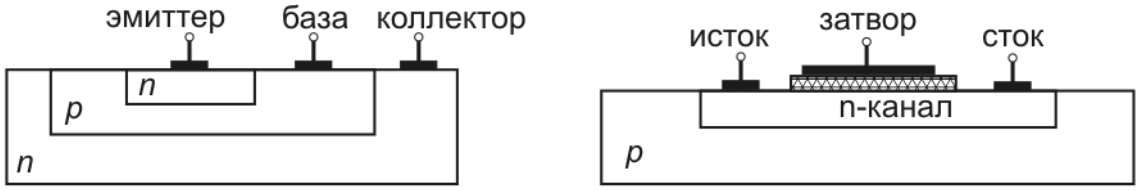


Рис. 4.2. Транзисторы в полупроводниковой подложке микросхемы: биполярный npn и полевой с каналом n-типа

В цифровых ИС используется два основных вида логики – транзисторно-транзисторная (ТТЛ) и комплементарный металл-оксид-полупроводник (КМОП) (рис. 4.3).

ТТЛ была разработана в 1961 г. в связи с появлением полупроводниковых ИС, и явилась развитием диодно-транзисторной логики. Для выполнения логических функций в ТТЛ на входах использовались многоэмиттерные биполярные транзисторы, а не диоды, транзисторы усиливали и выходной сигнал. ТТЛ получила широкое распространение и применялась до частот в десятки мегагерц.

Технологию КМОП в 1963 г. изобрёл Фрэнк Вонлас из Fairchild Semiconductor, а первые КМОП-микросхемы появились в 1968 г. Долгое время КМОП рассматривалась как энергосберегающая, но медленная альтернатива ТТЛ. Со временем технология комплементарных пар оказалась самой быстродействующей из существующих. Т.к. ТТЛ давно стала промышленным стандартом, то, для совместимости, в КМОП схемы часто устанавливаются ТТЛ буферы [4.2].

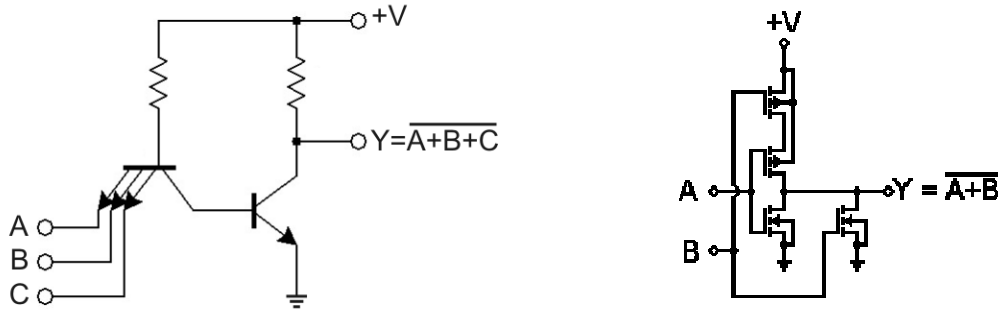


Рис. 4.3. Схемы логических элементов ИЛИ-НЕ, выполненные по технологиям ТТЛ и КМОП (иллюстрация из [4.3])

Современная цифровая техника построена, в основном, по полупроводниковой технологии на КМОП-транзисторах, которые в 1990-е гг. вытеснили биполярные транзисторы из цифровой техники. Основой самых быстродействующих микропроцессоров сегодня является технология «кремний на диэлектрике» (SOI). Что касается быстродействия, то, в отличие от транзисторов, для которых основной параметр частота, для ИС не менее важным параметром является длина затвора, задающая плотность компоновки элементов. Поскольку изготовить быстрый транзистор недостаточно – требуется еще уместить их на небольшой площади. Длина затвора транзистора с традиционной планарной МОП-структурой от 10 мкм в 70-х годах до 130 нм, достигнутых к 2001 г., эволюционировала путем простого масштабирования, т.е. уменьшения длины затвора, толщины диэлектрика и глубины залегания p-n-переходов. Дальнейший прогресс связан с переходом от использования света к электронно-лучевой литографии, переходом на новые материалы (напряженный кремний, германий, индий, арсенид галлия) отказ от «плоскостной» структуры транзистора. В 2008 г. характеристический размер транзистора достиг 45 нм, в 2009 были представлены микросхемы, выполненные по технологии 22 нм.

При линейной экстраполяции существующих технологий один из «пределов» настанет в 2020 г., когда размеры элементов уровня, на котором их поведение будет описываться не электроникой, а квантовой физикой. Следующим фундаментальным пределом для электроники может считаться точность определения положения электрона в пространстве, которой является комптоновская длина волны:

$\lambda_c = h/m_e c = 2{,}4\times10^{-12}$ м = 0,002 нм, где h – постоянная Планка, m_e – масса электрона, c – скорость света.

Дата достижения этого «предела» попадает на 2030-е гг. [4.4].

§4.3. Квантово-размерные структуры [4.5]

В последнее время в микро- и оптоэлектронике все шире используются структуры, в которых проявляется эффект размерного

квантования – или квантово-размерные структуры. Это связано как с успехами в области квантовой теории твердого тела, так и с технологическим прогрессом – молекулярно-лучевая эпитаксия, нанолитография, явление самоорганизации и др. позволяют создавать структуры любого профиля с толщиной до одного атомного слоя. Если один из размеров такой структуры не превышает дебройлевскую длину волны используемых частиц (электронов, дырок, фотонов), то энергетический спектр по одному из квантовых чисел из непрерывного становится дискретным.

Такую систему, в которой движение частиц ограничено, называют квантовой ямой (quantum wells). Примерами систем с ограничением по одному направлению являются тонкие пленки (в т.ч. графен), гетеропереходы (контакт двух полупроводников с различной шириной запрещенных зон), системы с ограничением по двум направлениям – проводящие нити (квантовые нити), с ограничением по трем направлениям – кристаллы малого размера (квантовые точки).

Дальнейшим развитием одиночной ямы являются структуры с целым набором изолированных квантовых ям (multiple quantum wells), в которых электронные возбуждения в различных ямах связаны через электромагнитное поле, что существенно влияет на электрические и оптические свойства структуры. С уменьшением ширины барьеров между ямами периодическая структура изолированных квантовых ям превращается в тонкобарьерную сверхрешетку – периодическую полупроводниковую структуру из тонких чередующихся слоев полупроводника, в которых проявляются и туннельные явления.

Такие двумерные структуры, как полупроводниковые квантовые ямы и сверхрешетки, играют все более важную роль в современной электронике. На их основе создаются разнообразные приборы – лазеры, светодиоды, фотодиоды, диоды и транзисторы, электрооптические модуляторы и приключающие устройства, а также оптоэлектронные интегральные схемы.

Квантовые точки применяются в качестве активной среды в полупроводниковых лазерах, используются в качестве кубитов в моделях квантовых компьютеров, служат рабочим элементом в одноэлектронных транзисторах, используемых для построения процессоров и оперативной памяти.

Световым аналогом электронной сверхрешетки являются фотонные кристаллы, представляющие собой твердотельные среды, диэлектрическая проницаемость которых изменяется в пространстве с периодом порядка длины волны. Пространственная периодическая модуляция свойств среды позволяет управлять частотной дисперсией и дифракцией электромагнитных волн, что крайне важно для реализации новых классов устройств, таких как оптическая память и оптические

переключатели. На основе фотонных кристаллов возможно создание светоизлучающих структур с заданной диаграммой испускания, оптических перестраиваемых фильтров, активных матриц для оптических модуляторов и переключателей. Примерами фотонного кристалла могут быть трехмерная кубическая решетка диэлектрических шаров и двумерная квадратная решетка цилиндрических пор (рис. 4.4).

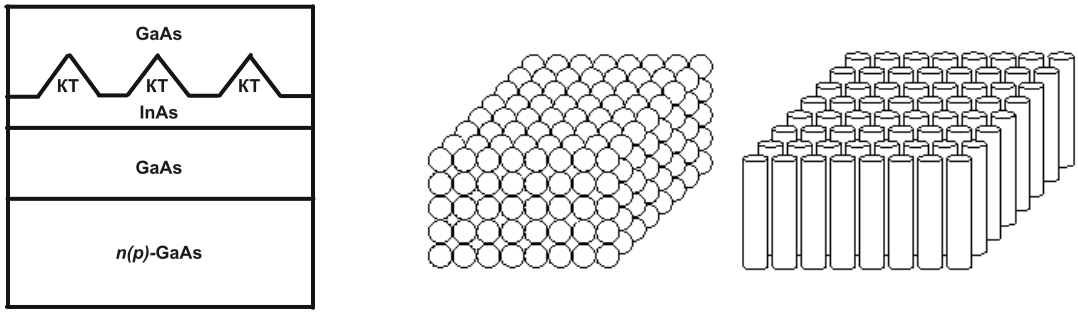


Рис. 4.4. Квантово-размерные структуры: слева – схема структуры с самоорганизованными квантовыми точками (КТ), изготовленными методом молекулярно-лучевой эпитаксии, справа – схема фотонных кристаллов двух типов.

§4.4. Поколения компьютеров [2.1, 2.3]

В вычислительной технике существует своеобразная периодизация развития электронных вычислительных машин. ЭВМ относят к тому или иному поколению в зависимости от типа основных используемых в ней элементов или от технологии их изготовления, а в последнее время – программных средств. Ясно, что границы поколений в смысле времени сильно размыты, так как в одно и то же время фактически выпускались ЭВМ различных типов, но для отдельной машины вопрос о ее принадлежности к тому или иному поколению решается достаточно просто. Утверждение понятия принадлежности компьютеров к тому или иному поколению относится к 1964 г., когда фирма IBM выпустила серию компьютеров S/360 на гибридных микросхемах, назвав эту серию компьютерами третьего поколения. Соответственно предыдущие вычислительные машины – на транзисторах и электронных лампах – были отнесены ко второму и первому поколениям. Такое деление на поколения было принято до середины 80-х годов, до появления т.н. «пятого поколения», и, с рядом оговорок, применимо и сейчас:

- 1 поколение – на электронных лампах (1945–1955).
- 2 поколение – на транзисторах (1956–1963).
- 3 поколение – на интегральных схемах (1964–1970).
- 4 поколение – на больших интегральных схемах (с 1971).
- 5 поколение – искусственный интеллект (1980-е).
- 6 поколение – компьютеры будущего (оптические, биологические, квантовые и нейрокомпьютеры).

1 поколение – ламповое (1945–1955)

Элементной базой компьютеров первого поколения стали вакуумные электронные лампы. Недостатками электронных ламп были низкая надежность и большие размеры (высота около 7 см). Громоздкими выходили и устройства на их основе. Для обеспечения термоэлектронной эмиссии требовался подогрев катода, поглощающий большую мощность, а выделяющееся тепло нужно было отводить. Тысячи ламп, составляющих первые компьютеры, размещались в металлических шкафах и занимали много места. Весила такая машина десятки тонн, при эксплуатации потребляла электрическую мощность, сравнимую с потреблением небольшого городка, и требовала специальных систем охлаждения – мощных вентиляторов и пр.

Устройств ввода в этих компьютерах не было, поэтому данные заносились в память при помощи соединения нужного штекера с нужным гнездом, а обслуживанием занимались бригады инженеров.

Из-за высокой стоимости реализации памяти на ламповых триггерах, при которой на один бит требовалось два ламповых триода, в качестве устройств хранения использовались ртутные линии задержки и электронно-лучевые трубки.

Примерами машин 1-го поколения могут служить ENIAC, EDSAC (Electronic Delay Storage Automatic Calculator) – первая машина с хранимой программой, UNIVAC, ставший первым серийным компьютером, и впервые использующим магнитную ленту, и МЭСМ.

2 поколение – транзисторное (1956–1963)

Первые компьютеры на основе транзисторов появились в конце 50-х годов. Применение транзисторов в качестве основного элемента привело к уменьшению размеров компьютеров в сотни раз и к повышению их надежности.

Выполняя те же функции, что и электронная лампа, транзистор вместе с тем работал с большой скоростью, имел значительно меньшие размеры, у него не было хрупкого стеклянного корпуса и тонкой нити накаливания, он не перегревался и потреблял гораздо меньше электроэнергии. Первые годы, из-за трудностей в производстве, цена прибора оставалась высокой: лучшие образцы транзисторов стоили до 8 долл. за штуку, в то время как цена лампы не превышала 75 центов, поэтому в массовое применение транзистор попал не сразу.

Одновременно с процессом замены электронных ламп транзисторами совершенствовались методы хранения информации. С появлением памяти на магнитных сердечниках цикл ее работы уменьшился до десятков микросекунд. Увеличился объем памяти, а магнитную ленту, впервые примененную в UNIVAC, начали

использовать как для ввода, так и для вывода информации. А в середине 60-х годов получило распространение хранение информации на магнитных дисках. Большие достижения в архитектуре компьютеров позволили достичь быстродействия в миллион операций в секунду. Начали создаваться системное программное обеспечение, алгоритмические языки и компиляторы, появились пакетные операционные системы.

В середине 60-х появляются первые миникомпьютеры – небольшие маломощные компьютеры, доступные по цене небольшим фирмам или лабораториям. Так, фирма DEC (Digital Equipment Corporation) выпустила в 1965 г. первый мини-компьютер PDP-8 (Programmed Data Processor) размером с холодильник и стоимостью всего 20 тыс. долларов. Семейство миникомпьютеров PDP послужило прототипом для советской серии машин СМ ЭВМ (Система малых ЭВМ).

Примерами транзисторных компьютеров могут послужить IBM 701 Defense Calculator, суперкомпьютеры IBM 7030 Stretch и Atlas (Великобритания), а также советские ЭВМ БЭСМ-6, Минск и Урал.

3 поколение – интегральные схемы (1964–1970)

С переходом на ИС малой и средней степени интеграции, вмещающих до тысячи элементов на один кристалл, производство компьютеров приобретает промышленный размах. Семейство System/360 стало первой серией полностью совместимых друг с другом компьютеров от малых, размером с небольшой шкаф, до самых мощных и дорогих моделей. Оно стало самым распространенным в мире.

Производство ЭВМ третьего поколения оказалось значительно дешевле, чем производство машин второго поколения, благодаря чему многие организации смогли приобрести такие машины. Это, в свою очередь, привело к росту спроса на универсальные ЭВМ, предназначенные для решения самых различных задач. Большинство созданных до этого ЭВМ являлись специализированными машинами, на которых можно было решать задачи какого-то одного типа.

В это же время появляются операционные системы, поддерживающие многозадачность, режим разделения времени, а также управление памятью, устройствами ввода-вывода и другими ресурсами. Тогда же появляется полупроводниковая память, которая и по сей день используется в персональных компьютерах в качестве оперативной.

Примерами компьютеров 3-го поколения считаются System/360 и разработанная на его основе в СССР серия ЕС ЭВМ (Единая серия).

4 поколение – сверхбольшие интегральные схемы (с 1971)

К началу 70-х годов развитие микроэлектроники сделало

возможным размещать на одном кристалле, площадью всего в четверть квадратного дюйма, до миллиона элементов. Появились сверхбольшие интегральные схемы (СБИС). Рост плотности размещения сопровождался и качественными изменениями в архитектуре.

Примерно в 1971 г. разработчики из нескольких фирм, таких как Intel, Texas Instruments и Garrett AiResearch, независимо друг от друга пришли к идее ограничить возможности процессора, заложив в него небольшой набор операций, микропрограммы которых должны быть заранее введены в постоянную память. Оценки показали, что применение ПЗУ с микропрограммами объемом 16 кбит позволит исключить 100–200 обычных интегральных микросхем.

Поначалу такие микропроцессоры содержали немного элементов, и применялись в калькуляторах или как встраиваемые решения. Они обрабатывали только 4 бита информации, но стоили в десятки тысяч раз дешевле процессоров больших компьютеров. В 1974 г. Intel выпустила микропроцессор 8080, с которого началась эпоха микрокомпьютеров и персональных ЭВМ. Благодаря этому вычислительная техника стала по-настоящему массовой и общедоступной. Несмотря на отставание персональных и миникомпьютеров от больших машин, львиная доля новшеств 90-х годов – современные операционные системы, графический пользовательский интерфейс, новые периферийные устройства, глобальные сети – обязаны своим появлением и развитием именно этой «несерьезной» технике. Большие компьютеры и суперкомпьютеры, конечно же, продолжали развиваться, но теперь они уже не доминировали на компьютерной арене, как было раньше.

С середины 70-х принципиальных новаций в компьютерной науке становится все меньше. Прогресс идет в основном за счет повышения мощности и миниатюризации элементной базы и самих компьютеров, так, теперь ИС вмещают сотни миллионов элементов на кристалл.

В связи с провалом программы «пятого поколения» иногда пятым поколением считают «поколение микропроцессоров», началом которого считается выпуск процессоров i8008 или i8080 (1974). В этом случае четвертое поколение описывает жизнь исключительно СБИС, существует параллельно с пятым поколением, и завершается с выходом в 1984-м году компьютеров на базе микропроцессора i80286, достигшего уровня производительности компьютеров IBM 370.

Примерами компьютеров 4-го поколения можно назвать IBM 3081 и Fujitsu M 380.

5 поколение – создание искусственного интеллекта (1980-е)

Компьютеры пятого поколения – широкомасштабная 10-летняя правительственная программа по развитию компьютерной индустрии и искусственного интеллекта, принятая в Японии в 1982 г. и серьезно

воспринятая в остальном мире. Целью программы было создание «эпохального компьютера» с производительностью суперкомпьютера и мощными функциями искусственного интеллекта. С помощью языков логического программирования (Lisp и Prolog) и новшеств в конструкции компьютеров планировалось вплотную подойти к решению одной из основных задач этой ветви компьютерной науки – задачи хранения и обработки знаний. Т.е. для компьютеров «пятого поколения» не пришлось бы писать программ, а достаточно было бы объяснить на «почти естественном» языке, что от них требуется.

Проект создания пятого поколения оказался провальным: попытку создания новой архитектуры не спасла даже значительная финансовая поддержка. Тем не менее, с проектом связывают прорыв в области проектирования прикладных программных продуктов. Одной из задач проекта было сделать общение конечного пользователя с компьютером максимально простым, подобным общению с любым бытовым прибором. Для решения поставленной задачи предлагались следующие направления:

1. Разработка простого интерфейса, естественно-языкового или графического, позволяющего конечному пользователю вести диалог с компьютером для решения своих задач. Задача поддержки естественно-языкового диалога до сих пор не решена. Доступный в настоящее время графический интерфейс операционных систем решает проблему только наполовину – позволяет конечному пользователю обращаться к заранее спроектированному программному обеспечению, не принимая участие в его разработке.

2. Привлечение конечного пользователя к проектированию программных продуктов. Это направление позволило бы включить заказчика непосредственно в процесс создания программ, что в конечном итоге сократило бы время разработки программных продуктов и, возможно, повысило бы их качество. Подобная технология связана с автоформализацией профессиональных знаний конечного пользователя и предполагает два этапа проектирования программных продуктов: создание программистом «пустой» универсальной программной оболочки и заполнение пользователем этой оболочки знаниями, носителем которых он является.

§4.5. Компьютеры будущего

Любая технология имеет свои пределы возможностей, развитие за границами которых возможно только на качественно ином уровне. Это относится и к полупроводниковой микроэлектронике, на базе которой построена современная вычислительная техника. На протяжении последних десятилетий рассматривались различные варианты возможного развития вычислительной техники, наибольший акцент

сделан на модели, в которых вместо электронов используются кванты света, построенные по аналогии с нервной системой, основанные на биологических молекулах или использующие квантовые эффекты. Перечисленным вариантам соответствуют оптические, биологические, квантовые и нейрокомпьютеры. В то же время ведутся работы по выводу на новый уровень и полупроводниковой электроники, либо рассматриваются варианты её применения в создаваемых принципиально новых вычислительных системах.

Полупроводниковая микроэлектроника

Одним из вариантов развития традиционной микроэлектроники является способ, при котором стараются использовать явления, которые ранее являлись преградой. Например, утечка вследствие туннелирования электронов традиционно является ограничителем предельной ширины затвора транзистора. Однако, при толщине барьера в несколько нанометров, энергетические уровни внутри него перестают занимать сплошной спектр и становятся дискретными, и эффективно туннелируют только те электроны, чья энергия совпадает с дискретными уровнями внутри барьера. Меняя энергию электронов или напряжение, подаваемое на барьер, становится возможным управлять величиной самого туннельного тока. Такие транзисторы имеют много меньшую ширину затвора, небольшой логический перепад напряжений, малые токи, быстродействие до десятков ГГц, и создаются по отработанной технологии полупроводниковой эпитаксии (наращивание слоев), но первое время функционировали только при криогенных температурах. В настоящее время успехи достигнуты в работах по интеграции с традиционными транзисторами, разрабатываются и модули памяти (т.н. TFET SRAM) [4.6].

Варианты ищутся не только на уровне технологий изготовления элементной базы, но и на уровне, например, логик. Так, с 1965 г. существует «размытая логика» (fuzzy logic), оперирующая со значениями, находящимися в диапазоне от 0 до 1, и к настоящему времени занявшая свое место в экспертных системах и управляющей электронике. Недавно в электронику попала и «вероятностная логика», оперирующая значениями из непрерывной шкалы от 0 до 1, где 0 – невозможное событие, 1 – практически достоверное. Вероятностная логика рассматривалась Аристотелем, Лейбницем и Булем, к ней обращался Нейман, рассматривавший синтез надежных систем из ненадежных автоматов. А теперь компания Lyric Semiconductor, созданная в 2006 в MIT, при поддержке агентства DARPA разрабатывает процессоры на основе «байесовских логических вентилей», оперирующих аналоговыми сигналами, отображающими значения вероятности различных процессов. Такие вентили построены на

транзисторах, которые работают не по принципу ключа (как в обычных цифровых схемах), по принципу диммера – регулятора мощности. По заявлениям компании, такие схемы позволят реализовывать адаптивные алгоритмы, а также на порядки быстрее обрабатывать ряд задач, таких как распознавание образов. Но в настоящее время компанией выпущен только чип для коррекции битов в микросхемах памяти [4.7].

Оптические (фотонные) компьютеры

Более кардинальный вариант развития предполагает замену электронов на фотоны, которые имеют ряд преимуществ при передаче и обработке информации – они не взаимодействуют друг с другом, меньше взаимодействуют со средой, выше их скорость и выше опорная частота (около 1000 ТГц). Также они не являются источником электромагнитных помех и, в свою очередь, не подвержены их воздействию. В простейшем случае фотоны можно использовать для коммуникации традиционных электронных схем. В более перспективном – для обработки информации, которая может осуществляться как в процессе переноса изображения через оптическую систему, так и путем осуществления переключений в оптических логических элементах.

Простейшие вычислительные оптические операции, осуществляющиеся путем переноса через оптическую систему, часто встречаются в обыденной жизни. Это масштабирование изображения при помощи линз, сложение и деление световых потоков в волоконных оптических разветвителях, умножение сигнала на коэффициент пропускания среды при прохождении через неё и пр.

Если же в оптический системе входная и выходная плоскости совпадают с передней и задней фокальной плоскостями сферической линзы и на вход такой системы поступает оптический сигнал $U_I(x_I,y_I)$, то выходной сигнал $U_H(x_H, y_H)$ совпадет с фурье-образом входного сигнала. Над подобными фурье-образами можно производить различные математические операции методами пространственной фильтрации. Такая оптическая система может состоять из следующих компонентов – источника света, двух линз – простейших систем преобразования Фурье, устройства ввода информации, пространственного модулятора света и детектора выходных сигналов [4.8]. В этом случае операции, подобные фурье-преобразованию, при любом размере двумерного сигнала проводятся всего за один такт. Теоретически, при линейных размерах изображения 1 см, пространственном разрешении 3 мкм и длине оптической системы порядка 30 см достижима пиковая производительность порядка 10^{16} элементарных операций в секунду. Способность оптической вычислительной системы образовывать параллельные связи между большим числом элементов и одновременно выполнять операции взвешивания и суммирования хорошо подходит и

для построения нейросетей.

С 1970-х годов исследованиями и разработками оптоэлектронных вычислительных систем стали заниматься в лабораториях DARPA. Основные задачи работ носили военный характер: отслеживание и сопровождение целей, навигация, связь, контроль территорий и т. д. О результатах работ агентство не сообщало.

Первым процессором, осуществляющим оптическую обработку информации путем переноса изображения, стала модель фирмы OptiComp (США), разработанный в 1994 г. Процессор по правилам булевой логики на тактовой частоте 100 МГц перемножал массив 1×64 бит на матрицу 64×64 (рис. 4.5). В качестве матрицы использовался акустооптический пространственный модулятор света. Производительность позволяла производить поиск в текстовых документах со скоростью 80 000 страниц в секунду. Спонсировали проект такие организации, как Office of Naval Research, Strategic Defense Initiative, NASA, в дальнейшем компания сотрудничала с Air Force Research Laboratory, DARPA, Boeing и Naval Air Systems Command. Дальнейшие разработки компании по оптической обработке информации ориентированы на применение в военных, авиационных, космических и банковских системах. Также компания занимается разработкой систем с трех- и четырехмерным оптическим преобразованием (N^3 и N^4, в сравнении с обычным N^2, т.е. N×N).

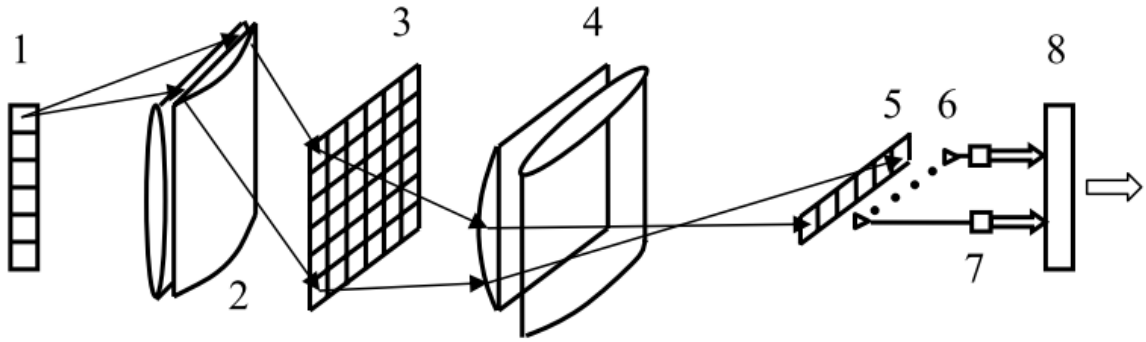


Рис. 4.5. Оптический векторно-матричный перемножитель. 1 – устройство ввода (матрица светоизлучающих элементов); 2, 4 – оптические системы, проецирующие каждый излучатель в плоскость модулятора; 3 – пространственный модулятор света; 5 – линейка фотоприемников; 6 – аналоговый электронный тракт; 7 – АЦП; 8 – цифровой сумматор [4.9].

Упомянем еще одну разработку – в 2004 г. компания «Lenslet» начала промышленный выпуск первого коммерческого гибридного оптического процессора EnLight256, его ядро основано на оптических технологиях, а все входы и выходы – электронные. Процессор способен выполнять до 8×10^{12} операций в секунду, а компьютер на базе EnLight256 способен обрабатывать 15 видеоканалов стандарта HDTV в

режиме реального времени. Входной «столбец» образует линейка из 256 лазерных диодов, а оптический пространственный модулятор MQWSLM (Multi-Quantum Well Spatial Light Modulator) выполнен на основе гетероструктур, образующих целые наборы квантовых ям.

Для развития оптических процессоров немаловажно и то, для его создания вполне может подойти обычный технологический процесс изготовления полупроводниковых микросхем.

Примеры прикладной реализации – это распознавание образов в режиме реального времени, обработка изображений и, в общем случае, обработка двумерных массивов данных. В качестве запоминающих и управляющих элементов оптических компьютеров изначально предлагалось использовать голограммы, сейчас фирмы GE и InPhase предлагают голографические диски памяти объемом до 500 ГБ для обычных компьютеров.

Первый рабочий оптический компьютер, продемонстрировавший возможность выполнения цифровых и логических операций, создали в 1990 году в исследовательском центре Bell Labs. Основу компьютера составляли двумерные матрицы бистабильных элементов на основе квантоворазмерных полупроводниковых структур – чередующиеся слои GaAs и GaAlAs толщиной по 95 нм, образующие структуру множественных квантовых ям. Объединение ячеек с возникновением положительной обратной связи позволило образовывать из них логические оптические элементы И-НЕ и ИЛИ-НЕ.

Показана возможность создания на основе фотонных кристаллов оптического транзистора, в котором управляющий поток света влияет на прохождение управляемого потока. Рассматриваются и другие варианты, например – управление интенсивностью одного потока света, проходящего через оптический микрорезонатор, другим потоком, вызывающим генерацию фононов (акустических фотонов) и микроколебаний образца. Исследуются возможности применения квантовых точек, фуллеренов, нанотрубок, графена и пр. В целом, развитие элементной базы тормозится недостаточным уровнем теории, которая заметно отстает от достижений разработчиков.

Нейрокомпьютеры [4.10]

Нейрокомпьютеры функционируют по принципу нервных клеток – нейронов, образующих массив параллельно работающих простых вычислительных элементов (нейросеть). В нейроне на разветвленную структуру коротких отростков (дендритов) от внешних нейронов поступают сигналы, которые, в зависимости от расположения контакта между нейронами (синапса) на дендрите, являются возбуждающими или тормозящими. При превышении суммарным сигналом некоторого порога тело нейрона возбуждается и генерирует импульс, который передается

по единственному длинному отростку (аксону) на другие нейроны (рис. 4.6). Нейроны выполняют простые функции с невысокой скоростью – около 3 мс, но, за счет большого количества (около триллиона) и за счет связей между ними (каждый нейрон связан с несколькими тысячами других нейронов), достигается высокое быстродействие. В искусственной нейросети сигнал ячейки нейрона S является линейной функцией входных значений X_i, при этом весовые коэффициенты w_i, а также уровень срабатывания функции порогового детектора F(S), могут меняться со временем, позволяя системе, при наличии обратной связи, самообучаться. В зависимости от используемого подхода, нейрокомпьютеры могут строиться на настоящих нейронах, на электронных «нейронах», программно эмулироваться на обычных компьютерах или строиться на базе оптических вычислительных систем. К основным преимуществам нейрокомпьютеров относят высокую параллельность, обучаемость и устойчивость к разрушению сети.

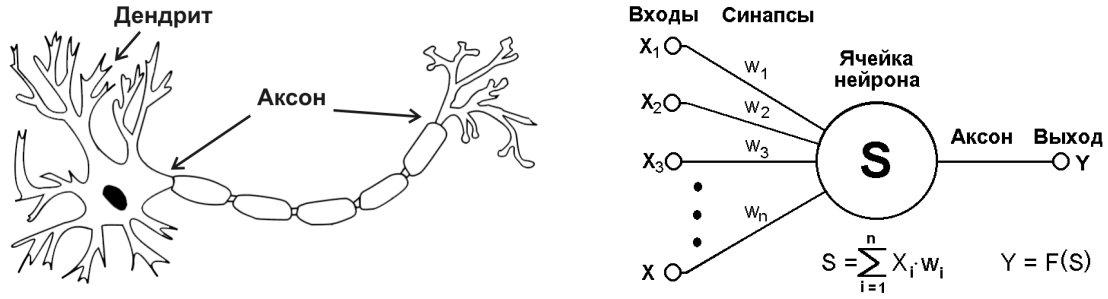


Рис. 4.6. Схема нейрона и его математическое представление.

История нейрокомпьютеров начинается в 1943 г., когда нейрофизиолог Уоррен Маккалох и математик Уолтер Питтс опубликовали теорию деятельности головного мозга, предложили конструкцию сети из электронных «нейронов» и показали, что подобная сеть может выполнять разнообразные числовые и логические операции. Их идеи опиралась на труды таких ученых, как Винера и Шеннона.

В 1957 г. нейрофизиолог Франк Розенблатт предложил модель искусственной нейронной сети, названную перцептроном, и реализовал её в виде машины «Mark-1», ставшей первым нейрокомпьютером. Перцептрон передавал сигнал от «глаза», состоящего из фотоэлементов, в блоки соединенным случайным образом электромеханических ячеек памяти, оценивающих относительную величину сигнала, на выход перцептрона поступает суммарный сигнал от ячеек памяти. При «обучении», заключавшемся в предъявлении изображений, весовой коэффициент блоков изменяется тем сильнее, чем больше на него поступило «синфазных» сигналов от фотоэлементов. В результате перцептрон может распознавать простейшие образы, например

некоторые буквы. Частично закрытая буква уже не распознавалась, но стало казаться, что мыслящая машина – дело ближайшего будущего.

В 1969 г. исследователь в области искусственного интеллекта (ИИ) Марвин Минский математически доказал ограниченность класса задач, решаемых перцептронами и показал, что основные задачи, которые он должен был решать, можно решить более простыми методами. С перцептронами были связаны завышенные ожидания, и критика Минского послужила одной из причин временной потере интереса к проблеме ИИ, продлившейся до 80-х гг. (т.н. «AI winter», «Зима ИИ»).

К настоящему времени накоплено большое количество архитектур нейронных сетей и правил их обучения, подходящих для решения разных прикладных задач, но задача универсальности не решена. Возможно, эта технология и не приведет к созданию компьютера нового поколения либо искусственного интеллекта, но разработки, включающие в себя нейросети, сегодня используются сегодня во многих областях. Среди существенных отметим обработку видеоизображений, обработку статических изображений, обнаружение летающих объектов, тепловизоры и криптографию. Разрабатываются методы организации сетей из биологических нейронов, подключенных к электрическим цепям, а также подключения компьютеров к нервной системе человека.

Квантовые компьютеры [4.11]

Квантовый компьютер (КК) – это вычислительное устройство, использующее при работе квантовомеханические эффекты. Это теоретически обеспечивает большой параллелизм вычислений. Идея квантового компьютера высказана в 1980 г. Ю.И. Маниным, обратившим внимание на способность двухуровневых квантовых систем находится в суперпозиции булевых состояний, а в 1982 г. опубликовал статью Р. Фейнман, предложивший моделировать состояния микрочастиц с помощью квантовомеханических элементов.

В КК аналогом обычных битов являются кубиты – т.е. биты, обладающие квантовыми свойствами. В качестве кубита может быть использована любая двухуровневая квантово-механическая система – спин электрона, ядерный спин атома, направление поляризация фотона и пр. Такой кубит находится в состоянии ψ, являющемся суперпозицией двух базовых состояний 0 и 1, т.е. $|\psi\rangle = a|0\rangle + b|1\rangle$. Измерение состояния кубита с вероятностью a^2 даст 0, и с вероятностью b^2 – 1, но до момента измерения можно считать, что он находится одновременно и в состоянии $|1\rangle$, и в состоянии $|0\rangle$ – это есть когерентная суперпозиция. Тут надо иметь в виду, что кубит содержит информации не больше, чем классический бит. Вследствие суперпозиции система из N кубитов имеет 2^N базовых состояний, во всех из которых она находится одновременно.

Отдельные кубиты могут быть приведены в состояние «сцепленности» («запутанности») – например, когда они испущены одновременно и обладают общей волновой функцией. Тогда измерения, проводимые над одним кубитом, окажут мгновенное воздействие на сцепленные с ним, даже если они разделены в пространстве. При этом корреляция между результатами измерений оказывается выше, чем это предсказывает теория вероятностей.

В квантовых вычислениях логическими гейтами называют внешние воздействия, по определенному алгоритму производящие преобразования над кубитами. Математически действие гейтов эквивалентно умножению двумерного вектора состояния квантового регистра, состоящего из N кубитов, на унитарную матрицу $2^N \times 2^N$. В логических гейтах часто используется сцепленность, например, когда нужно инвертировать значение, не проводя измерение, или когда нужно воздействовать на несколько кубитов одновременно.

В течение заданного времени после записи начальных состояний кубиты взаимодействуют друг с другом, а также подвергаются воздействиям логических гейтов. Преобразование, происходящее при этом, называется эволюцией, оно сопровождается созданием новых суперпозиций состояний, и происходит одновременно во всех возможных состояниях системы. Эволюция происходит без измерения значения кубитов, поскольку оно нарушит преобразование.

Таким образом, процесс вычислений в КК состоит из приготовления начального состояния регистра ячеек, временной эволюции квантового состояния системы кубитов и измерении результирующего состояния регистра (рис. 4.7). Полученный результат принципиально носит вероятностный характер, но, добавляя операции в алгоритм, можно приблизить вероятность правильного ответа к 1.

Рис. 4.7. Схематическая структура квантового компьютера [4.11].

Кубиты строят на основе разных объектов, с разной степенью успеха удалось реализовать следующие виды:

– фотоны (поляризация, количество, время прибытия),

– электроны (спин, количество)
– куперовские пары электронов (направление движение и энергия)
– твердотельные квантовые точки (спин, заряд, энергия)
– ионы, захваченные магнитными ловушками;
– биологические молекулы в мембранах;
– изотопы в полупроводнике.

Одной из существенных проблем, связанными с созданием и применением квантовых компьютеров, являются необходимость высокой точности измерений. Эти требования растут, поскольку с ростом числа возможных состояний экспоненциально растет число энергетических уровней системы, и, следовательно, уменьшается расстояние между ними и затрудняется извлечение результатов. Эта же проблема касается записи – сложно воздействовать на один кубит, не повлияв на другой, особенно если они распложены близко – иначе их будет не сцепить.

Другой проблемой является чувствительность к внешним воздействиям, вследствие чего время хранения информации в кубите на текущий момент не превышает нескольких миллисекунд.

«Квантовое ускорение» вследствие параллелизма возможно не для всех задач. На КК реализуются следующие основные алгоритмы:

1. Алгоритм Гровера – быстрый поиск в неупорядоченной базе данных. Для N записей поиск осуществляется за время $\sim\sqrt{N}$, в то время как быстрейший линейный алгоритм требует ~N времени.

2. Алгоритм Шора – разложение натурального числа n на простые множители за полиномиальное время ~log(n). Подходит для взлома алгоритма шифрования с открытым ключом RSA.

3. Алгоритм Дойча-Джоза – нужно определить, является ли функция двоичной переменной $f(n)$ постоянной (принимает либо значение 0, либо 1 при любых аргументах) или сбалансированной (для половины области определения принимает значение 0, для другой половины 1). Для решения этой задачи классическому детерминированному алгоритму необходимо произвести $2^{n-1}+1$ вычислений функции f в худшем случае. Алгоритм Дойча-Джоза всегда дает верный ответ, совершив лишь одно вычисление значения функции f.

На данный момент, наибольший квантовый компьютер составлен из 7 кубитов. Помимо создания собственно компьютеров, широко ведется разработка элементной базы для их реализации и методов квантового программирования. Считается, что внедрение квантовых компьютеров не приведет к решению принципиально нерешаемых классических задач, а лишь ускорит некоторые вычисления. Основные изменения ожидаются в применении КК к передаче данных и связи (плотное кодирование, квантовая телепортация, криптография).

С 2003 года организовано несколько коммуникационных сетей, в которых данные передаются в квантовом запутанном состоянии – это

обеспечивает квантовую криптографию данных. При попытке «прослушать» трафик такой сети квантовая запутанность разрушается, что сигнализирует принимающей стороне о попытке перехвата.

Биологические компьютеры [4.12]

Биологические компьютеры (ДНК- или РНК-вычисления) – это собирательное название для различных техник, так или иначе связанных с молекулами ДНК или РНК. При ДНК-вычислениях данные представляются в виде молекулярной структуры, построенной на основе спирали ДНК, а чтение, копирование и управление данными выполняется белками-ферментами.

Молекула ДНК представляет собой двойную спираль, составленную из пар четырех нуклеотидов: аденина (А), тимина (Т), гуанина (Г) и цитозина (Ц). Образование связей при соединении в двойную ленту возможно только между комплементарными парами оснований А-Т и Г-Ц. В природе ДНК является носителем наследственной информации, которая копируется при каждом делении клеток. Для этого при делении спираль расплетается на две нити, после чего два фермента полимеразы, считывая исходную последовательность, добавляют комплементарные нуклеотиды, формируя две одинаковые двойные цепочки (рис. 4.8). В работе полимераз вполне можно усмотреть аналог работы машины Тьюринга. Эту же параллель видна в синтезе белка – рибосома последовательно считывает нуклеотиды группами по три (т.н. кодон), присоединяя к цепочке будущей молекулы белка аминокислоту, закодированную кодоном. При этом работа рибосомы начинается со стартового кодона, и завершается стоп-кодоном.

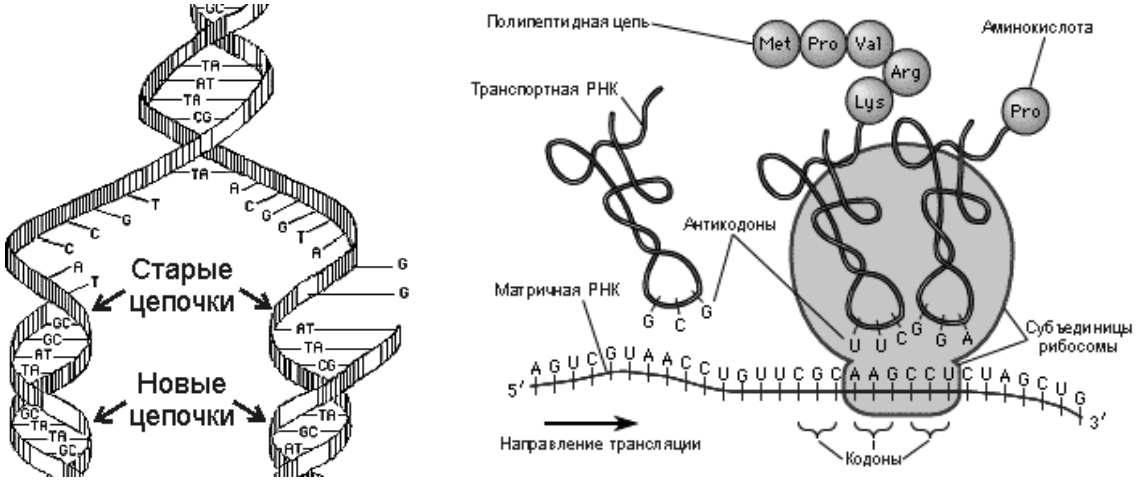


Рис. 4.8. Схемы репликации ДНК и синтеза белка (иллюстрация из [4.13]).

С помощью ферментов возможны и другие операции с ДНК, некоторые из которых аналогичны работе с одномерными массивами: модификация, разрезание и сшивание участков, размножение копий при

помощи полимеразно-цепной реакции в миллионах экземплярах, синтез цепочек нуклеотидов и определение их последовательности.

Основные надежды, которые возлагались на область ДНК-вычислений, были связаны с высочайшим параллелизмом и плотностью хранения информации – в одной пробирке может содержаться более 10^{15} нуклеотидов (т.е. более петабайта информации) и может идти одновременно более 10^{12} реакций, которые длятся от нескольких секунд до нескольких минут.

История ДНК-вычислений начинается с 1994 года, когда Леонард Эдлман (США) применил ДНК-вычисления для решения задачи коммивояжера на примере с 7 городами, соединенными 14 рейсами (поскольку метод предусматривал перебор всех возможных вариантов, то увеличение сложности потребовало бы гигантских количеств биологического материала). Эдлман синтезировал олигонуклеотиды (короткие одноцепочечные молекулы ДНК), кодирующие каждый город и маршрут между ними в виде последовательности нуклеотидов так, что половинки фрагментов «рейсов» были комплементарны городам, которые они связывали, а фрагмент «город» мог связать два фрагмента «рейса». Например, Атланта кодируется как ТГААЦГТЦ, Бостон – как ТЦГГАЦТГ, а рейс Атланта-Бостон – как ГЦАГТЦГГ. Тогда три фрагмента могут объединиться в цепочку, поскольку половина ГЦАГ фрагмента рейса комплементарна фрагменту ЦГТЦ «Атланта», а вторая половина рейса ТЦГГ – фрагменту АГЦЦ «Бостону» (рис. 4.9) [4.14].

Рис. 4.9. Эксперимент Эдлмана. Города и рейсы между ними закодированы комплементарными последовательностями нуклеотидов.

При синтезе образовывалась цепочка ДНК, соответствующая последовательному перемещению по городам. «Щепотка» молекул ДНК, содержащая около 10^{14} фрагментов, кодирующих города и рейсы, при растворении в пробирке примерно за секунду синтезировала триллионы разнообразных цепочек, среди которых был и правильный путь. Следующие несколько дней заняла фильтрация «неправильных» цепочек – которые не начинались и не заканчивались фрагментами, соответствующими местам старта и финиша, в которых повторялись фрагменты, длина которых не соответствовала числу 7 и т.д.

Эксперимент Эдлмана дал начало модели параллельной фильтрации, в которой множество всевозможных решений получается на первом шаге за счет того, что все взаимодействующие молекулы ДНК

спроектированы нужным образом. А основная часть алгоритма – это извлечение нужного результата из множества всевозможных результатов. Однако модель не годится для создания универсального вычислителя.

В 2001 г. Эхуд Шапиро (Израиль) реализовал конечный автомат, который был способен решать простейшие задачи – например, четное или нечетное количество символов находится во входной последовательности. Исходными данными для автомата является смесь фрагментов ДНК. Через час в растворе такая смесь, под действием двух ферментов, манипулирующих нитями из нуклеотидов, синтезирует молекулу ДНК, в которой закодирован ответ на поставленную задачу – истина или ложь. В одной пробирке помещается около триллиона элементарных вычислительных модулей, общая скорость вычислений достигла миллиарда операций в секунду, а вероятность получения правильного ответа 99,8 % [4.15].

Ввиду ограниченной универсальности авторы предлагают применять подобный вычислитель в медицинских целях – проанализировав матричные РНК на предмет экспрессии и репрессии определенных генов (т.е. активности и подавленности), идентификатор заболеваний может синтезировать лекарственный олигонуклеотид ssDNA, подавляющий синтез определенного белка, участвующего в заболевании (рис. 4.9).

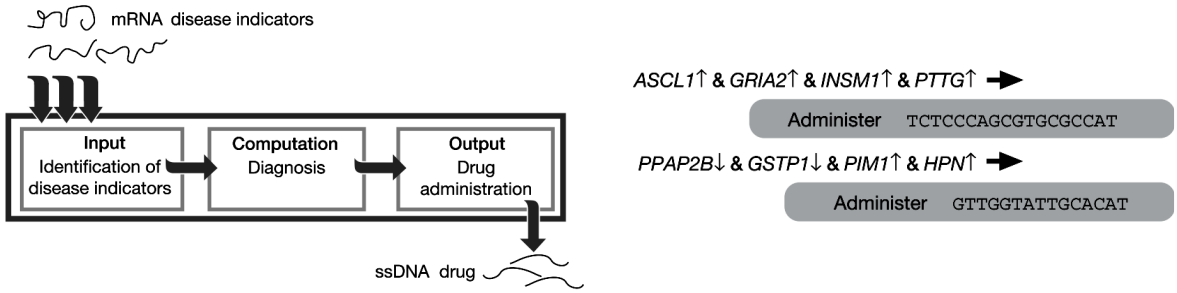


Рис. 4.9. Молекулярный компьютер Шапиро. Слева – схема операций, справа – примеры диагностических правил. Например, если гены ASCL1, GRIA2, INSM1 и PTTG1 – экспрессивны (обозначены «↑»), то назначается олигонуклеотид TCTCCCAGCGTGCGCCAT, позволяющий провести терапию карциомы легких. Второе правило – синтез запускается, если два гена репрессивны, и два – экспрессивны.

В 2003-2010 гг. в США исследователи создали несколько молекулярных вычислительных автоматов MAYA (Molecular Arrays of YES and AND gates) на основе молекул ДНК, способных обучаться и реализовывать стратегию логических игр. На доске 2×2 и 3×3, на примере напоминающих «крестики-нолики» игр, создатели показали, что автомат может сначала «обучаться» правилам игры, а затем «принимать» решения согласно предложенной стратегии. Каждая клетка доски представляла собой пробирку, содержащую раствор

дезоксирибозимов (молекул ДНК, обладающих ферментативной активностью) нескольких десятков типов. Каждый дезоксирибозим является логическим гейтом (И, И-НЕ, сдвоенным И) и имеет два или три участка (входа), к которым могут присоединяться олигонуклеотиды двух типов, активизирующих либо подавляющих работу дезоксирибозимов. Подобранные дезоксирибозимы позволяют охватить все варианты ответов на протяжении нескольких ходов, поэтому система напоминает программируемую логическую матрицу [4.16, 4.17].

Ответ автомата фиксировался по свечению люминесцентных меток, которые начинают люминесцировать в зависимости от конфигурации, которую принимает молекула ДНК в ходе химической реакции (свечение – 1, гашение – 0). «Обучение» заключалось в добавлении в пробирки нуклеотидов, способствующих образованию структур, позволяющих возникать люминесцентному свечению.

Созданный автомат реализовывает булевы функции, но отнести его к универсальным сложно. Одной из проблем является низкая скорость (до получаса на ход). Но устройство может найти применение в поиске генетических нарушений. В нем не используются токсичные материалы, в отличие от микросхем, также оно может работать в жидкой среде, например в крови. Возможно использование вместо молекул бактерий, изменяющих свое состояние под действием света. Теоретически скорость работы будет ниже, чем с ДНК, но варианты ответа не ограничиваются «да» и «нет». По мнению Шапиро, основное назначение молекулярных вычислителей, все-таки, тонкий химический синтез, сборка нужных молекул и конструкций, а не вычисления.

ДНК-вычисления нашли применение и в других областях, поскольку из нитей с заданной последовательностью нуклеотидов сворачиваются определенные фигуры. Синтезируя плоские фигуры, можно решать задачи по «замощению», а объемные фигуры позволяет создавать сложные молекулярные конструкций. Заданные структуры можно синтезировать как из ДНК, так и из белка – причем из него гораздо перспективнее за счет больших разнообразия элементов.

К 2002 относится начало разработок машин молекулярного состава на основе ДНК. Синтезируя фрагменты ДНК определенной последовательности и внедряя в биологические молекулы, можно заставлять молекулы самоорганизовываться и объединяться друг с другом, создавая сложные структуры и механизмы. В дальнейшем нити ДНК могут извлекаться либо заменяться на другие нити.

Способность биологических молекул к самосборке пытаются использовать и для электроники. Так, в 2003 г. в институте Вейцманна (Израиль) был создан самособирающийся транзистор на основе углеродных нанотрубок, ДНК, и ионов золота и серебра. Для этого они покрыли частицу ДНК белками бактерии E. coli, а нанотрубки покрыли

антителами к ним. В результате при смешивании ДНК оказались связанными с нанотрубками, и конструкция, получившаяся после всех манипуляций, работала как транзистор.

В 2004 г. в университете Миннесоты (проф. Ричард Киль) разработали экспериментальные биоэлектронные схемы. Для этого фрагменты ДНК собирались в плоскую структуру, способную в нужных местах принять органические молекулы или ионы металлов. Такие компоненты способны хранить электрический и магнитный заряд, поэтому открывается возможность создать быстродействующие схемы с высокой плотностью упаковки информации (характерное расстояние между деталями – треть нанометра).

Исследовалась и возможность использовать ДНК в квантовых компьютерах. Так, молекулы, прикрепленные к поверхности квантовых точек, могут «склеивать» их в более сложные конструкции, которые будут выполнять логические функции. Более того, квантовый гейт может быть реализован непосредственно в спирали ДНК. Для этого к цепочке, симметричной относительно центра, нужно прикрепить ловушки электронов или дырок и внедрить в них нужные частицы. Тогда квант света, падающий на центр цепочки, благодаря сбалансированности системы, сможет вводить частицы на противоположных сторонах в когерентное сцепленное состояние [4.18].

Завершая рассмотрение перспективных технологий, отметим, что, несмотря на кажущееся разнообразие в направлениях развития, на деле они оказываются смежными. При планируемых размерах все частицы – кванты, атомы, молекулы, квантовые точки, нити и плоскости – подчиняются законам квантовой физики.

Также следует иметь в виду, что большинство футуристических прогнозов не сбывается, поскольку все они делаются, экстраполируя тенденции прошлого в будущее. Но каждый новый шаг на пути прогресса требует все больших усилий, а каждое вложение дает все меньшую отдачу, поэтому в будущее стоит смотреть со сдержанным оптимизмом, четко представляя себе ограничения и пределы развития. Кроме того, никто не предсказывает появление точек поворота, бифуркаций. Ими может быть как открытие принципиально новой технологии, так и потеря актуальности задач, для которых разрабатываются эти технологии.

§4.6. Стандартизация вычислительной техники. System/360 [4.19, 4.20]

К началу 1960-х годов каждая новая модель компьютера была уникальной, их архитектура отличалась даже от изделий той же компании. Это приводило к отсутствию стандартов на компьютеры, периферийную технику, а также к невозможности переноса программ

между моделями. Степень разнообразия можно оценить, рассмотрев типы используемых данных.

В компьютерах, предназначенных для научных вычислений с числами в формате с плавающей запятой, длина слова выбиралась исходя из решаемых задач, под которые проектировался компьютер. Они оперировали словами длиной 32, 36, 54 или 60 бит, а разработанный в СССР БЭСМ-6 оперировал словами длиной 48 бит.

Компьютеры, предназначенные для финансовых и статистических вычислений, оперировали с двоично-десятичными числами, и не могли выполнять операции «плавающей» арифметики.

В самой компании IBM, совершившей несколько лет спустя переворот в производстве вычислительной техники, вычислительные машины также делились на два направления – научное и коммерческое. Научная линия состояла из 36-битных двоичных машин с аккумуляторной архитектурой, обрабатывающих числа с плавающей запятой (модели IBM 701, 704 и 709), а коммерческая линия состояла из десятичных машины со словами переменной длины и с 5-битовой длиной инструкций (модели IBM 702 и 705).

Проект IBM System/360 (1964)

В 1961 г. в компании IBM, в то время разрабатывающей два разных типа машин, сочли недостаточно революционной новую модель компьютера для научного применения (8000-ю серию), и раскритиковали его несовместимость с машинами общего (коммерческого) назначения. Работу по созданию «новой линии продуктов» (New Product Line), объединившую оба направления, возглавил Б. Эванс. Для выработки концепции NPL был создан комитет SPREAD (Systems Programming Research and Development), изложивший семь основных принципов:

- центральный процессор должен с равным успехом использоваться для научных и деловых вычислений;
- все члены будущего семейства должны быть способны работать с одним и тем же набором периферийных устройств;
- для центрального процессора была выбрана гибридная технология, допускавшая использование микросхем и навесной монтаж дискретных элементов (к тому времени уже существовали микросхемы, но они еще не обладали достаточной надежностью);
- в научных и бизнес-вычислениях должен использоваться один и тот же язык программирования высокого уровня;
- между всеми членами семейства должна сохраняться программная совместимость;
- адресация в семействе должна обеспечивать доступ к 16 млн. символов, а в перспективе – к 2 млрд.;

- минимальной единицей представления данных будет 8-битовый байт.

В дальнейшем проект был переименован из NPL в System/360, что должно было означать способность решения любых задач. По своему размаху проект создания System/360 один из крупнейших коммерческих в истории, сопоставимый с проектами полета на Луну и освоения западносибирских нефтегазовых месторождений. В течение короткого промежутка времени в один проект было инвестировано свыше 5 млрд. долл. (по курсу начала 2000-х гг. это свыше 30 млрд. долл.). Для IBM это была рискованная игра, т.к. за несколько лет было заново создано практически все: архитектура, элементная база и системное программное обеспечение.

Компьютеры серии System/360 были выпущены в 1964 г., они обладали производительностью в широком диапазоне – от 0,024 до 1,7 MIPS, имели память от 8кБ до 8МБ, и продавались по ценам от $2,700 за базовую конфигурацию до $115,000 за типичную большую мультиконфигурацию. Серия была доступна большому числу компаний. Сопоставимый диапазон в ценах начала 2000-х – от $133,000 до $5,500,000 [4.9].

Многое из того, что заложено в System/360, стало фундаментальной базой для развития компьютеров на следующие десятилетия. Это разнообразные аппаратные и программные технологии, коммуникационные возможности, микрокод, но прежде всего – программная совместимость в пределах всего семейства компьютеров – от самого младшего до самого старшего. В настоящее время в IBM совместимость поддерживается и между поколениями. Так, программы для System/360 будут работать и на новейших майнфреймах серии Z.

Основные разработчики System/360 – Эрих Блох (руководитель направления создания микросхем Solid Logic Technology), Боб Эванс (руководитель всех работ по созданию System/360), Фредерик Брукс (разработчик серии 8000, разработчик семейства операционных систем OS/360).

Проект повлиял и на развитие вычислительной техники в СССР, где был «клонирован» как ЕС ЭВМ.

Форматы представления данных в System/360

Форматы данных и команд, выбранные для System/360, повлияли на форматы, применяющиеся во всех других компьютерах, поэтому рассмотрим, почему были выбраны именно эти форматы:

•8-битные символы, 8-битная ячейка памяти;

•16-битные полуслова, 32-битные слова, 64-битные двойные слова;

•32/64-битный формат чисел с «плавающей» запятой;

•Двоично-десятичный формат (упакованный и разупакованный);

•Поддержка наборов символов ASCII и EBCDIC;
•Выровненные по 16 бит команды.

При определении оптимального размера символа разработчики сравнивали преимущества и недостатки вариантов с длиной 4, 6 и 8 бит. Они исходили из того, что десятичные цифры требуют 4 бита, а алфавитно-цифровые символы – 6 бит. Плюсы 6-битного подхода, используемого в компьютерах IBM 702-7080 и 1401-7010, так же как в системах других изготовителей – существующее оборудование ввода-вывода, простая спецификация полевой структуры, и соизмеримости с 48-битовым словом с плавающей запятой и 24-битовой областью команды. Подход 4/8 бит, используемый в семействе IBM 650-7074, имел большую эффективность кодирования, запасные биты в алфавитном наборе (возможность добавления знаков), и соизмеримость с 32/64-битным словом с плавающей запятой и 16-разрядной областью команды. Также рассматривались вариант 12-битного модуля для 3 цифр или 2 букв и вариант 7-битного символа.

Подход 4/6 бит был отклонен, потому что:

– сложно оперировать символами/потоками даже в моделях, где десятичная арифметика не используется, технические сложности этого подхода могли бы стоить больше, чем потраченные впустую биты в символе;

– решение ограничить набор символов 6 битами было сочтено близоруким.

Кроме того, 8-битное поле обеспечивало эффективное кодирование десятичных данных, т.к. в деловых отчетах они встречаются вдвое чаще, чем алфавитно-цифровые.

В качестве вариантов представления чисел в «плавающем» формате рассматривались слова длиной 32/48/64 бит. Короткие форматы обеспечивают скорость и эффективность кодирования, длинные – точность. 48-бит был самым распространенным форматом, а формат 32/64 давал возможность выбора между точностью и скоростью.

36- и 48-разрядность для ряда задач оказалась недостаточной, и выбор стоял между 48 и 64. Разработчики решили иметь одновременно возможность работы с числами 64- и 32-разрядной длины. Пользователи больших моделей, использующие большую часть времени слова 64-битной блины, получали полную поддержку обоих форматов. А пользователи младших моделей, использующих в большинстве случаев слова 32-разрядной длины, получали машину с аппаратной поддержкой 32-разрядных вычислений. Но все модели работали с двумя форматами – младшие модели получали поддержку длинного формата на программном уровне, зашитом в микрокод.

В выбранных 32- и 64-битных форматах на знак отводился 1 бит,

на порядок – 7, а оставшиеся разряды отводились под мантиссу. Современные форматы одинарной и двойной точности отличаются от выбранных только большим числом бит, отведенных под порядок.

В машинах также был оставлен двоично-десятичный формат, т.к. при коммерческом применении он упрощает десятичные расчеты.

Выбор в 1961 г. 8-битной кодировки символов оказался верным решением, т.к. в 1963 г. Ассоциации по стандартам адаптировала 7-битный код для обмена информацией (стандарт ASCII). Но в серии System/360 поддерживался не только новый стандарт, но и распространенный расширенный двоично-десятичный код EBCDIC. Это обеспечило совместимость как с информацией на старых перфокартах, так и с имеющейся периферией, созданной для предыдущих моделей.

Команды в процессоре были выровнены по 16 бит и могли быть длиной 16, 32 и 48 бит. Процессор компьютера был построен по регистровой архитектуре, имел шестнадцать 32-битных регистров общего назначения и четыре 64-битных регистра для работы с дробными числами. В процессоре применялся микрокод, позволяющий сохранять функциональность без усложнения архитектуры.

Помимо форматов данных, влияние IBM проявилось даже в нотации. Так, распространенная восьмеричная запись двоичных чисел была вытеснена шестнадцатеричной, выбранной для System/360.

§4.7. БЭСМ-6 (1967) [4.21]

В развитии отечественной вычислительной техники особое место занимает ЭВМ БЭСМ-6 (Быстродействующая Электронно-Счетная Машина). БЭСМ-6 была разработана под руководством С.А. Лебедева, производство начато в 1967 г. и продолжалось до 1987 г. Она была задумана как ЭВМ для расчетов в самых различных областях науки и техники и для оснащения крупных вычислительных центров. Формально ее относили к ЭВМ 2-го поколения, так как ее элементная база выполнена на дискретных элементах. Но по всем прочим признакам – иерархической системе памяти с постоянной организацией, наличию виртуальной памяти, большому числу каналов, обслуживающих периферийные устройства и внешнюю память, наличию эффективных операций с плавающей запятой, наличие сверхбыстрого ЗУ, системы индексации команд, системы прерывания, наличию операционной системы и т.д. – БЭСМ-6 являлась вычислительной системой 3-го поколения. В течение нескольких лет она была самой высокопроизводительной ЭВМ в Европе. «Особое» место БЭСМ-6 занимает и потому, что в 1967 году было принято решение руководства СССР о копировании System/360, и БЭСМ-6 стала последним оригинальным компьютером для массового применения.

В системе команд БЭСМ-6 предусмотрено 50 команд, разделяемых

на шесть групп по назначению, и макрокоманды. Код макрокоманды вызывает прерывание, и операционная система передает управление программе с номером данной макрокоманды.

Ряд необычных схемных решений в арифметическом устройстве (АУ) обеспечил возможность при тактовой частоте 10 МГц получить высокую скорость выполнения операций: сложение – 1,1 мкс, умножение – 1,9 мкс, деление – 4,9 мкс, прочие операции – 0,5 мкс. Это позволяет оценить среднее быстродействие АУ в 1 млн. оп/с. Чтобы быстродействие АУ не ограничивалась низкой скоростью записи и считывания данных и команд из устройств памяти, в БЭСМ-6 были применены 16 полноразрядных регистров сверхбыстрой памяти, что существенно ускорило ход вычислений. Отбор чисел и команд в них велся автоматически из наиболее часто повторяющихся при выполнении программы адресов.

БЭСМ-6 применялась до начала 90-х гг. С её помощью обрабатывалась телеметрическая информация и о полете «Союз-Апплон» в 1975 г., и о полете «Бурана» в 1988 г. В начале 80-х выпускался и вариант БЭСМ на базе интегральных схем, что подтверждает прогрессивность архитектуры комплекса.

§4.8. Разработка вычислительной техники в ИТМО [4.22]

В 1931 г. в институте, в составе факультета Счетно-измерительных приборов, была основана кафедра «Математических и счетно-решающих приборов и устройств». Примечательно, что еще до создания кафедры на факультете некоторое время существовала лаборатория счетно-решающих приборов. Термин «счетные устройства» достаточно точно определяет средства, которые использовались для вычислений – инструментов счетовода был арифмометр, основным инструментом инженера – логарифмическая линейка. На базе счетно-перфорационного оборудования, такого как перфораторы, наносящие данные на перфокарты, табуляторы, сортировальные и печатающие устройства, создавались комплексы, выполнявшие достаточно сложные последовательности табулирования и сортировок. Вспомним, что это было самое начало пути развития вычислительной техники.

После войны кафедра, под руководством С.А. Изинбека, создателя первых систем приборов управления стрельбой и ученого в области орудийной наводки, совершила шаг к аналоговым системам управления специального назначения. В конце 50-х гг. на кафедре был развернут уникальный крупный комплекс электромеханического действия для управления стрельбой тяжелых крейсеров. Работа кафедры со счетно-перфорационной и с аналоговой электромеханической техникой отвечала реальным потребностям времени.

В 1956 г., за год до начала выпуска в СССР первого серийного

компьютера (Урал-1), на кафедре началась разработка собственной вычислительной машины ЛИТМО-1, основанного на электронных лампах (рис. 4.10). В 1959 г. компьютер был введен в эксплуатацию и задействовался в расчете оптических систем, создававшихся в институте. ИТМО стал первым высшим учебным заведением в стране, наряду с МИФИ, разработавшим и получившим собственный компьютер.

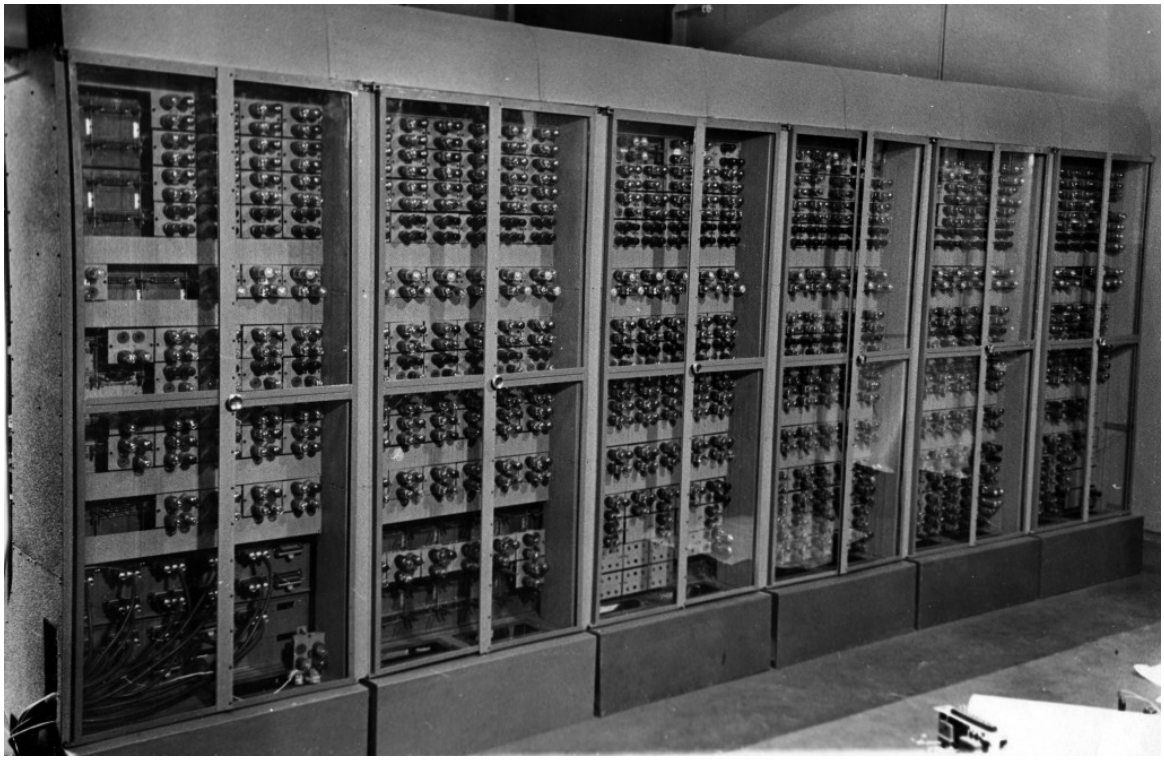


Рис. 4.10. Компьютер ЛИТМО-1

В 1962 г., под руководством С.А. Майорова, незадолго до этого возглавившего кафедру, в ИТМО начался проект создания второго компьютера – миниЭВМ ЛИТМО-2. Проект охватил все аспекты разработки компьютера – от производства комплектующих до создания приложений. Для сравнения – компания DEC начала производство миникомпьютеров серии PDP для массового потребителя в 1959 г., добившись в 1966 г. массового успеха модели PDP-8. В СССР проект первой миниЭВМ был начат в 1962 г., а производство – в 1965.

С появлением в 1963 г. в институте первого серийного компьютера Минск-2 кафедра стала первым партнером разработчиков компьютера по созданию FORTRAN-подобного языка и транслятора с него.

В связи с получением в 1975 г. первого компьютера ЕС ЭВМ Вычислительная лаборатория кафедры ВТ была преобразована в Вычислительный центр ЛИТМО. В организации ВЦ ключевую роль играл С.А. Майоров, а кафедра ВТ обеспечивала все кадровые потребности института в компьютерных специалистах. Среди выпускников кафедры – В.Г. Парфенов, организовавший и возглавивший

в 2000 г. факультет Информационных технологий и программирования, а также являющийся одним из создателей и руководителей Центра подготовки одаренных молодых программистов.

Микропроцессорная эра для кафедры ВТ наступила в 1977 г., с появлением секционированных микропроцессорных комплектов К584. В 1978 г. был произведен микрокомпьютер ЛИТМО-3 с оригинальной системой команд, в определенном смысле предвосхитивший RISC-подход. Дальше создавались варианты на микропроцессоре К580, востребованные и практически использовавшиеся в организациях, но не дошедшие до серийного производства. В настоящее время кафедра продолжает разрабатывать сложные информационные системы на современных программно-аппаратных платформах.

Говоря о настоящем времени, с обучением, исследованиями и работой в области информатики и вычислительной техники в ИТМО непосредственно связаны два специализированных компьютерных факультета – Компьютерных технологий и управления, созданный в начале 1990-ых годов на базе факультета Точной механики и вычислительной техники, и Информационных технологий и программирования, выделившийся из факультета КТУ в 2000 г., а также факультет Фотоники и оптоинформатики, созданный в 2002 г. и занимающийся оптическими технологиями обработки информации.

В рамках Национального исследовательского университета в 2009 г. в ИТМО создано 6 научно-образовательных центров по двум приоритетным направлениям развития – «Информационные системы, технологии программирования и управления» и «Оптические и лазерные системы, материалы, технологии». Исследования ведутся в областях технологий программирования, систем искусственного интеллекта, интеллектуальных систем обработки информации, робототехники, высокопроизводительных вычислений, передачи, обработки и записи информации, теории оптических и квантовых систем, создания оптических материалов и квантово-размерных структур.

ГЛАВА 5. РАЗВИТИЕ АРХИТЕКТУРЫ МИКРОПРОЦЕССОРОВ

Микропроцессоры – это программно управляемая сверхбольшая интегральная схема (или сборка схем), управляющая работой компьютера и выполняющая большую часть обработки информации. Микропроцессоры, появившись в 1971 г. как «начинка» калькулятора, где они выполняли базовые арифметические операции с двоично-десятичными числами, через два десятилетия стали основой всех вычислительных машин.

Из основных дат развития микропроцессоров отметим следующие:

- самый первый микропроцессор SLF – 1968 г.;
- первый массовый микропроцессор Intel 4004 – 15 ноября 1971 г.;
- первые микропроцессоры для компьютеров i8080 и MC6800 – 1974;
- основные процессоры Intel 8086 – 1978 г., 286 – 1982, 386 – 1986, Pentium Pro – 1995, Pentium 4 – 2000, Pentium M – 2003, Core – 2006;
- разработка RISC-процессоров – около 1980 г.;
- первый двухядерный процессор – 2001 г. (Power4).

§5.1. Основные архитектурные решения, применяемые в микропроцессорах

- Принстонская/гарвардская архитектуры
- Конвейерная архитектура
- Суперскалярная архитектура
- полный и урезанный наборы команд (CISC/RISC-процессоры)
- Многоядерность
- Кэширование
- Аккумуляторная/стековая/регистр-регистровая архитектуры
- Векторность

Большинство архитектурных методов, воплощенных в микропроцессорах, было перенесено из процессоров «больших» ЭВМ, которые в 70-е гг. опережали в своем развитии на десятилетия.

Гарвардская архитектура, предусматривающая раздельный доступ к инструкциям и данным, была разработана в 1930-е гг. в Гарварде Г. Эйкеном, создавшим позднее Harvard Mark I. Ввод инструкций в Mark I осуществлялся с перфоленты, данных – набором регистров. Раздельный ввод данных и команд позволял ускорить работу, но удорожал схему.

Принстонская архитектура, предусматривающая хранение

программ в общей памяти с данными, была разработана в 1940-е гг. в Пенсильванском и Принстонском университетах. По имени одного из руководителей работы она также называется «фон Неймановской». Совместное размещение инструкций и данных повысило гибкость вычислительных систем в плане обработки данных. Кроме того, такая архитектура была проще в реализации, поэтому она нашла повсеместное применение.

В 1970-е гг. вновь обратились к гарвардской архитектуре, она стала применяться в микроконтроллерах. Затем, вследствие возникновения т.н. «бутылочного горлышка фон Неймана», её элементы стали появляться и в микропроцессорах – например, в виде раздельной кэш-памяти для команд и данных или в виде запрета на исполнение данных как инструкций.

Конвейер для инструкций (pipelining) был впервые применен в компьютерах ILLIAC II (1962 г.) и IBM 7030 Stretch (1961 г.), позже С. Крей в суперкомпьютерах XMP применил его для операции многократного умножения и сложения (1982 г.).

Конвейерная архитектура позволяет повысить быстродействие за счет начала обработки команды до окончания обработки предыдущей. Обычно для выполнения каждой команды требуется осуществить некоторое количество однотипных операций, например: выборка кода операции из памяти, дешифрация команды, адресация операнда и его выборка из памяти, выполнение команды, запись результата в память. Каждой из этих операций сопоставляют одну ступень конвейера. После освобождения от выполнения элементарной операции одной команды ступень конвейера сразу приступают к работе над следующей командой. Так процессор обрабатывает одновременно несколько команд, находящихся на разных стадиях выполнения. Быстродействие, в самом оптимистичном случае, пропорционально длине конвейера.

Ряд факторов снижает эффективность конвейерной архитектуры – это простой конвейера, когда некоторые ступени не используются, ожидание, если следующая команда использует результат предыдущей, и очистка конвейера при выполнении перехода. Для повышения эффективности конвейера команды делают максимально схожего формата, применяют внеочередное выполнение команд и предсказание переходов. Последний способ особенно развит в современных процессорах, некоторые из которых имеют более 30 ступеней.

Суперскалярность – это способность параллельного выполнения нескольких машинных инструкций, которая обеспечивается работой нескольких декодирующих блоков, нагружающих множество исполнительных блоков.

Первым суперскалярным компьютером считается CDC 6600, разработанный С. Креем в 1964 г., а первыми суперскалярными

массовыми микропроцессорами – SuperSPARC (1992) и Pentium (1993). Практически все современные процессоры являются такими, они оснащены несколькими блоками выборки, декодирования, арифметических операций и пр. В архитектуре VLIW суперскалярность закладывается в инструкции на стадии компиляции.

CISC (Complex Instruction Set Computing) – концепция проектирования процессоров, характеризующаяся полным набором команд. Объединение нескольких машинных операций в одной команде позволило непосредственно поддерживать языки высокого уровня, увеличить плотность кода и уменьшить обращения к памяти. В начале 1960-х это позволило добиться большой экономии на памяти и большей производительности при программировании на ассемблере, т.к. языки высокого уровня, такие как Алгол или Fortran, были не всегда доступны. Первым представителем CISC среди компьютеров считается System/360, среди микропроцессоров это Intel 8086 (1978) и Motorola 68000 (1979).

RISC (Reduced Instruction Set Computing) – вычисления с сокращённым набором команд. Концепция разработана независимо в IBM (1975) и университете Беркли (1980) с целью преодоления недостатков микропроцессоров CISC. Архитектура основана на статистическом анализе используемых команд и операндов, развитой регистровой архитектуре, отказе от микрокода и отказе от использования сложных команд в пользу нескольких простых.

Архитектура RISC позволила существенно упростить структуру процессора и получить большую производительность, чем CISC, поэтому с 1990-х гг. почти все процессоры являются RISC или RISC-подобными (Power), либо это CISC-процессоры с RISC-ядром (x86).

Многоядерность подразумевает использование несколько процессорных ядер в одном корпусе (на одном или нескольких кристаллах). Первой двухядерность применила IBM в процессорах Power4 (2001). На данный момент массово доступны процессоры с несколькими ядрами. В 2006 Intel продемонстрировала прототип 80-ядерного процессора.

Кэширование – это использование быстродействующей памяти (кэш-памяти) для хранения копий блоков информации из устройств памяти, вероятность обращения к которым в ближайшее время велика. Идея кэширования связана с иерархией запоминающих устройств, описанной еще фон Нейманом, но непосредственно идея была осуществлена позднее, в 1960-х. Сам термин «кэш-память» появился в 1967 г. как обозначение высокоскоростного буфера в компьютерах System/360. С появлением микропроцессоров кэширование оперативной памяти на кристалле процессора стало одним из самых простых архитектурных способов повышения производительности.

По организации регистров процессора выделяют несколько типов

архитектур. **Регистр-регистровая** архитектура характеризуется свободным доступом к внутренним регистрам для выборки аргументов и записи результата. Архитектура свойственна крупным компьютерам и RISC-процессорам. До появления RISC архитектуры микропроцессоры организовывались по **аккумуляторной** схеме – из регистров выделялся регистр-аккумулятор, являющийся для одного из аргументов источником и приемником результата вычислений. Это позволяло кодировать операции в однооперандные инструкции, а также сократить число регистров на кристалле. В **стековой** архитектуре операции производятся над значениями на вершине стека, результат кладется также на вершину. При этом структура процессора выходит очень простой, а производительность – низкой, поэтому применение нашлось только в простейших микроконтроллерах или цифровой обработке сигнала.

В **векторных** процессорах операндами команд могут выступать упорядоченные массивы данных – векторы. Идея векторной обработки появилась в начале 1960-х в корпорации Westinghouse Electric, планировавшей существенно увеличить математическую производительность путем использования множества простых математических сопроцессоров. Сопроцессоры должны были запускаться одной командой, поступающей на центральный процессор, и обрабатывать собственные данные. Затем идею попытались реализовать в проекте многопроцессорного ILLIAC IV (1966–1976), который сочли провалившимся, хотя компьютер оказался самым быстрым в мире. Первыми успешными реализациями архитектуры считаются TI Advanced Scientific Computer (1973) и CDC STAR-100 (1974), но известность пришла с выпуском суперкомпьютера Cray-1 (1976). Поначалу векторные процессоры были основой суперкомпьютеров, но в 1990-е гг. они стали вытесняться массовыми процессорами, которые, в свою очередь, стали получать векторные расширения (такие как MMX и SSE). Позднее персональные компьютеры обзавелись векторными процессорами в составе графических ускорителей и видеокарт.

§5.2. Архитектура CISC (1964)

Основоположником архитектуры CISC считают компанию IBM с базовой архитектурой System/360, в которой набор сложных команд реализован с помощью микропрограмм. К классическим CISC-архитектурам относят процессоры DEC VAX, к ней же близки и микропроцессоры x86 компании Intel (рис. 5.1).

Для CISC-процессоров характерно большое количество машинных команд, некоторые из которых функционально аналогичны операторам высокоуровневых языков программирования, большое количество методов адресации и большое количество поддерживаемых форматов команд различной разрядности.

На первых этапах реализация в компьютерах большого количества инструкций была оправдана, т.к. языки высокого уровня только начали развиваться, и это было удобно для программистов. Использование сложных инструкций позволяло сократить размеры программ, а значит уменьшить требования к памяти и время выполнения. А использование микропрограмм смягчало проблему пропускной способности памяти.

Сравнительно небольшое число регистров общего назначения в микропроцессорах было вызвано тем, что внутренние регистры перегружали ограниченную площадь чипа при существующем уровне технологии. Кроме того, использование большого числа регистров требовало применения большого числа битов инструкции для адресации отдельных регистров, что увеличивало бы программный код.

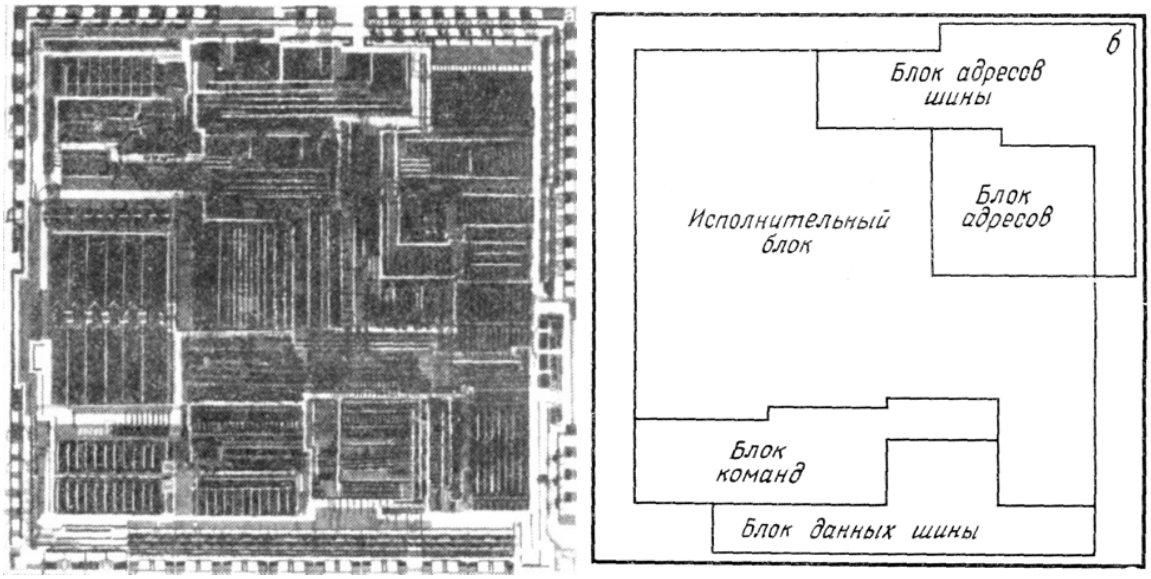


Рис. 5.1. Микропроцессор iAPX 286. Микрофотография и компоновочный план кристалла [5.1]

К началу восьмидесятых годов классические CISC полностью исчерпали себя, расширять набор инструкций больше не имело смысла. Процессор VAX поддерживал инструкцию «провести интерполяцию полиномом», а процессоры Motorola 68k поддерживали до двенадцати режимов адресации. Из-за высокой сложности процессоров оказалось трудно наращивать их тактовую частоту, а инструкции сложно было не только выполнять, но и декодировать (см. размер исполнительного блока на рис. 5.1). Чтобы машинный код CISC-процессоров из-за сложных инструкций не разрастался, инструкции имели неоднородную структуру и сильно отличающуюся длину (в x86, например, длина инструкций варьируется от 1 до 15 байт), что затрудняло работу конвейера. Часть инструкции оказалось невозможным выполнить аппаратно без существенного усложнения процессора, и поздние CISC-процессоры заменяли некоторые сложные команды на последовательность более

простых. Из-за такой «замены» возникали абсурдные ситуации, когда, например, инструкция INDEX выполнялась на VAX медленнее, чем вручную написанный цикл, выполняющий ровно тот же объем работы. Это происходило из-за того, что разработчики тратили гораздо меньше времени на улучшение сложных команд, чем на улучшение простых.

В результате все CISC-процессоры оказались весьма трудоемкими в проектировании и изготовлении. Но что самое важное, к 1980-м гг. исследования показали, что все сложные конструкции изобретались зря – и компиляторы языков высокого уровня, и программисты, пишущие на ассемблере, все эти «сверхвозможности» почти никогда не использовали.

§5.3. Архитектура RISC (1975–1983) [5.1]

Идея создания RISC процессоров пришла после того, как в 1970-х годах исследователи из IBM обнаружили, что многие функциональные особенности процессоров игнорируются программистами. Этот эффект лишь отчасти объяснялся сложностью компиляторов, которые могли использовать лишь часть из набора команд процессора, и тем, что некоторые сложные операции выполнялись медленнее, чем те же действия, выполняемые набором простых команд.

Исследователи провели анализ недостатков CISC, к числу которых были отнесены сложность блока дешифровки, конвейера, и «раздутый» блок микрокода. Также было проведено статистическое исследование частоты использования в программном коде инструкций и переменных. 80–90% переменных оказались локальными, при этом в 94–100% программ использовалось менее 12 переменных, а в 97–100% – менее 6-ти. Большинство используемых констант оказались существенно меньше двух байт, которые отводились под них (98% констант $\leq |511|$, 56% $\leq |15|$), а сложные операции, такие как mul/div, составляли 3–5% от всех математических операций.

RISC-процессоры проектировались в расчете на типовой код, генерируемый компиляторами. Разработчики свели к минимуму набор инструкций и количество режимов адресации памяти, создав простой и удобный для декодирования регулярный машинный код. В частности, из инструкций, работающих с памятью, оставлены только две – загрузки и выгрузки (архитектура Load/Store), а все сложные инструкции разбиты на ортогональные. Удалены инструкции вроде вычисления синуса, косинуса или квадратного корня (их можно реализовать «вручную»). В нескольких ранних экспериментальных моделях предпринимались даже попытки отказаться от аппаратного умножения и деления.

Второе важное усовершенствование RISC-процессоров, целиком вытекающее из Load/Store-архитектуры, – увеличение числа регистров общего назначения (до нескольких десятков). Эти регистры равноправны, что позволяет сохранять большую часть промежуточных

данных именно в них, а не в стеке или оперативной памяти. Все математические операции проводятся с данными, расположенными в регистрах, а при переключении между задачами вместо стека используется более быстрый механизм регистровых окон. Почти любой RISC-процессор обладает куда большим набором регистров, чем даже самый продвинутый CISC. Для сравнения: в классическом x86 всего восемь регистров общего назначения, причем многим из них приписано то или иное «специальное назначение» (например, в ESP хранится указатель стека) затрудняющее или делающее невозможным его использование (рис. 5.2).

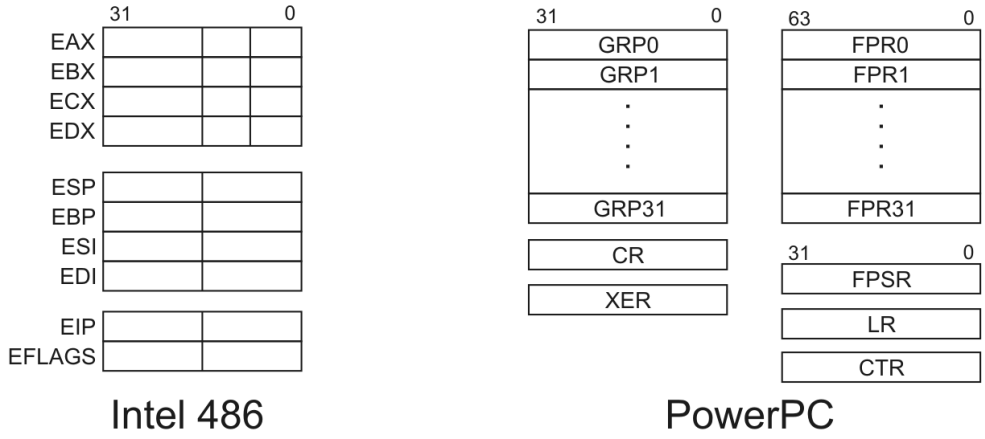


Рис. 5.2. Программная модель регистров процессоров i486 и PowerPC

Первая попытка создать RISC-процессор была предпринята в IBM. Работа, начавшаяся в 1975 г. с разработки коммуникационного процессора, привела к созданию в 1980 г. семейства процессоров общего назначения 801, которые широко использовались в различных устройствах IBM, а позднее привели к архитектуре POWER. RISC системы также разрабатывались в рамках университетских исследовательских программ, финансировавшихся программой DARPA VLSI. Но широкую известность технология получила благодаря проекту RISC в Университете Беркли (1980), в ходе которого группа студентов создала процессоры RISC-I (1982) и RISC-II (1983) (рис. 5.3). Они выполняли 50-100 команд вместо 100-200 у обычных процессоров CISC.

Достижения архитектуры, а именно резко уменьшившаяся сложность процессора и сопутствующее увеличение тактовой частоты и ускорение исполнения инструкций, уравновешивались возросшими размерами программ (на 30%) и сильно упавшей их вычислительной плотностью (средним количеством вычислений на единицу длины машинного кода). Но в то время, когда начались разработки RISC-архитектуры, в процессорах появился конвейер, реализация которого в рамках RISC архитектуры оказалась намного проще, чем в CISC. Это и вывело архитектуру в лидеры по производительности.

В настоящее время многие архитектуры процессоров являются RISC-подобными, к примеру, ARM, DEC Alpha, SPARC, AVR, MIPS, POWER и PowerPC. Процессоры архитектуры x86, начиная с Intel486DX, являются CISC-процессорами с RISC-ядром, преобразующими CISC-инструкции в набор внутренних инструкций RISC. При этом архитектурные регистры, начиная с P6, отображаются на 40 физических регистров.

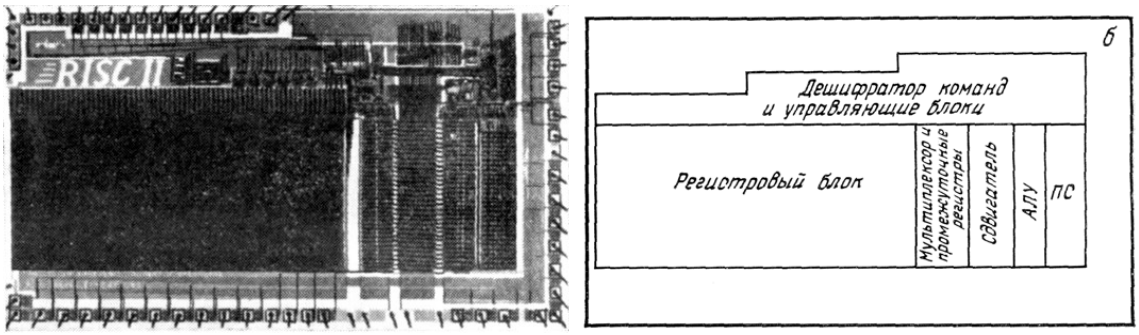


Рис. 5.3. Микропроцессор RISC II. Микрофотография и компоновочный план кристалла

§5.4. Архитектуры MIPS и VLIW

MIPS (1985) [5.1]

MIPS (англ. Microprocessor without Interlocked Pipeline Stages) – «микропроцессор без блокировок в конвейере». Основная идея – сильно упростив внутреннее устройство RISC-процессора и используя очень длинный конвейер, можно получить процессор, не умеющий выполнять сложные инструкции, зато работающий на очень высоких тактовых частотах, позволяющих скомпенсировать потери производительности на эмуляцию этих сложных инструкций. Изначально предполагалось, что MIPS-процессоры не будут аппаратно поддерживать даже операции умножения и деления, благодаря чему можно было обойтись без сложных в реализации блокировок конвейера, но в чистом виде идея безостановочной работы оказалась неработоспособной.

Процессоры MIPS начали разрабатываться в 1981 г. в Стэнфорде под руководством Джона Хэннесси, который в 1984 г. основал фирму MIPS Technologies, выпустившую через год первый процессор. Позднее, из-за конкуренции с процессором Intel Itanium, разработки были свернуты, а архитектура была лицензирована другим формам. Архитектура MIPS использовалась в компьютерах SGI, во встроенных системах и в игровых консолях Nintendo 64, Sony PlayStation, Sony PlayStation 2 и Sony PSP. Идея MIPS также была применена в процессорах P4 (Hyper Pipelined Technology).

В настоящий момент процессоры архитектуры MIPS используются

во встраиваемых устройствах, где критична производительность.

VLIW (1984–1987) [5.2]

VLIW (англ. Very long instruction word – «очень длинная машинная команда») – архитектура процессоров с параллелизмом на уровне команд, предусматривающая содержание в одной инструкции процессора нескольких операций, которые должны выполняться параллельно, и использование нескольких вычислительных устройств (рис. 5.5). Задача распараллеливания в таких процессорах решается во время компиляции, поэтому в инструкциях явно указано, какое вычислительное устройство должно выполнять какую команду.

Архитектуру VLIW можно считать логическим продолжением идеологии RISC, расширяющей её на архитектуры с несколькими вычислительными модулями. Так же, как в RISC, в инструкции явно указывается, что именно должен делать каждый модуль процессора. Из-за этого длина инструкции может достигать 128 или даже 256 бит.

Подход VLIW сильно упрощает архитектуру процессора, перекладывая задачу распределения вычислительных устройств на компилятор. Поскольку отсутствуют большие и сложные узлы, сильно снижается энергопотребление. В то же время, код для VLIW обладает невысокой плотностью. Из-за большого количества пустых инструкций для простаивающих устройств программы для VLIW-процессоров могут быть гораздо длиннее, чем аналогичные программы для традиционных архитектур.

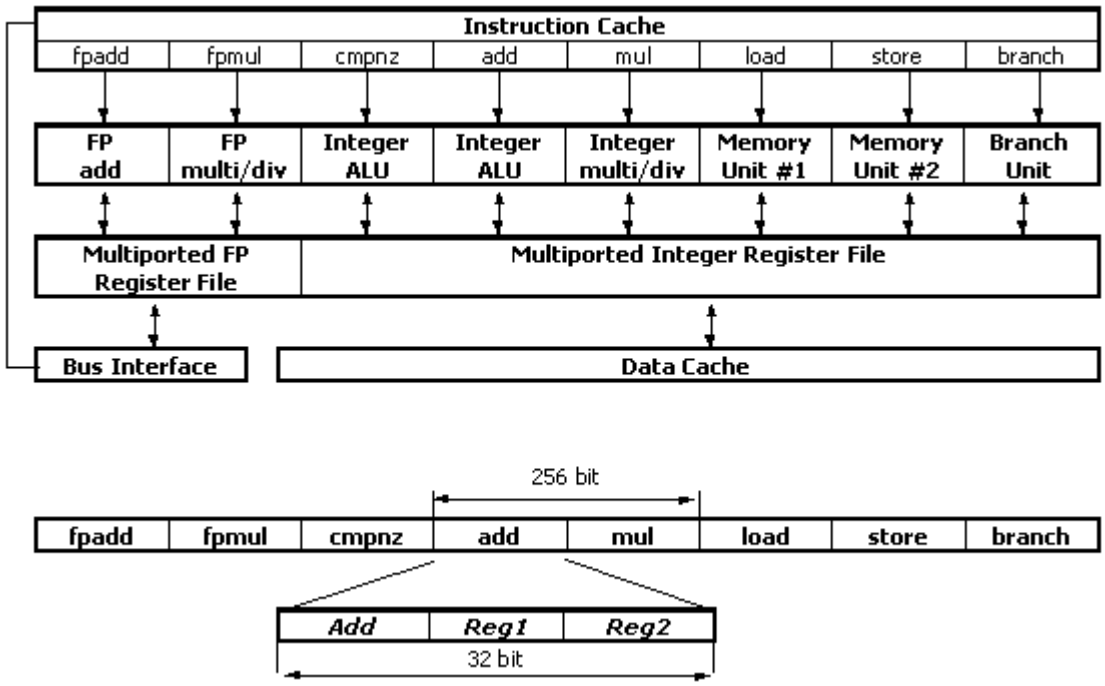


Рис. 5.5. Структура гипотетической инструкции VLIW [5.2]

Концепция VLIW появилась в начале 80-х гг. в Йельском университете, в 1984 г. разработчик концепции Дж. Фишер основал компанию Multiflow и в 1987 г. выпустил первый процессор MultiFlow 7/300. Процессор содержал по 2 АЛУ для целочисленных и дробных вычислений и блок ветвлений, его 256-битное командное слово содержало 8 кодов операций (4 для целых чисел или работы с памятью, 2 для дробных, 1 для ветвлений и 1 для служебной информации). Архитектура VLIW возникла в ответ на требования со стороны научно-технических организаций, где при вычислениях особенно необходимо большое быстродействие процессора. Со временем она стала популярна, т.к. сложность компиляторов и так значительно возросла, и перенос оптимизации на компилятор усложнял его не значительно. Основная сфера применения – встраиваемые сигнальные и векторные (для научных расчетов) процессоры и системы.

Примеры процессоров – Transmeta Crusoe, DSP C6000 фирмы Texas Instruments, Intel Itanium, графические процессоры Radeon R600.

§5.5. Архитектура POWER (1990) [5.3]

Разработанная в IBM архитектура POWER («Performance Optimization With Enhanced RISC») во многих отношениях представляет собой классическую RISC-архитектуру. Она придерживается наиболее важных отличительных особенностей RISC: фиксированной длины команд, регистр-регистровой архитектуры, простых способов адресации, простых (не требующих интерпретации) команд, большого регистрового файла и трехоперандного (неразрушительного) формата команд. Однако архитектура POWER имеет также несколько дополнительных свойств – «смешанные» и расширенные команды, расширенный регистр условий, а также отсутствие «задержанных переходов».

В рамках архитектуры POWER альянсом Apple, IBM и Motorola в 1991 г. был создан процессор PowerPC, успешно соперничавший с процессорами x86 до 2001 г., и применяющийся в серверах, майнфреймах и суперкомпьютерах IBM до сих пор.

При разработке процессора PowerPC компания IBM предложила идею CMP (Chip MultiProcessing), и, расположив в микросхеме два микропроцессора, создала первый многоядерный процессор.

Архитектура PowerPC положена в основу многоядерных процессоров игровых приставок шестого поколения (от Sony, Microsoft и Nintendo), как всем доступная, технологически более простая, нежели x86, и достаточно производительная архитектура. Она же положена в качестве аппаратной основы концепции Cell, которая рассматривается IBM важным шагом в развитии вычислительной техники.

Концепция Cell анонсирована компаниями IBM, Toshiba и Sony в 2001 г., а первый процессор для игровых приставок выпущен в 2005. Cell

использует в рамках одного кристалла одно управляющее RISC-ядро PPC970 и несколько (до 8) векторных SIMD ядер (рис. 5.6), каждое из которых ненамного быстрее процессора графической карты [5.4].

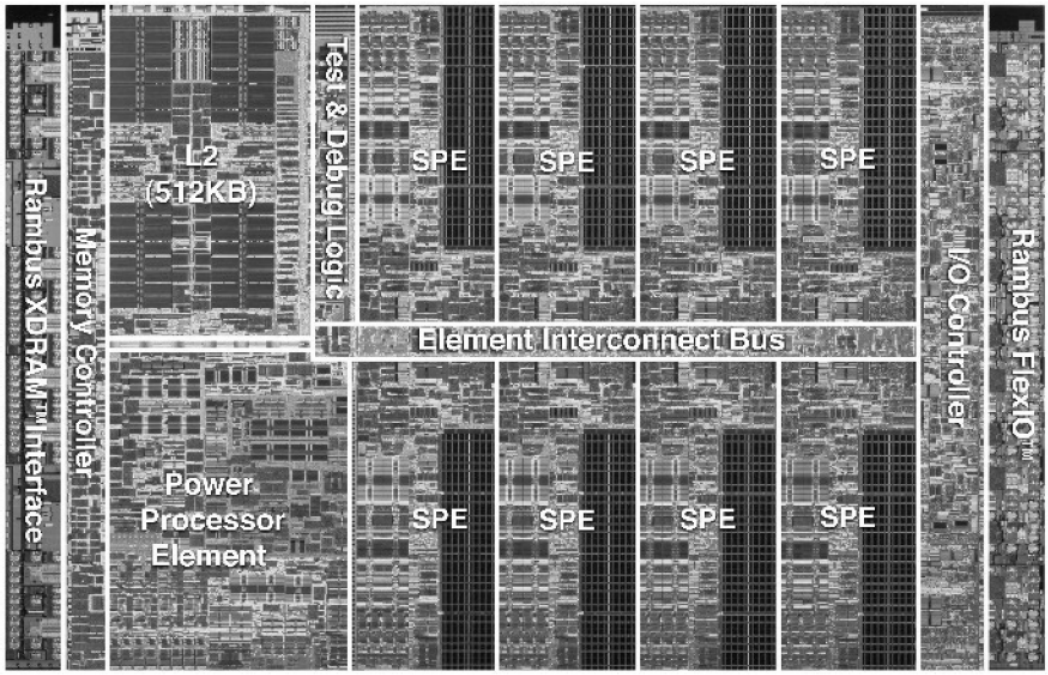


Рис. 5.6. Внешний вид кристалла микропроцессора CELL [5.4]

§5.6. Архитектура EPIC

В 1989 г. исследователи HP пришли к выводу, что архитектура RISC имеет явный предел суперскалярности. Число одновременно выполняемых инструкций, т.е. и число суперскалярных модулей, не может превышать 5–6 из-за того, что сложность буфера переупорядочивания инструкций увеличивается в геометрической прогрессии, при этом реальная загрузка VLIW процессоров составляет те же 5–6 блоков. В HP в альянсе с Intel началась разработка архитектуры EPIC (Explicitly Parallel Instruction Computing) на основе VLIW, приведшая к созданию процессоров IA-64.

Архитектура EPIC предусматривает явный параллелизм команд, и является дальнейшей разработкой VLIW – распараллеливанием команд занимается компилятор, а не процессор. В архитектуре сочетаются многие технологические решения, довольно разнотипные, которые в результате сочетания дают значительное повышение скорости обработки и решение некоторых проблем трансляции программ.

Команды IA-64 группируются по три в 128-битную «связку», также содержащую шаблон, указывающий зависимости между командами (рис. 5.7). Каждая связка соответствует набору из трех функциональных блоков процессора. Процессоры IA-64 могут содержать различное количество таких наборов блоков, тогда

процессору с N наборов будет соответствовать командное слово из N×3 команд (N связок). Таким образом, обеспечиваться масштабируемость.

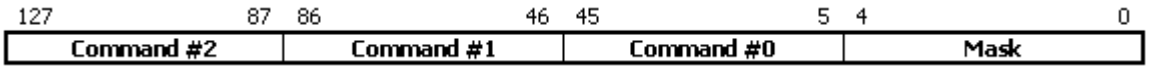


Рис. 5.7. Структура команды IA-64

В архитектуре возможно предикатное выполнение команд, исключающее переходы. При этом команды из разных частей условного ветвления снабжаются предикатными полями и запускаются параллельно. Также используется спекулятивное исполнение команд – вынесение команд загрузки далеко вперёд инструкций, использующих эти данные [5.5].

Поскольку архитектура массовых процессоров, основанная на Pentium Pro, оказалась надежнее и удобнее, архитектура EPIC за пределами серверных процессоров Itanium применения не нашла.

В настоящее время в качестве кандидатов на смену полупроводниковой технологии изготовления микропроцессоров, приближающейся к технологическому пределу по миниатюризации и быстродействию, рассматриваются разные технологии. Это процессоры с традиционной архитектурой, но построенные на базе оптических переключателей либо полученные путем самосборки молекул ДНК, и принципиально новые решения, такие как квантовые компьютеры.

ГЛАВА 6. СУПЕРКОМПЬЮТЕРЫ И СПЕЦИАЛИЗИРОВАННЫЕ ВЫЧИСЛИТЕЛЬНЫЕ СИСТЕМЫ

§6.1. Суперкомпьютеры

В общем случае суперкомпьютер – это компьютер, гораздо более мощный, чем доступные для большинства пользователей машины, находящийся на переднем крае производительности. Из-за большой гибкости самого термина до сих пор распространены довольно нечёткие представления о понятии суперкомпьютер. Термин «супервычисления» относили в 1920 г. к табуляторам IBM, а шутливая классификация Гордона Белла и Дона Нельсона, разработанная приблизительно в 1989 году, предлагала считать суперкомпьютером любой компьютер, весящий более тонны.

Ранние суперкомпьютеры в основном были обычными машинами, всего лишь оснащёнными быстрыми для своего времени скалярными процессорами. Появление рынка «суперкомпьютеров» связано с выпуском в 1976 г. компьютера Cray-1 (появление ILLIAC IV в 1975 г. успеха не имело). Эти суперкомпьютеры использовали передовые архитектурные решения, такие как конвейер, векторность и многопроцессорность, и отличались этим от обычных машин. Например, Cray-1, выпущенный под типично математические расчеты (загрузка массива, обработка, выгрузка), работал на частоте 80 МГц, содержал по 8 целочисленных (24 бит) и float-point регистров (64 бит), имел 12 параллельно работающих АЛУ (рис. 6.1), и обладал быстродействием 180 MFlops. ILLIAC IV обладал сходной производительностью, но в нем упор был сделан на многопроцессорность, а не векторность – одновременно работало 16 процессоров при меньшей частоте.

К середине 80-х несколько параллельно работающих векторных процессоров (от 4 до 16) стало стандартным суперкомпьютерным решением. Конец 80-х и начало 90-х годов охарактеризовались сменой направления развития от векторно-конвейерной обработки к большому числу параллельно соединённых скалярных процессоров.

Возможность параллельной работы различных устройств, совместно с использованием высокоскоростных элементов, стала основой ускорения основных вычислительных операций. Массивно-параллельные системы стали объединять в себе сотни и тысячи отдельных процессорных элементов, причём ими могли служить не только специально разработанные, но и общеизвестные и доступные в свободной продаже процессоры. Большинство массивно-параллельных компьютеров создавалось на основе мощных процессоров с архитектурой RISC, наподобие PowerPC или PA-RISC.

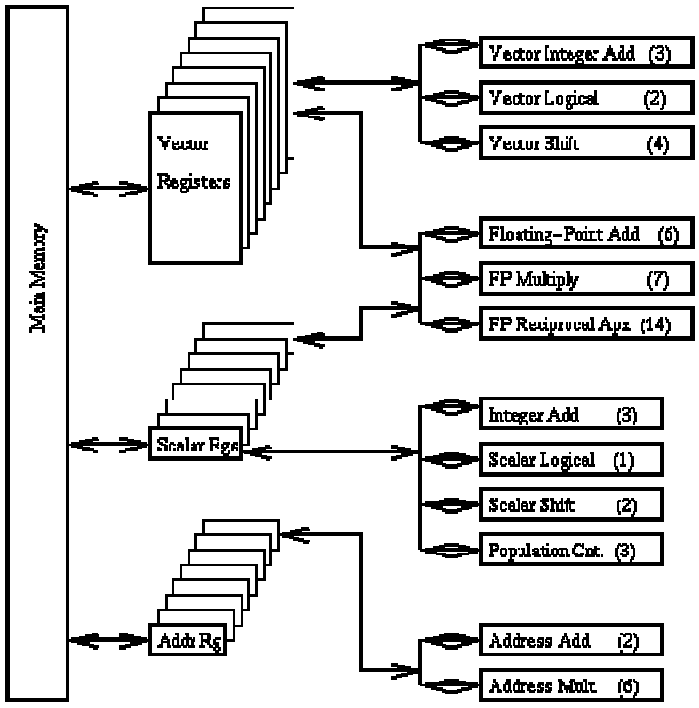


Рис. 6.1. Архитектура компьютера Cray-1, регистры и конвейеры (числа обозначают глубину конвейера) [6.1]

Взрывной рост количества задач, подходящих для параллельной обработки, приходится на конец XX века – это климатические, биологические, ядерные и пр. расчеты. Но сама параллельная обработка данных была изобретена еще в конце XVIII века, когда Гаспар де Прони организовал процесс расчета логарифмических и тригонометрических таблиц. Он свел сложные задачи к набору рутинных операций для 20 вычислителей. Вычислители находились под контролем образованных «технологов», и от них требовалось только аккуратно складывать и вычитать. Математическим обеспечением процесса занимались несколько выдающихся математиков. Такой подход сохранялся до середины XX века, когда рассчитывали первые ядерные бомбы.

Своеобразным «пионером» в параллельной обработке данных может считаться и академик А.А. Самарский, выполнявший в начале 50-х математическое моделирование ядерных взрывов. Самарский проводил расчеты, в частности, эволюции взрывной волны, методом сеток. При этом в узлы сетки садились девушки с арифмометрами, передающие друг другу данные на словах и откладывающие необходимые цифры на арифмометре. Точность оказалась невелика, т.к. узлов в сетке было мало, а время счета – большим. Но первые реалистические расчёты макрокинетики цепной реакции ядерного взрыва, приведшие к практически важным оценкам мощности ядерных боеприпасов, были выполнены.

§6.2. Классификация суперкомпьютеров [6.2, 6.3]

Основная классификация суперкомпьютерных систем основана на делении, предложенном в 1966 г. Флинном. На основе числа потоков инструкций (I) и потоков данных (D) выделяют четыре класса архитектур: SISD, MISD, SIMD и MIMD. Суперкомпьютерам в этой классификации соответствуют типы SIMD («один поток инструкций, много потоков данных», это векторные процессоры) и MIMD («много потоков инструкций, много потоков данных», это многопроцессорные системы). Большой вклад в производительность вносит и MISD – конвейерная организация. Но классификация Флинна не делает различия по другим важным для вычислительных моделей характеристикам, например, по уровню «зернистости» параллельных вычислений и методам синхронизации, которые становятся существенными при большом количестве одновременно работающих процессоров.

Классический тип суперкомпьютерной архитектуры использует общую оперативную память, обращение к которой осуществляется через системную шину. Это упрощает программирование, но с ростом числа процессоров наличие общей памяти приводит к возрастанию нагрузки на системную шину, которая является уязвимым местом такой архитектуры при 8 и более процессорах.

Параллельная архитектура (векторная SIMD и матричная MIMD) позволяет избежать проблем с системной шиной, т.к. каждый процессор снабжается своей локальной памятью, а доступ к чужой памяти осуществляется через сеть связи, объединяющая процессоры в систему. Это позволяет строить суперкомпьютеры из тысяч процессоров. Но возникает сложность программирования, особенно для задач, которым необходима память, превышающая размер локальной оперативной памяти одного процессора. Синхронизация также затруднена.

В 90-е годы стала широко использоваться массивно-параллельная архитектура (рис. 6.2), отличающаяся хорошей масштабируемостью. Она состоит из вычислительных узлов, представляющих собой матрицу (или решетку) с несколькими процессорами и общей для них памятью. При этом вычислительные узлы объединяются в единую систему с помощью сети связи, которая обслуживает запросы на доступ в память других узлов.

В эти же годы, в связи с распространением персональных компьютеров и развитием сетевых технологий, развивается кластерная архитектура. Такие системы являются самыми дешевыми, поскольку собираются на базе стандартных комплектующих элементов. В середине 2000-х гг. кластерная архитектура в списке Top-500 обходит по популярности массивно-параллельную архитектуру. Тогда же вместо традиционных процессоров начинают применять процессоры видеокарт (технология GPGPU), которые в ряде задач оказываются быстрее.

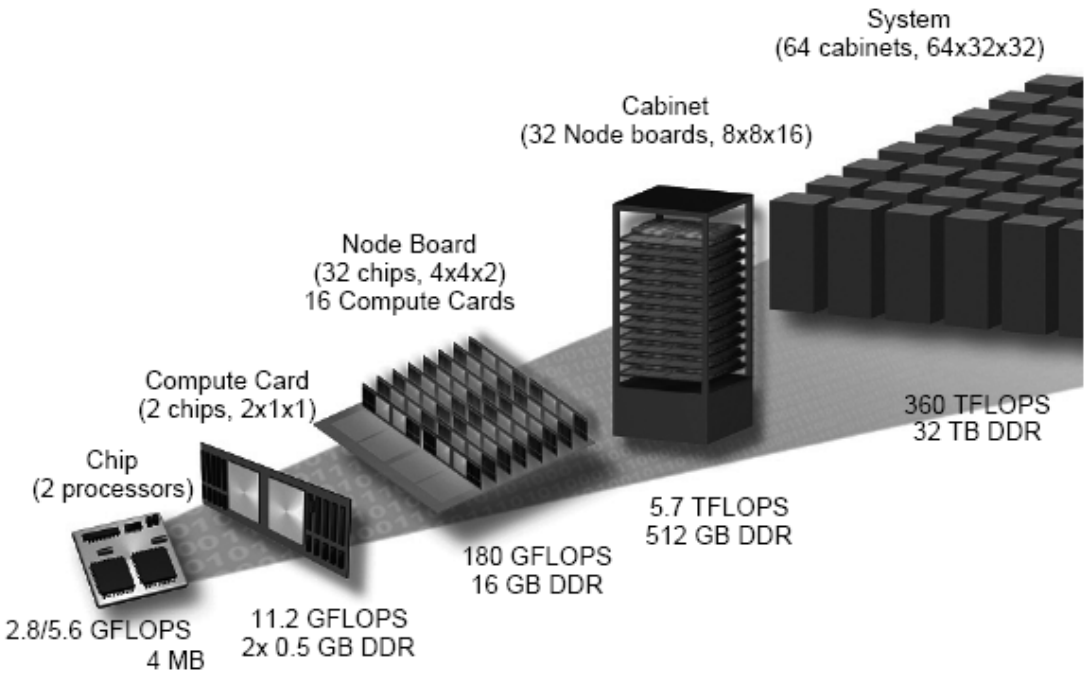


Рис. 6.2. Структура массивно-параллельных суперкомпьютеров IBM BlueGene/L. Компьютер имеет масштабируемую сотовую архитектуру и собирается из однотипных стоек. Каждая стойка объединяет до 1024 вычислительных узлов [5.4]

§6.3. Отечественные суперкомпьютеры [6.4]

К числу суперкомпьютеров можно отнести советскую БЭСМ-6, но также, как и в случае с Cray-1, отчет лучше начинать с определенного уровня архитектурных решений. Такой точкой отсчета является разработанный в 1967 г. М.А. Карцевым проект многомашинного комплекса М-9, в состав которого входила и векторная числовая машина. Комплекс в производство не пошел, но лег в основу многопроцессорной системы М-10, производство которой началось в 1973 г. и продолжалось до конца 80-х.

Производительность системы была 5 млн. оп/с, М-10 могла вести параллельную обработку данных различных форматов, динамически изменяя кластеризацию процессоров для соответствия формата данных. Распараллеливание операций осуществлялось компилятором. Для комплекса был разработан свой язык программирования и операционная система. Уступая по производительности Cray-1 из-за отставания элементной базы, М-10 не уступал по архитектурным возможностям. По производительности М-10 превосходила БЭСМ-6 в 4 раза, а старшие модели ЕС ЭВМ (т.е. и System/360) примерно в 6 раз.

В 1983 году начался выпуск многопроцессорной векторно-конвейерной системы М-13, построенной на базе больших интегральных схем и выполнявшей до 2,4 млрд. оп/с. Вычислительные комплексы М-10 и М-13 применялись в составе средств предупреждения о ракетном нападении и до 90-х гг. часть информации о них была закрытой.

§6.4. Специализированные вычислительные системы [6.5]

Классификация специализированных вычислительных машин

Компьютеры, предназначенные для выполнения специальных задач в условиях, существенно отличающихся от обычных, называют специализированными. В большинстве случаев это военные компьютеры. К ним, в первую очередь, предъявляются требования надежности и безотказности в сложных условиях работы, в то время как от обычных компьютеров обычно требуется высокое быстродействие и низкая цена. По климатическим и механическим условиям работы выделяют три группы таких компьютеров:

•стационарные – для работы в помещениях или на земле.

•транспортируемые – включаются в работу после установки на позиции, перевозятся воздушным, железнодорожным, автомобильным транспортом.

•бортовые – установленные на подвижном объекте и работающие в процессе их перемещения.

В свою очередь бортовые компьютеры по месту их установки разделились на следующие группы: 1) возимые; 2) самолетно-космические; 3) ракетные; 4) морские.

Бортовые компьютеры и первый микропроцессор

Баллистические расчеты были одной из причин появления компьютеров, а одним из самых первых компьютеров вообще является бортовой компьютер ракеты «Фау-2». Автономный полет требует проведения большого объема расчетов, поэтому в самолетную и ракетную технику, начиная с 1960-х гг., вместо аналоговых и дискретных счетно-решающих устройств стали внедряться полноценные бортовые компьютеры. Возможность решения более сложных алгоритмов позволяла существенно повысить точность стрельбы и увеличивала живучесть системы.

В 1968 г., на год раньше появления компании Intel, американские инженеры Рэй Холт и Стив Геллер создали 20-разрядный чип SLF (Special Logic Function), который содержал арифметическое вычислительное устройство, декодер инструкций и поддерживал управляемую логику. Чип SLF являлся основой бортового компьютера CADC (Central Air Data Computer), предназначенного для новейшего истребителя F-14 с изменяемой геометрией крыла.

До того на самолеты устанавливали преимущественно механические вычислители, которые не могли обеспечить необходимый для F-14 объем и точность вычислений. Обычные компьютеры, вроде 12-разрядного DEC PDP-8, во-первых, также не обеспечивали требуемую

точность, а во-вторых, могли работать только в комнатных условиях.

Созданный на базе SLF миниатюрный многоцелевой компьютер CADC содержал три параллельно работающих чипа SLF, поддерживал 20-разрядные слова, решал задачи в режиме реального времени, был оптимизирован для одновременного выполнения нескольких интенсивных вычислительных процессов, успешно функционировал во время полета, потреблял мало энергии и легко стыковался с другим оборудованием. Для него также придумали и создали чипы памяти и фактически первыми ввели и реализовали концепцию математического сопроцессора, ускорявшего операции умножения и деления. Влияние на микропроцессорную индустрию этот проект не оказал как из-за секретности, так и из-за уверенности руководства компаний в бесперспективности этих разработок.

Ракетные и космические бортовые ЭВМ

Рассмотрим принципы обеспечения надежности бортовых компьютеров на примере вычислительного комплекса, обеспечивавшего работу автономной инерциальной системы управления ракеты Р-36М «Воевода» (1975). Расчет ведут полностью параллельно 3 бортовые ЭВМ. Проверяющее устройство сравнивает результаты вычислений всех трех ЭВМ и данные с трех каналов информации, в случае обнаружения отказа вычислительная машина или сбойный канал блокируются. Все линии связи продублированы, надежная элементная база обеспечивает срок жизни до 25 лет. Быстродействие системы относительно невысокое – 100 тыс. оп/с (у новых модификаций 2 млн. оп/с), ОЗУ 1 кБ, ПЗУ 32 кБ (521 кБ), частота 2,5 МГц (8 МГц) [6.6].

Схожие принципы просматриваются в организации бортового компьютера космического корабля «Буран» (1988), ставшего первым кораблем, совершившим полет в космос и приземление полностью в автоматическом режиме. Корабль разрабатывался для полетов в беспилотном режиме, и требовалось заранее предусмотреть все возможные неисправности и сложные ситуации. Проблема обеспечения надежности также решена с помощью аппаратной избыточностью – применено четыре вычислительных машины. Такая система обеспечивает гарантированное возвращение с орбиты при отказе до двух систем, которые в случае сбоя отключаются. При отказе третьей «случайно» отключается одна из оставшихся, тогда вероятность возвращения оценивается в 50%. Все компьютеры работают синхронно по единой программе, и управляются блоком синхронизации, работающим от пяти синхронно работающих тактовых генераторов.

В бортовой вычислительной системе Международной космической станции (2000) также большое внимание уделено резервированию. Главные компьютеры имеют трехкратное аппаратное резервирование,

компьютеры второго уровня – двукратное. В российском сегменте все компьютеры имеют трехкратное резервирование. Задача обеспечения отказоустойчивости возложена и на специальные программные средства. Процессорные платы стандартны и выполнены на процессоре i386. Быстродействия достаточно для выполнения поставленных задач, при этом минимально потребление энергии. Похожая ситуация со стандартной шиной обмена – она весьма медленна (1 Мбит/с), но имеет хорошие характеристики по резервированию. Так, каждая шина состоит из двух каналов, проложенных по разным сторонам модулей.

В качестве примера возимой вычислительной системы можно рассмотреть ЭВМ мобильных систем ПВО С-300 (1975), создававшихся под руководством академика В.С. Бурцева [6.7]. Первыми системами были 5Э261 и 5Э262, затем было выпущено несколько модификаций на другой элементной базе и с большим быстродействием. Но их все объединяют общие принципы работы. Это многопроцессорность (двух- и трехпроцессорность), высокоэффективное резервирование, модульность исполнения (обеспечивает ремонт в полевых условиях заменой блоков) и возможность восстановления процесса управления при сбоях и отказах аппаратуры. Также системы приспособлены для работы в широком диапазоне климатических и механических воздействий, и обеспечиваются развитым математическим обеспечением и системой автоматизации программирования. Вес первых модификаций достигал 3 тонн.

Отличия отечественных специализированных ЭВМ

Развитие американских военных компьютеров пошло в направлении использования схем универсальных компьютеров, что приводило к более высоким требованиям к их скорости, объемам памяти и надежности. В Советском Союзе значительное число типов спецкомпьютеров не следовало всем фон-неймановским принципам построения. В этих компьютерах память для команд и память для чисел были независимы. Такое построение увеличивало производительность и исключало всякие случайности, связанные с программами (например, с возможностью появления вирусов), а также повышало их защищенность от несанкционированных действий.

Ввиду невозможности прямой конкуренции по производительности из-за отставания элементной базы, широко использовались оригинальные схемотехнические решения, позволяющие поднять производительность, и широко применялось низкоуровневое программирование. Отставания по уровню внедряемых архитектурных решений не наблюдалось. Большинство специальных компьютеров, созданных в СССР, соответствовало по современной терминологии архитектуре RISC.

ГЛАВА 7. КОМПЬЮТЕРНЫЕ СЕТИ

§7.1. Сети 1950-х. Идеи П. Бэрена

Сети начального периода развития иногда представляли собой пару компьютеров, соединенных линиями связи, но чаще – один крупный компьютер, связывающий удаленные узлы. Связь между несколькими компьютерами можно было организовать через телефонную сеть посредством модемов, которые начали появляться в 50-е гг. (коммерческие модели AT&T стали доступны в 1962 г.). Но такая связь была коммутируемой, должна была устанавливаться напрямую между компьютерами, а общепринятые правила обмена (протоколы) отсутствовали. Все это не позволяло организовать большие и надежные сети.

События, способствующие развитию компьютерных сетей

•1957 – запуск спутника в СССР, продемонстрировавший возможность доставки ядерного заряда в любую точку мира. Такое оружие делало уязвимым существующие сети на основе универсальных вычислительных машин, главным недостатком которых являлось то, что при разрушении центрального узла любые соединения становились невозможны.

•1958 – создание NORAD (North American Aerospace Defense Command, Североамериканское командование по защите воздушного пространства), единого компьютеризированного подземного центра управления в горах Колорадо-Спрингс. Центр должен обеспечивать своевременное оповещение о возможной опасности и обеспечивать управление в критической ситуации.

•1960-е – создание термоядерных зарядов большой мощности (40–50 мегатонн), способных разрушать подземные центры управления. Например, т.н. «Царь-бомба» (1961).

Возможность полного разрушения центров управления резко снижало эффективность системы NORAD, не говоря уже об обычных системах на основе универсальных компьютеров. Одним из вариантов решения проблемы неустойчивости сети стало применение децентрализованной структуры.

Идеи П. Бэрена (1964) [7.1]

Работая над безопасностью технологий связи с 1959 г., сотрудник корпорации RAND Пол Бэрен в 1962 г. предложил идею распределенных сетей, обеспечивающих устойчивость связи при ядерном ударе. Под распределенной сетью Бэрен понимал децентрализованную сеть со множеством путей между двумя точками связи (рис. 7.1). В 1963 г. идея

была предложена корпорацией Пентагону и отвергнута им как «смехотворная». В 1969 г., после очередного рассмотрения этой идеи, Пентагон диаметрально сменил точку зрения.

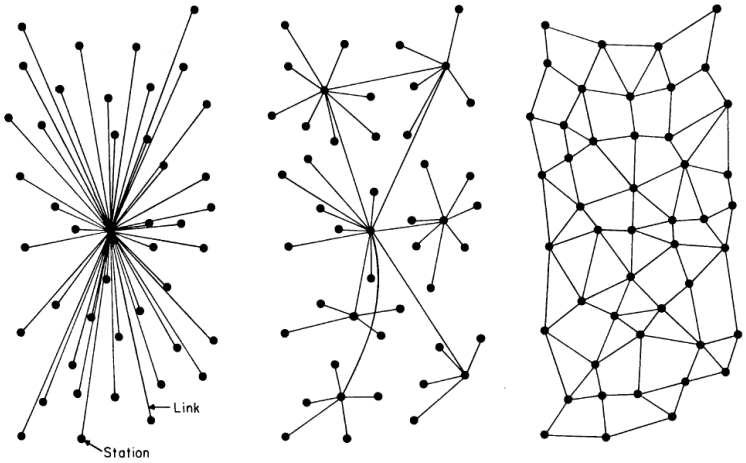


Рис. 7.1. Типы сетей по П. Бэрену – централизованная, децентрализованная и распределенная

Бэрен исследовал избыточность узлов связи, эквивалентную отношению количества связей к узлам в бесконечном массиве станций (рис. 7.2). При уровне выше трех становилось возможным альтернативное построение сетей, но самое главное – при такой избыточности разрушение даже половины узлов распределенной сети не мешает работе оставшейся части сети.

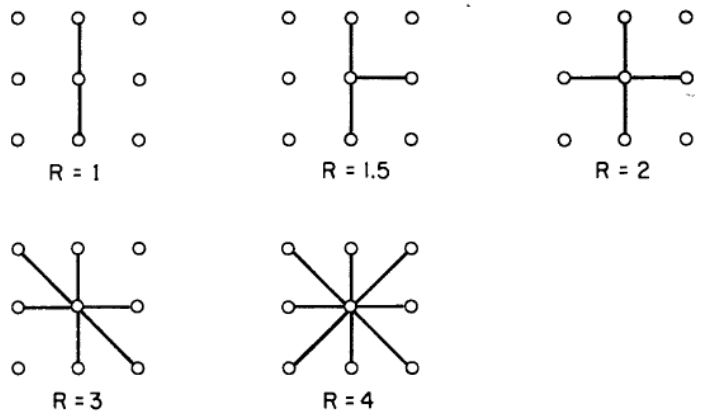


Рис. 7.2. Узлы связи с разным уровнем избыточности (R)

П. Бэрен предполагал совместное использование в составе сети разных видов связи, основанных на разных физических принципах и обладающих разной скоростью передачи данных. Это могла быть телефонная, телевизионная, кабельная, оптоволоконная, микроволновая (радиорелейная) или спутниковая связь (7.3).

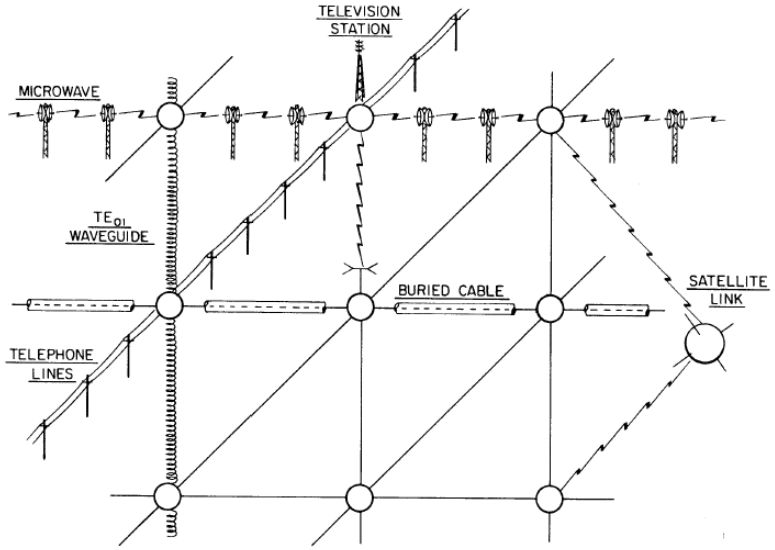


Рис. 7.3. Разнообразие каналов связи

Сообщение пользователя в такой разнородной сети должно быть разбито на стандартные блоки с последующей передачей через сеть (рис. 7.4). Несмотря на разные скорости и разные принципы передачи данных (т.к. передача идет по разнородным каналам связи), сам блок данных остается неизменным. Для согласования таких каналов применяются буферы данных. Время прохождения пакета может значительно превышать время его передачи.

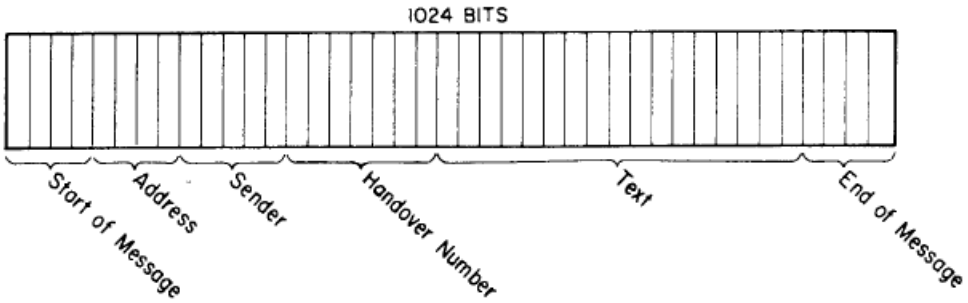


Рис. 7.4. Стандартный блок сообщения (пакет)

Схожие работы в 1965 г. вел в Англии Дональд Дэвис в Англии (Britain’s National Physics Lab), он же ввел в обиход термин «пакет».

§7.2. ARPAnet (1969) [7.2, 7.3]

В 1958 г. в США было создано Агентство передовых исследований ARPA (Advanced Research Projects Agency). Задачами агентства являлось финансирование дорогостоящих разработок и внедрение фундаментальных исследований в военное использование. В 1966-м агентство совместно с рядом университетов стало обсуждать программу совместного использования ресурсов сетей, при этом целью ARPA было повышение производительности компьютеров и децентрализация

хранения информации, а интересом правительства и военных – поиск способа хранения и распространения информации, работающего в условиях ядерного нападения.

Агентство планировало связать все научные и правительственные заведения страны, но первоначально ограничилось 4 компьютерами – по числу изготовленных интерфейсных процессоров (Interface Message Processor, IMP), обеспечивающих пакетный обмен. Демонстрация работы такой сети состоялась в 1969 г., сеть связала 4 компьютера, работающих под управлением разных операционных систем и находящиеся в нескольких университетах на востоке США, при этом для связи использовались традиционные телефонные линии (рис. 7.5).

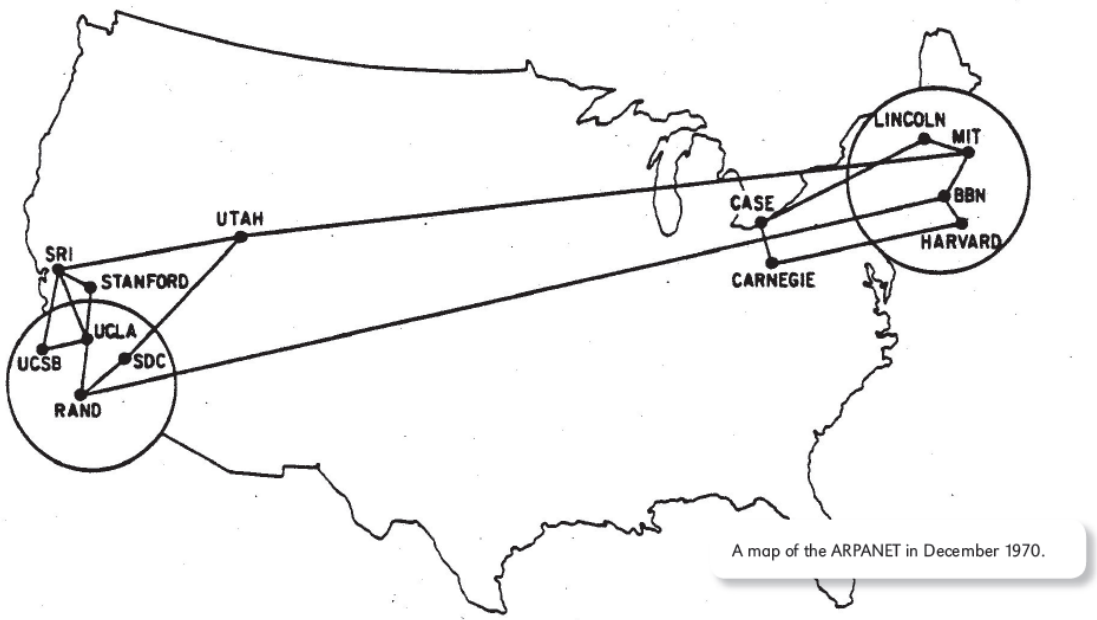


Рис. 7.5. Сеть ARPAnet в 1970 г. Четыре самых первых узла сети – отделения Калифорнийского университета в Санта-Барбаре (UCSB) и Лос-Анджелесе (UCLA), Стэндфордский исследовательский институт (SRI) и университет штата Юта (UTAH) [7.2]

Хотя исследования корпорации RAND и первоначальные варианты агентства ARPA рассматривали варианты создания сети, устойчивой к воздействию ядерного удара, по заявлениям разработчиков сети собственно к ARPAnet это прямо не относится. Разработчики главной задачей видели обеспечение надежности в условиях, когда узлы переключения и связи сети были и так не надежны, даже без ядерного нападения. Основной их идеей было предоставление доступа множеству исследователей, географически отделенных от университетов, к ограниченному числу мощных университетских компьютеров.

Сеть стала первой в мире сетью с пакетной коммутацией. Для её функционирования потребовалась разработка специализированных интерфейсных процессоров, что дало толчок к развитию сетевого

оборудования. В 1972 г. сеть была преобразована в публичную, что было необычно для такой передовой технологии. К основным достижениям, полученным в ходе создания сети, относят удаленный доступ в систему, передачу файлов по сети и электронную почту. Создание электронной почты позволило координировать действия большого числа разработчиков.

§7.3. Internet

Со временем сеть ARPAnet охватила большое количество компьютерных центров страны и была соединена с другими сетями – в т.ч. на Гавайях и в Европе, но она оставалась сетью для служебного пользования университетов, корпораций и правительства, и многие не знали о её существовании. К 1983 г. завершился перевод сети на протокол TCP, а саму сеть все чаще стали обозначать термином, возникшим еще в 1974 г. и обозначающим глобальную сеть – «Internet». В том же 1983 г. военная часть сети была отделена. В 1980-х гг. стали проявляться недостатки архитектуры сети, построенной на IMP, и пользователей начали переводить на более перспективную сеть NSFNet, и в 1990 г. проект был закрыт как достигший своих целей.

В 1985 г. для обеспечения доступа исследователей к суперкомпьютерным центрам была организована межуниверситетская сеть NSFNet (National Science Foundation Network, Компьютерные сети национального фонда науки), также включившая в себя ряд более мелких сетей. Она была организована по принципу ARPANet, но работала на больших скоростях. Число пользователей сети лавинообразно увеличивалось, повышалась пропускная способность сети, это привело к полному вытеснению ARPANet, а термин «интернет» стали относить уже к NSFNet.

В 1990-е гг. значительная часть каналов сети была передана в общедоступную коммерческую эксплуатацию, оставшаяся часть превращена снова в межуниверситетскую. В рамках закрытой части в настоящее время тестируются новые разработки (Интернет-2).

Отцы Интернета [7.4]

Как и у компьютера, у Интернета много «отцов», внесших вклад на разных этапах его создания:

- В. Буш – описал прообраз гипертекстового письма (система Memex);
- Paul Baran – разрабатывал распределенные сети, пакетную коммутацию;
- Donald Davies – работы по распределенным сетям в Великобритании;
- Len Kleinrock – теория массового обслуживания (пакетный обмен

в сети), иерархические стеки протоколов, участвовал в ARPANET и Internet;

•J.C.R. Licklider (Джозеф Карл Робнетт Ликлайдер) – в 1962г. предложил концепцию «галактической сети», директор по развитию компьютерного отдела ARPA;

•Bob Taylor – инициатор проекта ARPAnet как гражданский куратор от министерства обороны, сотрудник Xerox PARC;

•Larry Roberts – «отец ARPANET», руководитель группы инженеров-создателей ARPANET;

•Vinton Cerf – сотрудник DAPRA и CNRI, учёный в области теории вычислительных систем, один из разработчиков протокола TCP/IP, часто именуется «отцом интернета»;

•Robert Kahn – сотрудник DAPRA и CNRI, совместно с Винтоном Серфом – изобретатель протокола TCP/IP;

•Д. Энгельгарт – разработал гиперсреду NLS (первая успешная реализация гипертекста), провел первую мультимедийную конференцию (1968 г.).

§7.4. ALOHAnet (1970) [7.5, 7.6]

В 1970 г. профессор Норман Абрамсон, работавший над вопросами радиолокации и цифровой радиосвязи в Беркли и Стэнфорде, перешел на работу в Гавайский университет и предложил организовать сеть передачи данных, связывающую университетские кампусы, расположенные на островах архипелага, вытянувшегося на 600 км в длину. В том же году сеть заработала, став первой беспроводной сетью с пакетной передачей данных. Название сети было взято от «Алоха» – неофициального названия штата Гавайи, одно из местных значений которого одновременно обозначает «здравствуйте», «до свидания», «я люблю вас» и просто пожелание мира и радости. Это была первая в мире беспроводная радиосвязь с использованием пакетной передачи данных, и первая в мире сеть с множественным доступом к среде.

Первый протокол работы сети имел два основных пункта:

– если у вас есть данные, вы их посылаете,

– если передача сообщения совпала с другой передачей, ваше сообщение должно быть передано позже.

ALOHAnet была выполнена с использованием стандартных телеграфных радио модемов со скоростью 9600 бод и работала на частоте около 410 МГц. Топологией сети была «звезда», центральный компьютер университета получал сообщение от какого-нибудь узла, получившего доступ к эфиру, а затем ретранслировал сообщение на все узлы сети на другом частотном канале. Это позволило минимизировать вероятность столкновений. Пакеты, используемые для связи, имели заголовок 32 бит, могли нести до 80 байт данных и снабжались 16-

битными полями для проверки четности.

Из-за столкновения пакетов («коллизий») сеть имела предельную пропускную способность 18% от максимума. Математическое моделирование процесса получения случайного пакета как случайного события с распределением дало тот же результат. Усовершенствованием к оригинальному протоколу стало введение дискретных моментов времени, в которые станция может послать пакет. Это уменьшило столкновение пакетов и удвоило максимальную пропускную способность до 37%. Стоит отметить, что достигнутые характеристики не очень отличаются от современных беспроводных систем типа Wi-Fi.

По мере роста числа пользователей система становилась неэффективной и требовала других способов согласования порядка начала передачи данных. Позже, в качестве одного из таких решений, а также для увеличения покрытия сети, были добавлены ретрансляторы, которые также выступали в качестве центров. В 1972 году ALOHAnet стала первой сетью, подключенной к ARPAnet, следом последовали и другие сети.

Схема работы ALOHA была проста, она наглядно показала, что можно организовать работающую сеть не решая проблему «коллизий», которые раньше приходилось преодолевать вручную повторными попытками установить связь. Это повлияло на разработки как беспроводных, так и проводных сетей.

Различные версии протокола ALOHA в дальнейшем использовались в спутниковой связи, а также послужили основой для сетей типа GSM. Работа в ALOHAnet была частью диссертации будущего создателя сети Ethernet Роберта Меткалфа, вдохновившая его на разработку протокола доступа к среде (MAC, Media Access Control) с большей пропускной способностью – CSMA (Carrier Sense Multiple Access).

§7.5. Локальные вычислительные сети.

Из технологий локальных вычислительных сетей отметим Token ring (1984), Ethernet (1976) и ARCNET (1977). Исторически первым, под влиянием ALOHAnet, возникли идеи Ethetnet. Роберт Меткалф, сотрудник корпорации Xerox PARC, представил его идеи еще в 1973 г., но реализовать их в виде стандарта передачи данных, при поддержке компаний DEC, Intel и Xerox, ему удалось только в 1980 г.

Технология ARCNET была анонсирована в 1977 г. и стала первым способом организации локальной вычислительной сети, связывающей разнотипные компьютеры. До неё были доступны только локальные сети, организованные между однотипными вычислительными машинами – это такие сетевые решения, как DECnet (DEC) или SNA (Systems Network Architecture, IBM). Сеть организовывалась по топологии

«звезда», позволяла недорого организовать локальную сеть и в середине 1980-х гг. стала очень популярна. Затем она была вытеснена сетями Ethernet, обеспечивавшими большую скорость.

Разновидности сетей, организованных по топологии «кольцо» с использованием маркера, стали появляться с 1981 г., самая известная из них – разработанная IBM в 1984 г. сеть «Token ring». Являясь детерминисткой сетью, и считаясь более надежной, она уступила той же Ethernet по скорости и по цене.

Сети, первоначально появившиеся как средство связи, постепенно становятся и средством вычислений – это распределенные вычисления, нейронные сети, и не только. Так, в 2010 г. Би-Би-Си опубликовала сообщение под заголовком «Мозг больше похож на Интернет, чем на компьютер», ссылаясь на исследования, опубликованные в американском академическом журнале PNAS [7.7]. Основные положения статьи основаны на демонстрации петель и связей в мозгу в сетевом стиле, наличие обратных связей и перекрестных связей между разными областями мозга. Авторы сравнивают такую модель с иерархической моделью мозга, аналогичной модели компьютера с центральным процессором и иерархией периферийных устройств. Разумеется, такое сравнение находится на уровне начала XX века, поскольку идеи обратной связи и перекрестного взаимодействия между различными отделами мозга в нейрологии не новы, и применяются в нейросетях. Но топология связей и узлов сложной сети, вроде Интернета, действительно напоминает связи в мозгу (рис. 7.6).

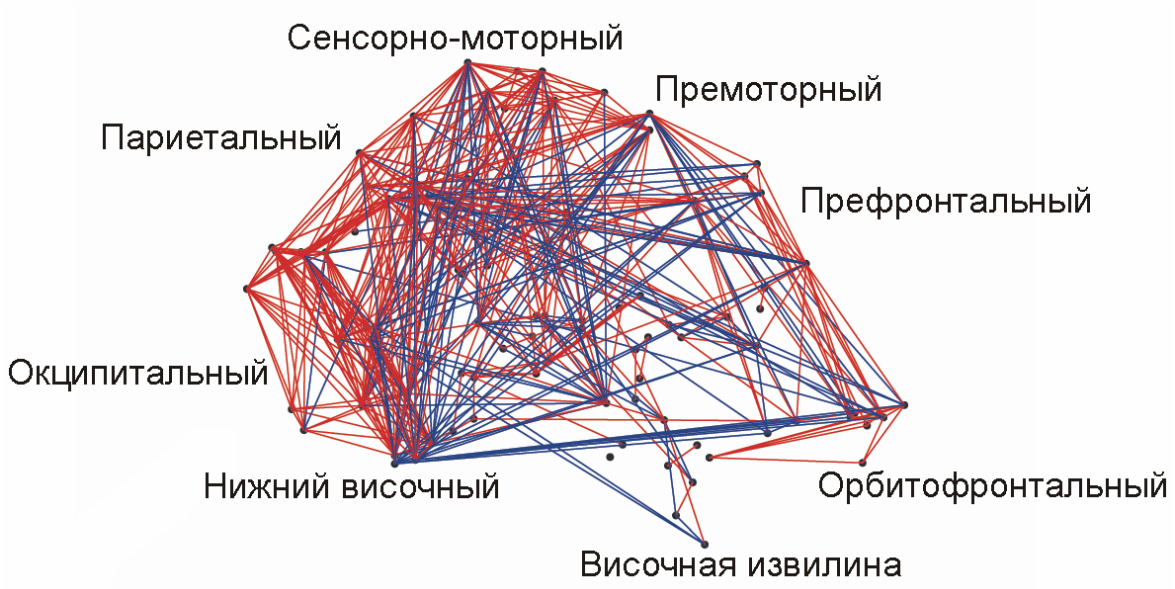


Рис. 7.6. Функциональная схема связей отделов головного мозга [7.8].

ГЛАВА 8. ОПЕРАЦИОННЫЕ СИСТЕМЫ

§8.1. Эволюция операционных систем [8.1]

Операционная система (ОС) – это комплекс взаимосвязанных программ, выполняющий набор таких функций, как обеспечение выполнения других программ, распределение ресурсов, ввод-вывод данных, обеспечение безопасности и пр. Каждая из приведенных функций обычно реализована в виде подсистемы, являющейся структурным компонентом ОС. Эти компоненты не были изначально придуманы как составные части операционных систем (часто это были отдельно поставляемые утилиты), они появлялись в процессе развития операционных систем и реализовывались по-разному.

Историю развития вычислительных машин и операционных систем обычно рассматривают вместе, потому что аппаратное и программное обеспечение эволюционировало совместно, оказывая взаимное влияние друг на друга. Появление новых технических возможностей приводило к прорыву в области создания удобных, эффективных и безопасных программ, а свежие идеи в программной области стимулировали поиски новых технических решений. Опираясь на этапы развития компьютеров, можно выделить следующие периоды в развитии операционных систем:

- 1 период. Ламповые машины. Операционных систем нет.
- 2 период. Компьютеры на основе транзисторов. Пакетные операционные системы.
- 3 период. Компьютеры на основе ИС. Первые многозадачные ОС.
- 4 период. Персональные компьютеры. Классические, сетевые и распределенные системы.

Первый период (1945–1955). Операционных систем нет

В середине 40-х, когда были созданы первые ламповые вычислительные устройства, и в проектировании, и в эксплуатации, и в программировании вычислительной машины участвовала одна и та же группа людей. Программирование осуществлялось исключительно на машинном языке, все задачи организации вычислительного процесса решались вручную каждым программистом с пульта управления. Программа загружалась в память машины с помощью панели переключателей, позднее – с колоды перфокарт.

Вычислительная система выполняла одновременно только одну операцию (ввод-вывод или собственно вычисления), сами программы выполнялись строго последовательно. Отладка программ велась с пульта управления с помощью изучения состояния памяти и регистров машины. В конце этого периода появляется первое системное программное обеспечение: в 1951–1952 гг. возникают прообразы первых

компиляторов с символических языков (Fortran и др.), а в 1954 г. Nat Rochester разрабатывает Ассемблер для IBM-701. Операционные системы отсутствовали.

Второй период (1955–1960). Пакетные операционные системы

С середины 50-х годов начался следующий период в эволюции вычислительной техники, связанный с переходом на новую элементную базу – полупроводниковые элементы. Применение транзисторов вместо часто перегоравших электронных ламп повысило надежность компьютеров и существенно увеличило время безотказной работы. Теперь на них можно было возложить выполнение практически важных задач. Снизилось потребление электроэнергии, уменьшились размеры, снизилась также стоимость эксплуатации и обслуживания вычислительной техники. Началось использование компьютеров коммерческими фирмами. Одновременно наблюдается бурное развитие алгоритмических языков (Lisp, COBOL, ALGOL-60, PL-1 и т.д.). Появляются первые настоящие компиляторы, редакторы связей, библиотеки математических и служебных подпрограмм. Упрощается процесс программирования. Происходит разделение персонала, ранее ответственного за весь цикл разработки и использования компьютеров, на программистов и операторов, специалистов по эксплуатации и разработчиков вычислительных машин.

Изменяется сам процесс прогона программ. Теперь пользователь приносит программу с входными данными в виде колоды перфокарт и указывает необходимые ресурсы. Такая колода получает название задания. Оператор загружает задание в память машины и запускает его на исполнение. Полученные выходные данные печатаются на принтере, и пользователь получает их обратно через некоторое время. Смена запрошенных ресурсов вызывает приостановку выполнения программ, в результате процессор часто простаивает.

Для повышения эффективности использования компьютера задания с похожими ресурсами начинают собирать вместе, создавая пакет заданий. Так появляются первые системы пакетной обработки, которые просто автоматизируют запуск одной программы из пакета за другой и тем самым увеличивают коэффициент загрузки процессора. При реализации систем пакетной обработки был разработан формализованный язык управления заданиями, с помощью которого программист сообщал системе и оператору, какую работу он хочет выполнить на вычислительной машине. Системы пакетной обработки стали прообразом современных операционных систем, они были первыми системными программами, предназначенными для управления вычислительным процессом.

Третий период (1960-е–1980). Компьютеры на основе интегральных микросхем

Переход от отдельных полупроводниковых элементов к интегральным микросхемам повысил быстродействие компьютеров и сделал вычислительную технику еще более надежной и дешевой. Выросли сложность и количество задач, решаемых компьютерами.

Архитектура компьютеров также получила ряд нововведений, повысивших общую производительность. При работе с низкоскоростными механическими устройствами ввода-вывода (считыватель перфокарт, принтер) стала применяться подкачка-откачка данных, или spooling (Simultaneous Peripheral Operation On Line). Пакет заданий с перфокарт предварительно записывался на магнитную ленту, а затем на диск. Когда в процессе выполнения задания требовался ввод данных, они читались с диска. Выходные данные записывались на ленту или диск, и распечатывались после завершения задания. Вначале действительные операции ввода-вывода осуществлялись с использованием более простых, отдельно стоящих компьютеров. В дальнейшем они стали выполняться на том же самом компьютере. Введение техники подкачки-откачки в пакетные системы позволило совместить реальные операции ввода-вывода одного задания с выполнением другого задания, но потребовало разработки аппарата прерываний для извещения процессора об окончании этих операций.

Применение магнитного диска, являющегося, в отличие от магнитной ленты, устройством произвольного доступа, сделало возможным выбирать очередное задания для выполнения. Пакетные системы начинают заниматься планированием заданий, выбирая то или иное в зависимости от наличия запрошенных ресурсов, срочности вычислений и т.д.

Дальнейшее повышение эффективности использования процессора было достигнуто с помощью мультипрограммирования – одновременного выполнения нескольких программ, относящихся к разным задачам. При возникновении задержки при выполнении одной из программ, например из-за поиска на магнитной ленте участка, где хранятся исходные данные, её выполнение прерывается и осуществляется переход к диспетчер-программе, которая передаёт управление следующей программе. Когда операция поиска (или ввода-вывода) заканчивается, процессор возвращается к выполнению первой программы. При этом каждая программа загружается в свой участок оперативной памяти, называемый разделом, и не влиять на выполнение другой программы.

Появление мультипрограммирования требует настоящей революции в строении вычислительной системы. Особую роль здесь играет программная и аппаратная поддержка (многие аппаратные

новшества появились еще на предыдущем этапе эволюции).

К программным средствам поддержки мультипрограммирования относятся диспетчер-программы и проблемно-ориентированные языки программирования. К аппаратным средствам относятся защита памяти и организация прерываний. Для реализации защиты памяти программы не должны иметь самостоятельного доступа к распределению ресурсов и должны быть изолированы друг от друга. Команды разделяются на привилегированные и непривилегированные, первые могут исполняться только ядром операционной системы (супервайзером), появляется привилегированный режим работы ОС, супервайзер обеспечивает контролируемую смену режимов. Прерывания оповещают ОС о том, что произошло асинхронное событие, например завершилась операция ввода-вывода, или о том, что выполнение программы привело к ситуации, требующей вмешательства ОС, например деление на ноль или попытка нарушения защиты.

Мультипрограммные системы обеспечили возможность более эффективного использования системных ресурсов (например, процессора, памяти, периферийных устройств), но они еще долго оставались пакетными. Пользователь не мог непосредственно взаимодействовать с заданием и должен был предусмотреть с помощью управляющих карт все возможные ситуации. Отладка программ по-прежнему занимала много времени и требовала изучения многостраничных распечаток содержимого памяти и регистров или использования отладочной печати.

Логическим расширением систем мультипрограммирования стали системы разделения времени. В них процессор переключается между задачами не только на время операций ввода-вывода, но и просто по прошествии определенного времени. Эти переключения происходят так часто, что пользователи могут взаимодействовать со своими программами во время их выполнения, то есть интерактивно. В результате появляется возможность одновременной работы нескольких пользователей на одной компьютерной системе. Внедрение механизма виртуальной памяти, при которой в оперативную память загружался только фрагмент программы, который необходимо в данный момент выполнять, а остальная часть программы остается на диске, позволило создавать иллюзию неограниченности оперативной памяти. Это ослабило ограничения на количество одновременно работающих пользователей. В системах разделения времени, применяя электронно-лучевой монитор и клавиатуру, стало возможным производить отладку программы в интерактивном режиме.

Развивался параллелизм, прямой доступ к памяти и организация каналов ввода-вывода позволили освободить центральный процессор от рутинных операций.

Параллельно внутренней эволюции вычислительных систем происходила и внешняя их эволюция. До начала этого периода вычислительные комплексы были, как правило, несовместимы. В начале третьего периода появились семейства программно совместимых машин, работающих под управлением одной и той же операционной системы. Первым семейством программно совместимых компьютеров, построенных на интегральных микросхемах, стала серия машин IBM/360, для которых была выпущена операционная система OS/360. За ним последовала линия компьютеров PDP, несовместимых с линией IBM. Операционные системы, создаваемые для таких семейств компьютеров, выходили сложными, громоздкими, содержали большое количество ошибок и требовали новых методов организации всего процесса программирования.

Четвертый период (с 1980 г. по настоящее время). Классические, сетевые и распределенные системы

Резкое возрастание степени интеграции привело к переходу на большие интегральные схемы. Стоимость микросхем существенно понизилась, и компьютер, не отличающийся по архитектуре от компьютеров серии PDP, по цене и простоте эксплуатации стал доступен отдельному человеку, появились персональные компьютеры. Первоначально персональные компьютеры предназначались для использования в однопрограммном режиме, что повлекло за собой деградацию архитектуры этих ЭВМ и их операционных систем.

Однако рост сложности и разнообразия задач, решаемых на персональных компьютерах, необходимость повышения надежности их работы привели к возрождению практически всех черт, характерных для архитектуры больших вычислительных систем. А то, что компьютеры стали использоваться не только специалистами, потребовало разработки «дружественного» программного обеспечения.

В середине 80-х стали бурно развиваться сети компьютеров, работающих под управлением сетевых или распределенных операционных систем. В сетевых операционных системах пользователи могут получить доступ к ресурсам другого сетевого компьютера, при этом каждая машина в сети работает под управлением своей локальной операционной системы. Распределенная система внешне выглядит как обычная автономная система, но пользователь не знает и не должен знать, где его файлы хранятся – на локальной или удаленной машине – и где его программы выполняются.

§8.2. Самые первые операционные системы

В период первого поколения компьютеров программисты разрабатывали свои программы для «голой» аппаратуры, используя

машинные коды, понятные только для этой аппаратуры. В отсутствие операционной системы использовать всю большую и дорогую вычислительную машину в каждый конкретный момент времени могло только одно приложение (и один пользователь). Первые операционные системы были разработаны в 1950-е годы, они позволили запускать автоматически процессы от разных пользователей, которые раньше приходили в порядке живой очереди.

Первая система была создана в Массачусетском технологическом институте в 1954 г. для автоматизации считывания с перфоленты при расчете носа сверхзвукового бомбардировщика на UNIVAC 1103, установленном на воздушной базе Eglin Air Force Base (Флорида) [8.2].

В компании General Motors в 1955 г. создана операционная система I/O system, автоматизирующая выполнение задач на компьютере IBM 701 [8.3]. Годом позже той же лабораторией GM была создана более продвинутая система GM-NAA I/O, предназначенная для авиапроизводителя North American Aviation (NAA) и работавшая на компьютере IBM 704.

§8.3. UNIX (1969)

К концу 1960-х годов отраслью и научно-образовательным сообществом был создан целый ряд ОС, в том числе такие развитые системы, как OS/360 (IBM) и SCOPE (CDC). Эклектичный характер разработки систем привел к сложности и громоздкости, они были плохо масштабируемыми и полностью несовместимы между собой. Их разработка и отладка затягивались, документирование часто не велось, и создать новую систему часто было проще, чем пытаться отладить старую.

В 1960-е годы Массачусетский Технологический институт (MIT), Bell Labs и General Electric разрабатывали операционную систему Multics (Multiplexed Information and Computing Service) для компьютеров третьего поколения (в частности, GE-645). Успешной коммерческой системой Multics не стала, но благодаря новаторским идеям оказала сильное влияние на компьютерную технику. Вместе с этой ОС поставлялся язык C. При этом C был разработан и написан так, чтобы обеспечить переносимость разработки операционной системы, сама система обладала модульностью, обеспечивала механизмы разграничения пользователей, безопасности, имела развитую файловую систему и пр.

В 1969 году несколько разработчиков Multics – Кен Томпсон, Деннис Ритчи и Брайан Керниган, задумали разработать собственную ОС, которая отличалась бы максимальной простотой и минимальными размерами. Разработанная система UNICS (от «Uniplexed», т.е. односложная, в противоположность «Multiplexed», т.е. комплексной),

или UNIX, была выпущена AT&T, оказала огромное влияние на развитие компьютерных операционных систем и считается одной из самых исторически важных ОС. UNIX обладала файловой иерархической системой, предоставляла возможность использования сетевыми возможностями, поставлялась с компилятором C, обладала интерпретатором команд, поддерживала написание скриптов и пр.

С 1974 г. система стала бесплатно распространяться среди университетов и академических учреждений, что способствовало популяризации Unix и языка C, а также переносу на разные аппаратные платформы. В результате ОС Unix стала синонимом «Открытой системы». Одновременно с этим, помимо AT&T, свои версии стал выпускать университет Беркли (BSD UNIX) и другие компании и организации (IBM, Sun, DEC).

История Unix тесно связана с развитием сетей. Так в частности, в 1975 г. Unix был выбран как для мини-хостов ARPANET, став сетевой системой, а версия Unix BSD 4.2, вышедшая в 1983 г., поддерживала протокол TCP/IP и обеспечила ему широкое распространение.

В 1987 г. Эндрю Танненбаум с учебными целями создал микроядерную версию UNIX под названием MINIX (minimal UNIX), которая могла работать на небольших персональных компьютерах. Эта операционная система вдохновила Линуса Торвальдса на разработку системы Linux в начале 1990-х.

GNU (1983)

GNU, от англ. «GNU's Not UNIX», или «GNU – не Unix!». Проект GNU организован в 1984 году для разработки завершённой UNIX-подобной операционной системы, которая является свободным программным обеспечением: операционной системы GNU. В рамках проекта создано большое количество системных утилит и прикладных программ.

Свободное программное обеспечение, по классификации GNU, имеет 4 степени свободы:

0 – Свободно запускать программы для любых целей

1 – Свободно изучать, как работает программа, и иметь возможность свободно адаптировать ее под свои нужды

2 – Свободно распространять копии

3 – Свободно улучшать программы и делать доступными улучшения для общества.

POSIX (1988) [8.4]

Различные версии Unix выпускались многими компаниями и организациями, и различия между версиями постепенно становились все более значительными. Во-первых, в основе существовавших вариантов

Unix лежали разные базовые системы – например, 4BSD или разные релизы первой коммерческой версии Unix System V (SVR3, SVR4). Во-вторых, почти каждый разработчик либо адаптировал систему под возможности собственной аппаратной платформы, либо просто вносил некоторые усовершенствования. Такие усовершенствования реализовывались различным образом и обычно не согласовывались между собой, поскольку почти все существовавшие системы были коммерческими и закрытыми.

Это поставило под угрозу портируемость приложений, тем более что многие программы по-прежнему писались под абстрактный Unix. Итогом стало принятие стандарта POSIX (Portable Operation System Interface based on uniX) – набора требований, призванных обеспечить как переносимость самих систем на различные архитектуры, так и работу приложений в различных ОС, для чего окружение приложений стандартизировалось на уровне системных сервисов. POSIX, или POSIX-совместимые операционные системы, стал синонимом термина «Unix-подобная операционная система», или Unix'ов.

§8.4. Linux (1990) [8.5]

Система Unix, несмотря на распространенность в университетах и то, что её изучение входило в ряд дисциплин, оставалась малодоступной для образовательных целей в силу ряда причин – из-за ограничений на распространение, из-за привязки большинства вариантов к дорогим аппаратным платформам, и из-за своей сложности.

Это и стало причиной, по которой профессор Амстердамского университета Энди Танненбаум, исключительно для образовательных целей, в 1987 г. написал минималистский вариант Unix – Minix, и стал распространять ее на дискетах вместе со своим учебником по теории операционных систем. Учебная Minix была неспособна к выполнению реальных пользовательских задач из-за «урезанности», в ней сознательно не были реализованы даже некоторые особенности Unix – они полагались слишком сложными для студентов. Довольно быстро сложилось сообщество пользователей Minix, дополняющих систему доработками, превращавшими ее в среду для реального использования. Однако распространять такую модернизированную систему по условиям лицензии было нельзя, можно было устанавливать дополнения только к системе, купленной с книгой.

Именно так в 1990 г. поступил студент университета Хельсинки Линус Торвальдс, изучавший Unix – купил книгу Танненбаума вместе с Minix, скачал и установил соответствующие дополнения. Написанная Торвальдсом программа терминального доступа к университетскому компьютеру под управлением Minix работала плохо, и Линус пересобрал ядро, которое было размещено в новую файловую систему ext (то есть

Extended – расширение для файловой системы Minix). Затем Линус портировал в свое ядро командную оболочку bash, разработанную в проекте GNU и присоединил единственный свободный из наличных компиляторов языка C – GNU C Compiler (или gcc). В августе 1991 г. он объявил о создании новой операционной системы – Linux. Система была выложена в свободный доступ и стала распространяться по лицензии GNU GPL (Универсальная общедоступная лицензия), под которой распространялись такие важные ее компоненты, как bash и gcc.

Отметим ряд существенных моментов. Во-первых, Линус не занимался адаптацией Minix, и Linux не является клоном Unix. В работе Линус руководствовался описаниями системных вызовов, данными в стандарте POSIX, не привязанными к какой-либо конкретной реализации, и Linux можно считать первой настоящей POSIX-системой.

Во-вторых, Linux создавался на машине с процессором i386 для архитектуры Intel и, первоначально, – только для нее. Поэтому соответствие стандартам требовалось не для переносимости Linux самого по себе, а для обеспечения возможности компиляции в этой ОС всего ранее созданного программного ассортимента для Unix и POSIX-совместимых систем вообще. С другой стороны, следование стандарту сделало возможным перенос Linux-софта на другие Unix-подобные платформы.

В третьих, Линус создал метод разработки масштабных проектов Open Sources. Ранее свободное программное обеспечение создавалось либо в рамках университетских проектов при правительственном финансировании, либо энтузиастами-одиночками (или их небольшими группами). При прекращении финансирования, утрате интереса со стороны разработчиков и пр. в любой момент проект мог быть заморожен или прекращен. Выложив свою систему в открытый доступ и предложив всем желающим принять участие в ее дальнейшем совершенствовании, Линус обеспечил проекту непрерывность развития. Призыв был встречен с пониманием, тем более что соответствие системы стандартам гарантировало, что усилия программистов не пропадут даром и смогут быть использованы в любой POSIX-совместимой системе.

В результате деятельности такого коллектива разработчиков ядро Linux очень быстро обросло такими функциями, как поддержка сетей, протокола TCP/IP, оконной системы X, на нее были портированы все аналоги классических Unix-утилит и приложений, созданные как в рамках проекта GNU, так и независимыми разработчиками. А затем настал черед и чисто пользовательских приложений, следствием чего было привлечение не только новых разработчиков, но и конечных пользователей.

§8.5. Дисковые операционные системы (1960–1990-е)

Дисковая операционная система обеспечивает абстрагирование и управление такими устройствами хранения информации, как жесткие диски или дискеты. Размеры оперативной памяти в микрокомпьютерах были сильно ограничены, и дисковая операционная система обеспечивала загрузку компонентов только при необходимости. При отсутствии дисковой операционной системы доступ к диску ограничивался низкоуровневыми операциями на уровне секторов. Дисковая операционная система могла выступать или как компонент операционной системы, как её расширение, или как самостоятельная, если обеспечивала загрузку с диска, абстракцию и рациональное использование дисковых устройств.

Одной из первых таких операционных систем является DOS/360, разработанная для компьютеров System/360 в 1964 г. из-за задержек в создании OS/360. Было выпущено несколько версий дисковой системы для машин с разным объемом памяти и разными устройствами хранения. DOS/360 работала с 16 кбайт памяти, в то время как для OS/360, появившейся 2 года спустя, требовалось 64 кбайт.

Для микрокомпьютеров вроде Atari (1972) или Apple II (1977) дисковая операционная система являлась расширением операционной системы, поскольку не все первые микрокомпьютеры оснащались дисками из-за их высокой стоимости.

CP/M (1975)

Операционная система CP/M (Control Programs for Microcomputers) создана Килдэлом в 1975 г. для микрокомпьютеров, сконструированных на микропроцессорах Intel 8080. С конца 70-х и до середины 80-х CP/M была одной из наиболее популярных операционных систем и принята за своеобразный стандарт, поскольку использовалась многими производителями компьютеров в разных странах. Под нее созданы десятки тысяч прикладных программ. В 1981 г. в СССР аналог CP/M был разработан под названием МикроДОС. С выбором системы MS-DOS в качестве операционной системы для персональных компьютеров IBM популярность CP/M сошла на нет, а с 2001 г. CP/M распространяется как open-source.

Система CP/M предназначена для использования совместно с прикладным программным обеспечением, включая компиляторы языков высокого уровня (Фортран, Паскаль, Си и др.), средства организации баз данных, экранные редакторы, игры и многое другое. На этой системе впервые вышли: один из первых текстовых процессоров WordStar, программа для баз данных DBASE, Turbo Pascal, Multiplan (родоначальник Microsoft Excel), AutoCAD и пр.

CP/M состоит из двух частей: постоянной – базовой дисковой

операционной системы (BDOS), и переменной – базовой системы ввода/вывода (BIOS). Постоянная часть BDOS может использоваться в разных компьютерах без изменений. Здесь имеется специальный программный модуль, который принимает и интерпретирует команды, вводимые с клавиатуры. Кроме того, BDOS организует управление ресурсами системы, прежде всего файлами, и обменом информацией между различными периферийными устройствами. В BDOS входит ряд резидентных процедур управления работой дисковой системы (вывод на экран оглавления диска, удаления дискового файла и др.). Переменная часть – BIOS обеспечивает выполнение простейших операций передачи информации от микропроцессора к устройствам ввода/вывода с помощью драйверов нулевого уровня.

Система была выпущена также в многопользовательском варианте MP/M (1979), поддерживающем многозадачность, а позже портирована на процессоры 8086 под именем CP/M-86 (1982).

MS-DOS (1981) [8.6]

Самой известной дисковой операционной системой является MS-DOS (Micro-Soft DOS), выпущенная в 1981 г. Система являлась переработанной ОС 86-DOS, в свою очередь базирующейся на CP/M. Не являясь принципиально новой, система оказала влияние на рынок ОС благодаря тому, что устанавливалась на все персональные компьютеры IBM и на большинство IBM-совместимых компьютеров. Также MS-DOS являлась ядром для систем Windows 95, 98 и ME.

Поддержка MS-DOS прекращена производителем в 2000 г., но сторонними производителями продолжается выпуск полностью совместимых с MS-DOS дисковых операционных систем, таких как DR-DOS (Digital Research, 1988), PTS-DOS (Физтех-софт, 1993) и FreeDOS (GNU, 2006). Интерес пользователей к системе вызван несколькими причинами, например тем, что DOS предоставляет возможность прямого обращения к устройствам и удобен для программирования на ассемблере, а требования к аппаратуре для запуска DOS минимальны и система работает почти на любом компьютере. Для DOS существует большое количество программ, и работали с ним огромное число людей, поэтому клоны DOS остаются востребованными до сих пор.

OS/2 (1987)

Операционная система OS/2 разрабатывалась совместно IBM и Microsoft. Предполагалось, что разрабатываемая оконная система придет на смену DOS. Выпущенная в 1987 г. OS/2 стала первой рабочей 32-битной системой, она обладала многозадачностью, многонитевостью (выполнение одного приложения на нескольких процессорах), и способностью выполнять DOS-программы с помощью виртуальных

машин процессоров 8086. Система оказала влияние на стратегию Microsoft в области операционных систем, в частности она очень схожа с Windows 95.

С начала 90-х компании IBM и Microsoft разрабатывают собственные операционные системы. IBM для собственных майнфреймов выпускает POSIX-совместимые системы z/OS и z/VM, «родословную» которых можно провести к OS/360, а для серверов и рабочих станций серии RS/6000 – основанную на Unix систему AIX.

Современные операционные системы развиваются в направлении поддержки новых сетевых технологий, многопоточности и многоядерности. Развиваются проекты с открытым исходным кодом. Развивается виртуализация, обеспечивающая возможность выполнения приложения в среде другой ОС. В конце 2000-х гг. появляется поддержка облачных вычислений, большого успеха добиваются и операционные системы для мобильных платформ.

ГЛАВА 9. ЯЗЫКИ ПРОГРАММИРОВАНИЯ [9.1]

Языки программирования также можно разделить по этапам развития, как и вычислительную технику, и даже удастся выделить несколько первых поколений, но после них классификация теряет смысл, т.к. образующееся разнообразие языков не укладывается во временные рамки. Поэтому четко выделим только три первых поколения языков программирования:

1. Машинные языки
2. Ассемблеры
3. Языки структурного программирования

§9.1. «Доисторические» языки

К доисторическим языкам программирования относят нотацию Ады Лавлэйс (1843), рассматривавшей примеры программирования в комментариях к статье «Очерк Аналитической машины», и язык Plankalkül (нем. «Формальная система планирования»), разработанный Конрадом Цузе в Германии между 1942 и 1945 годами.

Создание языка программирования Plankalkül было естественным продолжением работ по конструированию компьютеров. Работы над языком были закончены около 1946 года, однако развития язык не получил. Цузе был занят попытками коммерциализации машины Z3, его статья о Plankalkül в 1948 г. никого не привлекла, т.к. программирование мыслилось исключительно в машинных кодах. Полное руководство по языку увидело свет только в 1972 году, а компилятор – в 1998. Из-за этого язык долгое время был мало кому известен и влияния на дальнейшее развитие индустрии не оказал [9.2].

Тем не менее Plankalkül является первым в мире высокоуровневым языком программирования. Основные концепции языка включают: наличие подпрограмм, наличие операции присваивания, циклы, условный оператор, возможность манипуляций с массивами и возможность манипуляций со списками. Для демонстрации возможностей языка Цузе попытался написать программу игры в шахматы (объемом около 60 страниц). Одной из проблем языка был сложный синтаксис.

Пример вычислений на Plankalkül строки «A[5] = A[4]+1»:

	A + 1 =>	A	
V	4	5	– V – строка для индексов
S	1.n	1.n	– S – строка для задания типов данных, 1.n – целое размером n бит.

§9.2. Языки программирования низкого уровня [9.3]

Машинные коды

Первоначально программирование велось в цифровых кодах, соответствующих электрическим схемам компьютеров (десятичных или двоичных). Для упрощения записи двоичная система записывалась в виде восьмеричной или шестнадцатеричной, но вводить коды нужно было все равно в системе исчисления с машинным основанием. Контроль ввода и конечного результата, а также поиск ошибок, осуществлялся по контрольным лампочкам на панели управления, либо по распечаткам.

Пример команд в машинных кодах для x86:

B8 01 00 – поместить в регистр AX число 1.
BB 02 00 – поместить в регистр BX число 2.
01 D8 – сложить AX с BX, результат в AX.

Эти же команды в двоичном виде:
101110000000000100000000
101110110000001000000000
0000000111011000

Ассемблеры

Ассемблер – это мнемоническая запись кодов центрального процессора. Программу, написанную на ассемблере, переводит специальная программа-транслятор, заменяя каждую ассемблерную команду на соответствующий двоичный код. Набор полученных кодов полностью соответствует исходному набору ассемблерных команд, он может быть непосредственно загружен в память машины и выполнен процессором. Однако, поскольку каждый процессор использует только свой набор кодов, такая программа не может быть непосредственно выполнена на машине с другим типом процессора.

Впервые мнемоника была введена разработчиками EDSAC (1949), обозначившими все 18 машинных команд заглавными буквами. Так, S обозначала «вычитание», I – «прочитать следующий ряд дырочек на входной бумажной ленте», T – «передать информацию в память», а Z – «остановка машины». Существенный вклад в популяризацию ассемблеров внесла компания IBM, разработавшая для System/360 язык IBM Basic assembly. Ассемблер широко использовался для написания программ до начала 1980-х, когда признание получил язык C.

Программы на ассемблере более понятны в тех объемах, в которых они существовали в виде машинных кодов, т.к. в них сразу видны используемые регистры. Но при увеличении количества команд прозрачность программ снова теряется. В самом деле, листинг из ста

команд ассемблера понятнее, чем листинг из ста многобайтовых двоичных чисел, но листинг из тысячи команд ассемблера столь же непонятен. Проблема отладки или создания больших программ поэтому по-прежнему остается.

Пример команд на ассемблере:

mov ax,0001 – занести в регистр AX значение 1.
mov bx,0002 – занести в регистр BX значение 2.
add ax,bx – сложить AX и BX, результат – в AX.

§9.3. Языки программирования высокого уровня

•Fortran (1956), научные и инженерные расчеты.
•COBOL (1959), для управления и бизнеса.
•ALGOL (1958/1960), для записи алгоритмов.
•PL/1 (1962), разработан для System/360.
•BASIC (1964), для записи простых программ в учебных целей.

Fortran (1956)

В 1953 г. Джон Бекус, ранее руководивший созданием высокоуровневого языка Speedcode для IBM 701 (язык состоял из псевдоинструкций для математических функций, но полученная программа работала в 10–20 медленнее написанной на ассемблере), предложил создать для компьютера IBM 704 язык, позволяющий записывать команды почти в обычной алгебраической форме. IBM стремилась сделать компьютеры более «дружественными», чтобы расширить рынки сбыта, и в 1956 был создан новый язык – Fortran (FORmula TRANslator). Также, как язык Speedcode и компьютеры IBM 701 и 704, этот язык предназначался для использования при математических расчетах в научных и инженерных задачах. К 1960 г. компиляторы стали доступны и для других моделей IBM. Компиляторы генерировали быстрый код, сопоставимый с ассемблером, а сам язык оказался очень популярен среди ученых. Но максимальную популярность языку обеспечили конкурирующие производители, начавшие выпускать трансляторы для своих компьютеров. Так Fortran стал первым языком, поддерживаемым множеством платформ, и первым языком, получившим широкое признание. Несмотря на удобство записи формул, язык был сложным для понимания написанной программы. Текст изобиловал метками и переходами GOTO, а один знак мог поменять весь смысл кода, например [9.4]:

DO 150 I=1,10 – цикл до строки 150, 10 раз подряд.
DO 150 I=1.10 – присвоение переменной DO150I значения 1.10.

В 1957 г., уже для бизнес-приложений, в IBM был создан эквивалент Фортрана – COMTRAN (COMmercial TRANslator).

Cobol (1960)

В конце 1959 года в Пенсильванском университете, на заседании производителей и пользователей компьютеров, была предложена концепция аппаратно-независимого языка программирования, разработанная ученым и офицером флота США Грейс Хоппер. Хоппер стремилась сделать язык программирования похожим на обычный человеческий, и ранее разработала язык Flow-Matic. Поддержка Министерства обороны, участие таких компаний, как IBM, RCA (Radio Corporation of America) и Honeywell, а также спрос на доступный бизнесу язык программирования способствовали быстрому написанию спецификации и компиляторов языка, названного COBOL (Common Business-Oriented Language, универсальный язык, ориентированный на задачи бизнеса). Спецификация языка во многом опиралась на три ранее разработанных – на язык Flow-Matic, на идеи бизнес-языка Comtran, и на элементы, позаимствованные из языка FACT компании Honeywell. Читабельность, самодокументируемость и простота языка, наряду с поддержкой разработчиками и аппаратной независимостью, сделали его очень популярным для применения в коммерческих компаниях и правительственных организациях в 60–70-х годах. В Коболе, в отличие от большинства других языков, все данные описываются в отдельной секции, которая не совпадает с секцией команд. Это позволяет использовать одни и те же описания данных в различных программах. Особенно эффективно в Коболе производятся простые арифметические операции с большими массивами данных, что довольно часто приходится делать в бухгалтерских расчетах. Кроме того, само существование языка стимулировало развитие многих других языков программирования высокого уровня, которые используют квази-английский синтаксис (от Бэйсика до PHP) и дало возможность изучать программирование более широким массам, чем раньше. В СССР этот язык тоже достаточно широко использовался, причем он, один из немногих, был переведен на русский язык.

Algol (1958)

В 1958 году в Цюрихе ряд ведущих программистов представили язык ALGOL (ALGOrithmic Language, алгоритмический язык программирования), позиционируя его как универсальный язык для широкого круга применений. Планировалось, что он избежит ряда проблем языка Fortran, но выявленные новые проблемы привели к тому, что «классический» ALGOL вышел только в 1960 г. Алгол применялся в основном в компьютерных науках в Европе и, в меньшей степени, в США. Его использование в коммерческих целях было затруднено из-за отсутствия средств стандартного ввода/вывода и отсутствия интереса к языку со стороны производителей компьютеров. Тем не менее, Алгол

почти на 30 лет стал стандартом описания алгоритмов в учебниках и научных работах, а многие языки, разработанные впоследствии, содержат многие идеи и решения, взятые из Алгола. В нем была доведена до логического завершения сама концепция операторных алгоритмических языков с заранее фиксированными типами данных и блочной структурой, появилась возможность разработки отдельных модулей проекта независимо друг от друга, а также реализации вызовов фрагментов кода программы, способов передачи параметров между процедурами и функциями, появилась рекурсия. В нем появилось представление о программе как о блочной структуре, а не списке команд. Основной блок и вложенные фрагменты программы ограничивались командами begin и end, что позволило отказаться от переходов GOTO.

PL/1 (1966)

В создании языков свое слово попыталась сказать и компания IBM, в 1961 г. начавшая разрабатывать System/360. На тот момент бизнес-пользователи переводились с языка Comtran на COBOL, а научные программы писались на Фортране, в то время как идея System/360 заключалась в создании для всех пользователей одной универсальной системы, в состав которой входил бы и единый язык программирования. Попытка расширить язык Fortran, из-за присущих ему недостатков, в 1963 г. была отвергнута, после чего компания, несмотря на просьбы пользователей, отказалась реализовывать для своих компьютеров язык Algol и приступила к разработке нового языка PL/1 (Programming Language One), в котором очень заметным было влияние Алгола. Первый компилятор PL/1 появился в 1966 г. Язык обеспечивал обработку данных и научные численные вычисления, рекурсии, связанные структуры данных, символьные и битовые строки. Синтаксис языка был похож на английский язык и подходил для описания сложных форматов данных, с широким набором функций, доступных для проверки и манипулировать ими. Со временем выяснилось, что вследствие попытки удовлетворить нужды всех пользователей язык получился перегруженным возможностями и концепциями. Это привело к сложностям при разработке компиляторов и тому, что генерируемый код оказывался далеким от оптимального, проигрывая в математических задачах Фортрану. Тем не менее, благодаря большим возможностям, чем COBOL, язык PL/1 широко применялся для бухгалтерских программ, а самой компанией IBM он применялся и в майнфреймах 1960-х, и в персональных компьютерах 80-х и 90-х, и продолжает применяться до сих пор. Также PL/1 известен как язык высокого уровня, на котором было впервые написано ядро операционной системы (Multics) [9.5].

Basic (1964)

В 1963 году профессора Дартмутского колледжа (университет в США) Томас Курц и Джон Джордж Кемени, работавший ранее с фон Нейманом, впервые предложили применять компьютеры в систематическом учебном процессе, обучая основам программирования студентов всех специализаций, даже гуманитарных [9.4]. Через год ими, с помощью группы студентов, был разработан язык BASIC (Beginner's All-purpose Symbolic Instruction Code, универсальный символический код для начинающих), позволявший писать программы, не имея специальной подготовки. Благодаря подключению к майнфрейму GE-265 до 20 терминалов, работающих в режиме разделения времени (эта идея была предложена в 1959 г. в MIT), стала возможной одновременная работа множества студентов. По синтаксису язык напоминал Фортран, но большинство команд состояло из простых английских слов – SAVE, PRINT, INPUT. Программы набирались на терминале, а не вводились с перфокарт, а язык предусматривал прерывание выполняемой программы и внесение в неё новых данных или операторов. Разработанный язык оказался очень универсальным, в сравнении с другими – небольшим, а сам компилятор разработчики сделали бесплатным, что сделало BASIC очень популярным. Но массовым он стал после 1975 г., когда Б. Гейтс и П. Аллен, только что учредившие компанию Micro-Soft (будущую Microsoft), чуть ли не от безделья написали интерпретатор языка для компьютера Altair. В скором времени интерпретатор Altair BASIC, ставший первым языком программирования для первого персонального компьютера, стал поставляться на компьютеры вместе с операционной системой CP/M. Затем интерпретатор появился и на персональных компьютерах IBM, работая под MS-DOS. Разработано много разновидностей языка, в настоящее время он существует, например, в виде интерпретатора VBScript или в виде языка Visual Basic .NET. Вопреки ожиданиям разработчиков, язык «для начинающих» оказался востребован даже для разработки бизнес-приложений. Из недостатков языка отмечают широко применяющийся оператор GOTO, а так же то, что программы через интерпретатор часто работали быстрее, чем откомпилированные.

§9.4. Универсальные языки программирования

•Pascal (1970), улучшение Алгола и задачи обучения.
•C (1973), для системного программирования.
•Ada (1980), для встроенных систем реального времени.
•Lisp (1958), Prolog (1972), обработка символьной информации.

Pascal (1970)

В начале 60-х годов число языков программирования начало резко

возрастать. Попытки создать универсальный язык программирования не увенчалась успехом, т.к. все равно каждый новый язык оказывался ориентирован на решение определенных задач. Такой заслуживающий внимания язык, как Алгол, широкого одобрения не получил, но оказал значительное влияние на развитие других языков, среди которых следует особо отметить Паскаль, разработанный в 1968–69 гг. швейцарским ученым Никлаусом Виртом, принимавшим участие в разработке языка Алгол-68. История языка похожа на историю Бэйсика – Вирт нуждался в языке, с помощью которого было бы легко обучать студентов навыкам программирования. Базовая концепция Паскаля была разработана Виртом примерно в 1970 году, и Паскаль очень быстро начал повсеместно распространяться, прежде всего, благодаря легкости в изучении и наглядности написанных на нем программ. Язык Паскаль требовал от программиста определения всех переменных в отдельной секции в начале программы. Так как эти определения задавались явным образом, то в программах появлялось сравнительно немного ошибок, и их было проще понять и исправить разработчику. Это сделало Паскаль популярным при создании больших программ. Одно время он даже был объявлен официальным языком программирования для учащихся средних школ, которые намерены специализироваться в области вычислительной техники и программирования в американских университетах. Паскаль был основным языком высокого уровня в первых компьютерах Apple, многие программы были написаны на объектно-ориентированных версиях языка – Delphi или Object Pascal.

C (1973)

В 1969–1973 гг. в компании Bell Labs для создаваемой ОС Unix Деннисом Ритчи был разработан язык С, и уже в 1973 г. на нем было переписано ядро Unix. Язык был призван обеспечить компиляцию с помощью простого компилятора, низкоуровневый доступ к памяти, максимальную близость к машинным командам и минимизировать время на отладку. Хотя язык С (как и Unix) создавался программистами для программистов и является самым популярным языком для системного программирования, область его применения значительно шире, т.к. оказались востребованы такие его свойства, как переносимость кода и эффективность, возможность доступа к прямым адресам оборудования, а также низкие требования к системным ресурсам. Так же язык часто используется для обучения программированию.

В 1970–80-е гг. разные версии языка С были реализованы почти на всех компьютерах и микрокомпьютерах, на нем даже стали писать компиляторы, библиотеки и интерпретаторы с других языков программирования.

В 1980-е гг. в компании Bell Labs язык был усовершенствован

Б. Страуструпом и обрел поддержку объектно-ориентированного программирования. Такой язык стал называться C++.

Ada (1980)

В 1970-х гг. министерство обороны США озаботилось большим числом языков программирования, используемых во встраиваемых компьютерах, особенно тем, что многие из них оказались устаревшими или аппаратно-зависимыми. Результатом стало создание спецификации языка для встраиваемых систем и объявление в 1977 г. конкурса на создание нового, поскольку языка, полностью удовлетворяющего требованиям, найдено не было. Так как все языки, дошедшие до последних туров конкурса, были основаны на Паскале, то новый язык, названный Ада (в честь Ады Лавлейс), можно характеризовать как Паскаль, расширенный добавлением новых элементов. В результате получился существенно более сложный язык. Причем было запрещено выпускать к нему компиляторы, не прошедшие официальное тестирование, поэтому его применение далеко за пределы военных встроенных систем не вышло. Особенностями Ада являются строгая типизация, модульность, проверка на наличие ошибок на этапе компиляции и в реальном режиме, поддержка параллельной обработки данных и обработка исключений. По этим причинам Ада широко используется в системах, где сбой может привести к серьезным последствиям – в авионике (управление Boeing 777), в системах вооружения (в т.ч. ядерное оружие), космических кораблях и пр. [9.6].

Lisp (1958) и Prolog (1972)

Следует упомянуть о таких языках обработки символьной информации, как Lisp (List Processing Language) и Prolog (Programming in Logic). Язык Lisp разработал в 1958 г. Дж. Маккарти (США). В нем и программы, и данные представляются в виде связанного списка символов, в итоге программа может управлять исходным кодом, как списком данных. Он стал основой ряда программных реализаций интеллектуальных систем и дал толчок к разработке множества специализированных языков искусственного интеллекта и языков представлений знаний. Наряду с Ada язык Lisp был стандартизован для военного и промышленного применения, это диалект Common Lisp.

В 1972 году появился язык Prolog, разработанный Аланом Колмари (Франция). Это единственный язык программирования, основанный на логическом выводе и решении поставленной задачи, что роднит его с искусственным интеллектом. В отличие от большинства других языков, перед ним достаточно поставить некоторую задачу, а затем он сам будет искать решение. Из-за необычности своей структуры он использовался при решении достаточно нестандартных задач, поэтому оказался

распространен не так широко, как другие языки. Максимальный всплеск интереса к Прологу относится ко времени программы разработки компьютеров пятого поколения в 1980-х гг., когда разработчики надеялись, что с помощью Пролога можно будет сформулировать новые принципы, которые приведут к созданию компьютеров более высокого уровня интеллекта. Тогда же «символьные» языки попытались причислить к «пятому» поколению компьютеров.

К началу 1980-х годов были разработаны принципы объектно-ориентированного программирования, что создало возможности для появления новых языков, таких как C++ или Object Pascal. Еще одну такую возможность создал в в середине 90-х гг. быстрый рост Интернета. Среди последних, например, язык Perl, изначально разработанный в 1987 г. в Unix для создания сценариев. Он получил широкое распространение в динамических веб-сайтах. Язык Java, созданный для применения во встраиваемых системах, стали использовать для веб-программирования. Во многом такие языки не являются принципиально новыми, а представляются уточнением и развитием существующих. В целом, современные языки и парадигмы программирования в значительной степени основаны на семействе языков программирования C.

Как о роли языков семейства C, так и о переплетении судеб других языков, каждый из которых создавался с собственной определенной целью, можно судить на примере алгоритмических языков. Собственно для этого создавался Algol, но взглянем на всемирные олимпиады по программированию, на которых решаются именно алгоритмические задачи. На Международной студенческой олимпиаде по программированию (**ACM/ICPC**), где сборные команды студентов ИТМО за период 1999-2009 гг. выиграли 8 золотых медалей и одно серебро, трижды становясь абсолютными победителями, в качестве официального языка используются C, C++ и Java. На Международной олимпиаде по информатике (IOI) среди школьников, где также решаются алгоритмические задачи, основными языками являются C, C++ или Pascal.

§9.5. «Эзотерические» языки программирования [9.7]

Существует большая группа языков, предназначенных для проверки границ применимости языков программирования, для демонстрации определенной концепции, или просто для шутки. Эти языки названы «эзотерическими» для того, чтобы отличаться от «популярных». При обычном программировании «эзотерические» языки не используются.

INTERCAL [9.8]

Первым из таких языков стал INTERCAL, созданный двумя студентами Принстона в 1972 г. Предназначение языка – полностью отличаться от существовавших на тот момент языков программирования, таких как FORTRAN, COBOL и ассемблер, при этом пародируя их. В руководстве языка справедливо отмечается, что язык иллюстрирует тот факт, что зачастую человек, чья работа никому непонятна, пользуется большим уважением начальства и окружающих. Большинство стандартных операций в нем оказываются нетривиальными, язык насыщен парадоксальными конструкциями, такими как COME FROM, FORGET или PLEASE ABSTAIN FROM CALCULATING (пожалуйста, воздержись от вычислений). По аналогии с разделом «Приложение» (Appendix, т.е. аппендикс) в обычных языках, INTERCAL содержит раздел «Миндалины». В языке используются разные модификаторы, например такие как «ПОЖАЛУЙСТА». Этот модификатор предусматривает два основания для получения ошибки компилятора: если он используется в программе недостаточно часто, то программа считается невежливой, а если слишком часто, то программа отвергается как слишком вежливая. В то же время фрагменты кода, которые действительно должны вызывать ошибку компиляции, тот же компилятор попросту пропускает, будто «закомментированные».

Тем не менее, язык обладает полнотой по Тьюрингу, и может решить любую задачу, которую машина Тьюринга способна решить, предъявляя минимальные требования к объему памяти. Но делать это будет очень медленно. Задача «решето Эратосфена», написанная на INTERCAL, в 1992 г. на компьютере SPARCStation-1 выполнилась за 19 часов против результата 0,5 секунды, полученного на C.

Brainfuck

Язык brainfuck, созданный в 1993 г. немцем Урбаном Мюллером, одной из целей разработки имел создание самого маленького компилятора. В отличие от INTERCAL'a язык создавался скорее для проверки концепции минимализма, чем для насмешки. Некоторые из компиляторов к языку занимают меньше 200 байт, а сам язык содержит всего 8 команд (> < + – . , [ ]). Программа на brainfuck представляет собой последовательность этих команд (все другие символы игнорируются и воспринимаются как комментарии). Указатель команд начинает с первой команды, и после её выполнения обычно переходит к следующей команде. Программа заканчивается, когда указатель команд выходит за последнюю команду. Помимо программы и указателя команд язык использует также массив не менее 30 кбайт, инициализируемых одним нулем, указатель данных и два байтовых потока для ввода и вывода. Язык также обладает Тьюринговой полнотой.

Befunge

Язык Befunge также был создан в 1993 г., но целью ставилась разработка языка, для которого невозможно создать компилятор. Программа записывалась в виде двумерной таблицы, в каждой ячейке которой размещалась команда в виде отдельного символа. По таблице пошагово перемещался указатель команд, который при достижении границы таблицы переходил на противоположный край, будто таблица свернута в тор. Сама программа имела возможность самомодификации. Тем не менее, вскоре было создано несколько компиляторов и для такого языка (это напоминает периодически появляющиеся новости об очередном взломе хакерами разных «невзламываемых» кодов).

Java2k [9.9]

Java2k – вероятностный, недетерминированный язык программирования. Разработчики заявляют, что придумали язык Java2k потому, что «обычная java – это уныло», но также добавляют, что язык заставляет программистов разрабатывать методы достижения большей вероятности получения правильного результата, т.е. попытаться написать программу, выдающую предсказуемый результат при непредсказуемом алгоритме. По утверждению разработчиков, это согласуется с физическим представлением о природе вселенной, в которой нет полной предопределенности, а всегда есть вероятность. Все встроенные функции языка имеют две реализации, и во время работы они выбираются с вероятностью 90% и 10%, для чего задействован генератор случайных чисел. При таком методе программа при каждом запуске работает по-разному. Не менее случайно работает очистка памяти – она освобождается или при выходе программы, или через произвольные моменты времени (смотря что случится раньше).

В настоящее время такой язык, разумеется, не может использоваться для создания практических приложений. Но задача, поднимаемая языком, чем-то близка к проблеме разработчиков квантовых компьютеров, правильный ответ в которых также возникает с определенной вероятностью.

Список литературы

Рекомендуемая:

1. Поппер К. Логика и рост научного знания. М., 1983.
2. Кун Т., Структура научных революций. Перевод с английского И.З. Налетова // T.S. Kuhn. The Structure of Scientific Revolutions. Chicago, 1962; М., 1975. 146 с.
3. Пригожин И. Конец Определенности. Время, хаос и новые законы природы. – Ижевск: НИЦ "Регулярная и хаотическая динамика", 2000.
4. Арнольд В. И. Теория катастроф // М.: Наука, 1990. 128 с.
5. Полунов Ю. Л. От абака до компьютера: судьбы людей и машин. М.: ИТД «Русская Редакция» , 2004. В 2 т.
6. Малиновский Б.Н. История вычислительной техники в лицах // К.: фирма «КИТ», ПТОО «А.С.К.», 1995. – 384 с. Режим доступа: http://lib.ru/MEMUARY/MALINOWSKIJ/0.txt, свободный.
7. Б.Л. Ван дер Варден. Пробуждающаяся наука. Математика древнего Египта, Вавилона и Греции. // М.: ГИФМЛ. 1959. – 462 с.
8. Sketch of The Analytical Engine Invented by Charles Babbage by L. F. Menabrea with notes upon the Memoir by the translator Ada Augusta, Countess of Lovelace // Bibliothèque Universelle de Genève, October, 1842, No. 82. Режим доступа: http://www.fourmilab.ch/babbage/sketch.html, свободный
9. Дж. фон Нейман. Теория самовоспроизводящихся автоматов // М.: Мир. 1971. 382 с.
10. John von Neumann. First draft of a report on the EDVAC. IEEE Annals of the History of Computing, Vol. 15, No. 4, 1993. P. 27–75.
11. Винер Н. Кибернетика, или управление и связь в животном и машине. М.: Наука, 1983. 344 с.
12. Винер Н. Я – математик // М.: «Наука. 1967. 356 с.
13. Paul Baran, «On distributed communications: 1. Introduction to distributed communications networks», 1964. Режим доступа: http://www.rand.org/pubs/research_memoranda/2006/RM3420.pdf, свободный
14. G.M. Amdahl, G.A. Blaauw and F.P. Brooks, Architecture of the IBM System/360 // IBM J. Res. Develop. Vol. 44. No 1/2. 2000. P. 21–36.
15. Компьютер обретает разум // Пер. с англ. Под ред. В.Л. Стефанюка. – М.: Мир, 1990. 240 с.
16. Акаев А. А., Майоров С. А. Оптические методы обработки информации (репринтное воспроизведение издания 1988 года) / Серия "Выдающиеся ученые ИТМО" – СПб: СПбГУ ИТМО, 2005. 240 с. Режим доступа: http://books.ifmo.ru/?out=book&id=258, свободный.

К главе 1:

1.1. Кохановский В.П., Лешкевич Т.Г., Матяш Т.П. и др. Философия науки в вопросах и ответах // Ростов-на-Дону, «Феникс», 2006.

1.2. Новиков А.М., Новиков Д.А. Методология. – М.: СИНТЕГ. 2007. – 668 с.

1.3. Паранаука. Философия: Энциклопедический словарь». Под ред. А.А. Ивина. — М.: Гардарики, 2004. 1072 с. Режим доступа: http://dic.academic.ru/dic.nsf/enc_philosophy/906/ПАРАНАУКА, свободный.

1.4. Наука – источник знаний и суеверий. Ю. Шрейдер // Новый мир, №10, 1969.

1.5. Кондаков Н.И. Логический словарь-справочник // М.: Наука. – 1975. с. 144.

1.6. Малинецкий Г., Потапов А. Джокеры, русла или поиски третьей парадигмы. // Знание – сила. №3. 1998. с. 19–35.

1.7. Арнольд В. И. Теория катастроф // М.: Наука, 1990. 128 с.

1.8. Sridhar Condoor. Importance of Teaching the History of Technology // Proceedings Frontiers in Education 34th Annual Conference, Vol.1 (October 21, 2004), Session T2G, pp.7-10.

К главе 2:

2.1. Полунов Ю. Л. От абака до компьютера: судьбы людей и машин. М.: ИТД «Русская Редакция» , 2004. В 2 т.

2.2. Иванов Г.И. ...И начинайте изобретать. Глава «От зарубок к биокомпьютеру» // Иркутск: Вост.-Сиб. кн. изд-во, 1987. Фрагмент: Иванов Г.И. От зарубок к биокомпьютеру. Режим доступа: http://www.trizland.ru/trizba.php?id=40, свободный.

2.3. Прудников Е. История развития информатики и вычислительной техники. Тюмень. 2006. Режим доступа: http://go-ga.by.ru/, свободный.

2.4. Б.Л. Ван дер Варден. Пробуждающаяся наука. Математика древнего Египта, Вавилона и Греции. // М.: ГИФМЛ. 1959. – 462 с. Взято из: Цифры и системы счисления, «Энциклопедия Кругосвет». Режим доступа: http://www.krugosvet.ru/articles/41/1004115/1004115a1.htm#1004115-A-101, свободный.

2.5. Еремеев В.Е. Символы и числа «Книги перемен». 2-е изд., исп. и доп. М.: Ладомир, 2005. – 600 с. Режим доступа: http://science.rsuh.ru/eremeev/tri/symbol/index.htm, свободный.

2.6. Леонардо – человек, заглянувший в будущее. Сайт журнала «Человек без границ». Режим доступа: http://www.manwb.ru/articles/persons/great_europ/Leonardo/, свободный.

переведено с Erez Kaplan «The Controversial Replica of Leonardo da Vinci's Adding Machine». Режим доступа: http://192.220.96.166/leonardo/leonardo.html, свободный.

2.7. John Walker. The Analytical Engine. Режим доступа: http://www.fourmilab.ch/babbage/contents.html, свободный.

2.8. Ada Lovelace. Sketch of the Analytical Engine Invented by Charles Babbage, Esq. with notes by trans. / Scientific Memoirs. Vol. 3. 1842. Режим доступа: http://www.fourmilab.ch/babbage/sketch.html, свободный.

К главе 3:

3.1. Эдуард Пройдаков. Виртуальный компьютерный музей. Режим доступа: http://www.computer-museum.ru/, свободный.

3.2. Алферов А.П., Зубов А.Ю., Кузьмин А.С., Черемушкин А.В. Основы криптографии. М.: Гелиос АРВ, 2000. 480 с.

3.3. Boaz Tamir. The Differential Analyzer. 2008. Сайт онлайн-журнала «The Future of Things». Режим доступа: http://thefutureofthings.com/column/1011/the-differential-analyzer.html, свободный.

3.4. Helmut Hoelzer. Электронная энциклопедия «Smart Computing», издатель Sandhills Publishing Company. Режим доступа: http://www.smartcomputing.com/editorial/dictionary/detail.asp?guid=&searchtype=&DicID=17581, свободный.

3.5. Чуканов В.О., Гуров В.В. Логические и арифметические основы и принципы работы ЭВМ. Сайт Интернет Университета Информационных Технологий. Режим доступа: http://www.intuit.ru/department/hardware/archsys/10/, свободный.

3.6. Paul M.B. Vitanyi. Turing machine. Электронная энциклопедия «Scholarpedia». 2009. Режим доступа: http://www.scholarpedia.org/article/Turing_machine, свободный.

3.7. Дж. фон Нейман. Теория самовоспроизводящихся автоматов // М.: Мир. 1971. 382 с.

3.8. Horst Zuse. Z1. Режим доступа: http://user.cs.tu-berlin.de/~zuse/Konrad_Zuse/en/rechner_z1.html, свободный.

3.9. Horst Zuse. Z3. Режим доступа: http://user.cs.tu-berlin.de/~zuse/Konrad_Zuse/en/rechner_z3.html, свободный.

3.10. IBM's ASCC (a.k.a. The Harvard Mark I). Режим доступа: : http://www-03.ibm.com/ibm/history/exhibits/markI/markI_intro.html, свободный.

3.11. Н. Винер. Я – математик // М.: Наука. 1967. 356 с.

3.12. Аллан Р. Макинтош. Компьютер Атанасова // В мире науки. №10, 1988. С. 70–77. Режим доступа: http://vivovoco.rsl.ru/VV/JOURNAL/SCIAM/ATANASOFF/ATANASOFF.HTM, свободный.

3.13. Черняк Л. Colossus, победивший Lorenz. Издательство «Открытые системы». 2004. Режим доступа: http://www.osp.ru/os/2004/10/184688/, свободный.

3.14. Martin H. Weik. The ENIAC Story. Сайт U.S. Army Research Laboratory. Перепечатано с The Journal of the American Ordnance Association, January-February, 1961. Режим доступа: http://ftp.arl.mil/~mike/comphist/eniac-story.html, свободный.

3.15. Малиновский Б.Н. История вычислительной техники в лицах // Киев: фирма «КИТ», ПТОО «А.С.К.», 1995. – 384 с. На сайте «Библиотека Мошкова». Режим доступа: http://lib.ru/MEMUARY/MALINOWSKIJ/0.htm, свободный.

3.16. John von Neumann, First draft of a report on the EDVAC // IEEE Annals of the History of Computing, Vol.15, No. 4, 1993. P. 27–75. Режим доступа: http://cva.stanford.edu/classes/cs99s/papers/vonneumann-firstdraftedvac.pdf, свободный. Перевод: Беркс А., Голдстейн Г., Нейман Дж. Предварительное рассмотрение логической конструкции электронного вычислительного устройства // Кибернетический сборник. М.: Мир, 1964. Вып. 9.

3.17. Руслан Богатырев. Анатомия искусственного интеллекта. Издательство «Открытые системы». Режим доступа: http://www.osp.ru/pcworld/2004/11/169163/, свободный.

К главе 4:

4.1. Пленочная микроэлектроника. Под ред. Л. Холлэнда. // М.: Мир. 1968. 366 с.

4.2. Хоровиц П., Хилл У. Искусство схемотехники. Т.1 // М.: Мир. 1986. 596 с.

4.3. Inside Logic Gates. Режим доступа: http://www.play-hookey.com/digital/electronics/, свободный.

4.4. Джеймс Р. Пауэлл. Квантовое ограничение закона Мура // Технология и конструирование в электронной аппаратуре. 2009, №3. с. 26-27. Переведено из журнала «Proceeding of the IEEE», №8, Vol. 96, 2008. P. 1247-1248.

4.5. С.В. Гапоненко, Н.Н. Розанов, Е.Л. Ивченко, А.В. Федоров. А.М. Бонч-Бруевич, Т.А. Вартанян, С.Г. Пржибельский. Оптика наноструктур. Под редакцией А.В. Федорова: СПб. «Недра», 2005 г. – 326 с.

4.6. Linda Geppert. Quantum transistors: toward nanoelectronics // IEEE Spectrum. 2000. P. 46-51][Suman Datta, S. Mookerjea, D. Mohata, L. Liu, V. Saripalli, V. Narayanan and T. Mayer. Compound Semiconductor Based Tunnel Transistor Logic // CS MANTECH Conference, May 17th-20th, 2010, Portland, Oregon, USA. P. 203-206.

4.7. Создан первый «вероятностный процессор». 19 августа 2010.

Режим доступа: http://habrahabr.ru/blogs/hardware/102152/, свободный.

4.8. Белов П.А., Беспалов В.Г., Васильев В.Н. и др. Оптические процессоры: Достижения и новые идеи. СПб: СПбГУ ИТМО, Ассоциация молодых ученых «Оптика XXI век», 2006. С. 6–31.

4.9. Богатырева В.В., Дмитриев А. Л. Оптические методы обработки информации / Учебное пособие. – СПб: СПбГУИТМО, 2009. – 74 с.

4.10. Нейробудущее. // Хакер, №74, с. 78–81. Режим доступа: http://www.xakep.ru/magazine/xs/074/078/1.asp, свободный.

4.11. Валиев К.А., Кокин А.А. Квантовые компьютеры: надежда и реальность // Ижевск: Регулярная и хаотическая динамика. – 2001. 352 с. Режим доступа: http://aakokin.chat.ru/qc_book.htm, свободный.

4.12. Малинецкий Г.Г., Науменко С.А. Вычисление на ДНК. Эксперименты. Модели. Алгоритмы. Инструментальные средства. // Москва: ИПМ им. М.В.Келдыша РАН. 2005. Режим доступа: http://www.keldysh.ru/papers/2005/prep57/prep2005_57.html, свободный.

4.13. Protein: synthesis. Encyclopedia Britannica. 2006. Режим доступа: http://www.britannica.com/EBchecked/topic-art/479680/1692/Synthesis-of-protein, свободный.

4.14. Adleman L.M. Computing with DNA // Scientific American. August 1998. p. 34-41.

4.15. E. Shapiro, B. Gill, R. Adar, U. Ben-Dor and Y. Benenson. An Autonomous Molecular Computer for Logical Control of Gene Expression // Nature. Vol. 429, 2004. P. 423-429.

4.16. Joanne Macdonald, Yang Li, Marko Sutovic at all. Medium scale integration of molecular logic gates in an automaton // Nano letters. 2006. Vol. 6, No. 11. P. 2598-2603. Режим доступа: http://pubs.acs.org/doi/pdf/10.1021/nl0620684, свободный.

4.17. Renjun Pei, Elizabeth Matamoros, Manhong Liu, Darko Stefanovic& Milan N. Stojanovic. Training a molecular automaton to play a game. Nature Nanotechnology. 2010. Vol. 5, P. 773–777. Режим доступа: http://www.nature.com/nnano/journal/v5/n11/pdf/nnano.2010.194.pdf, свободный.

4.18. Russell Deaton. DNA and Quantum Computers // DNA, quantum, and molecular computing. Proceedings of the Genetic and Evolutionary Computation Conference (GECCO-2001). P. 989-996.

4.19. System/360 Announcement. Режим доступа: http://www-03.ibm.com/ibm/history/exhibits/mainframe/mainframe_PR360.html, свободный.

4.20. G.M.Amdahl, G.A.Blaauw and F.P.Brooks, Architecture of the IBM System/360 // IBM Journal of Research and Development. Vol. 44, № 1/2, 2000. P. 21–36.

4.21. Смирнов А.Д. Архитектура вычислительных систем: Учеб.

пособие для вузов.– М.: Наука. Гл. ред. Физ.-мат. лит., 1990. 320 с.

4.22. По материалам кафедры Вычислительной техники ИТМО.

К главе 5:

5.1. Электроника СБИС. Проектирование микроструктур: Пер. с англ. / Под ред. Айнспрука. – М.: Мир, 1989. – 256 с.

5.2. Лень М., Вайцман И. VLIW: старая архитектура нового поколения. 2002. Режим доступа: http://www.ixbt.com/cpu/vliw.shtml, свободный**.**

5.3. Шагурин И. И. Микропроцессоры и микроконтроллеры фирмы Motorola: Справ. Пособие. – М.: Радио и связь, 1998. – 560 с. Шагурин И. RISC-процессоры PowerPC. Режим доступа: http://www.chipinfo.ru/literature/chipnews/199910/3.html, свободный.

5.4. Luigi Brochard. From Blue Gene to Cell (презентация) // Power.org Moscow, JSCC Technical Day, 2005. Режим доступа: http://parallel.ru/IBM HPC Nov 05 Moscow PowerOrg.pdf, свободный.

5.5. Смирнов А. Будущее архитектуры x86 // Терабайт, 2002, №6 (26), с. 10–15.

К главе 6:

6.1. Basic Cray-1 Architecture. Режим доступа: https://csel.cs.colorado.edu/~csci4576/VectorArch/Cray1Arch.html, свободный.

6.2. Воеводин Вл.В., Капитонова А.П. Методы описания и классификации вычислительных систем // Издательство МГУ, 1994.

6.3. Комолкин А.В., Немнюгин С.А., Программирование для высокопроизводительных ЭВМ. Режим доступа: http://www2.sscc.ru/Links/Litera/HPC/Default.htm, свободный.

6.4. Частиков А.П. Архитекторы компьютерного мира. // СПб: БХВ-Петербург, 2002. 384 с.

6.5. Хетагуров Я. А. Из истории развития специализированных бортовых вычислительных машин // Сайт Виртуального компьютерного музея. Режим доступа: http://www.computer-museum.ru/histussr/special.htm, свободный.

6.6. Подгорнов П. Стратегический ракетный комплекс 15П014 (Р-36М) с ракетой 15А14. Сайт Информационной системы «Ракетная техника». Режим доступа: http://rbase.new-factoria.ru/missile/wobb/15a14/15a14.shtml, свободный.

6.7. Бурцев В. С. Развитие специализированных вычислительных систем ПВО и ПРО // Сайт ИТМиВТ. Москва. 2006. Режим доступа: http://www.ipmce.ru/about/press/articles/politeh2004/, свободный.

К главе 7:

7.1. Paul Baran. On distributed communications: 1. Introduction to distributed communications networks // The Rand Corp., Santa-Monica, California, 1964. Режим доступа: http://www.rand.org/pubs/research_memoranda/2006/RM3420.pdf, свободный.

7.2. Mitch Waldrop. DARPA and the Internet Revolution. Сайт DAPRA. 2008. Режим доступа: http://www.darpa.mil/Docs/Internet_Development_200807180909255.pdf, свободный.

7.3. Jonathan Strickland. How ARPANET Works. Режим доступа: http://computer.howstuffworks.com/arpanet.htm/printable, свободный.

7.4. Медведев Д. Л. Основоположники сети Интернет. // Электросвязь: история и современность. № 3–4, 2006 г., с. 21–26. Режим доступа: http://www.computer-museum.ru/frgnhist/internet_o.htm, свободный.

7.5. ALOHAnet. Режим доступа: http://en.wikipedia.org/wiki/ALOHAnet, свободный.

7.6. H. Norman Abramson. Development of the ALOHANET // IEEE Transactions on Information Theory, Vol. IT-31, No. 2, March 1985. P. 119–123. Режим доступа: http://citeseerx.ist.psu.edu/viewdoc/download?doi=10.1.1.119.8431&rep=rep1&type=pdf

7.7. Richard H. Thompson and Larry W. Swanson. Hypothesis-driven structural connectivity analysis supports network over hierarchical model of brain architecture // PNAS. August 9, 2010, doi:10.1073/pnas.1009112107. Краткий перевод: Мозг больше похож на интернет, чем на компьютер. Сайт БиБиСи. Режим доступа: http://www.bbc.co.uk/russian/science/2010/08/100810_brain_like_internet.shtml?s, свободный.

7.8. Ed Bullmore and Olaf Sporns. Complex brain networks: graph theoretical analysis of structural and functional systems // Nature reviews. Neuroscience. March 2009. Vol. 10. P. 186-198.

К главе 8:

8.1. Коньков К.А., Карпов В.Е. Краткая история эволюции вычислительных систем. Сайт Интернет-университета информационных технологий. Режим доступа: http://www.intuit.ru/department/os/osintro/1/2.html, свободный.

8.2. Doug Ross. MIT's (the world's?) first Operating System. // по материалу D. Ross, A personal view of the personal work station: some firsts in the fifties в сборнике A history of personal workstations, Нью-Йорк, 1988. Режим доступа:

http://www.csail.mit.edu/timeline/timeline.php/timeline.php?query=event&id=3, свободный.

8.3. Bernard A. Galler, The World's First Computer Operating System Implemented at General Motors Research Labs in Warren, Michigan in 1955. // по материалу Henry S. Tropp, FORTRAN Anecdotes, IEEE Annals of the History of Computing, vol. 06, no. 1, pp. 59–64, Jan–Mar, 1984. Режим доступа: http://millosh.wordpress.com/2007/09/07/the-worlds-first-computer-operating-system-implemented-at-general-motors-research-labs-in-warren-michigan-in-1955/, свободный.

8.4. Федорчук А. Введение в POSIX'ивизм. 2005. Режим доступа: http://www.linuxcenter.ru/lib/books/posixbook/, свободный.

8.5. М. Тим Джонс. Анатомия ядра Linux. 2007. Режим доступа: http://www.ibm.com/developerworks/ru/library/l-linux-kernel/, свободный.

8.6. Лидовский В. Жизнь после смерти. Сайт «Компьютерра-онлайн». Режим доступа: http://www2.computerra.ru/hitech/34452/, свободный.

К главе 9:

9.1. Михаил Плискин. Эволюция языков программирования. Режим доступа: http://schools.keldysh.ru/sch444/MUSEUM/LANR/evol.htm, свободный.

9.2. Первый в мире язык программирования. 2007. Режим доступа: http://lktalks.blogspot.com/2007/06/blog-post_24.html, свободный.

9.3. Дмитрий Румянцев. Кошмар программиста // Upgrate. Декабрь, № 11, 2009. Режим доступа: http://www.computery.ru/upgrade/numbers/2004/170/history_170.htm, свободный.

9.4. Томас Курц и Джон Кемени. Сайт «История компьютера». Режим доступа: http://chernykh.net/content/view/174/184/, свободный.

9.5. PL/1. Режим доступа: http://en.wikipedia.org/wiki/PL/1, свободный.

9.6. Ada (programming language). Режим доступа: http://en.wikipedia.org/wiki/Ada_(programming_language), свободный.

9.7. 12 языков программирования, которые потрясли мир (тем, что на них нельзя программировать). Режим доступа: http://www.xakep.ru/post/39418/default.asp, свободный. (оригинал – Ghosts in the Machine: 12 Coding Languages That Never Took Off. Режим доступа: http://www.softwaredeveloper.com/features/ghosts-in-machine-071207/, свободный.)

9.8. Eric S. Raymond. Intercal – the Language From Hell. Режим доступа: http://catb.org/~esr/intercal/stross.html, свободный.

9.9. Java2k. Режим доступа: http://p-nand-q.com/humor/programming_languages/java2k.html, свободный.