Организация ЭВМ и вычислительных систем

Лекции 1–2. Основные концепции

П. А. Макаров





1 и 15 сентября 2026 г.

Содержание

1. Знакомство и основные сведения о курсе
2. Литература
3. Основные понятия
4. Инструменты разработчика
5. Основы операционных систем
6. Информация и кодирование
7. Математические основы
8. Контрольные вопросы и задания

Знакомство и основные сведения о курсе

- Дисциплина: Организация ЭВМ и ВС.
- Преподаватели:
  - Макаров Павел Андреевич, makarovpa@ipm.komisc.ru, @mkrvpa;
  - Замыслов Михаил Владимирович.
- Трудоёмкость: 32 ч. лекций + 32 ч. п. з. + 64 ч. лаб. раб.
- Форма аттестации: 2 зачёта / зачёт + экзамен.
- Сайты в поддержку курса:
  - https://www.makarovpa.ru/cso;
  - https://mpa.komisc.ru.

Знакомство и основные сведения о курсе

Распределение часов по семестрам

Таблица 1: Структура курса

Семестр	**Лекции**	**Практики**	**Лабораторные работы**
Осень	1 – 8	1 – 8	Часть 1: 1 – 16
Аттестация:			**зачёт**
Весна	9 – 16	9 – 16	Часть 2: 1 – 8
Аттестация:			**зачёт (2 курс) / экзамен (3 курс)**

Зачёт

Итоговый балл

$$S = \sum_{i=1}^{N} L_i + T, \tag{1}$$

где балл за каждую лабораторную работу $L_i \leqslant \dfrac{70}{N}$, а $T \leqslant 30$ — промежуточный контроль в форме теста на зачёте.

Итоговый балл, S	< 60	$60 \div 100$
Оценка	не зачтено	зачтено

Таблица 2: Перевод баллов в оценки зачёта

Экзамен

Итоговый балл

$$S = \sum_{i=1}^{N} L_i + E, \tag{2}$$

где балл за каждую лабораторную работу $L_i \leqslant \dfrac{70}{N}$, а $E \leqslant 30$ — промежуточный контроль на экзамене. Работа на экзамене может быть заменена выполнением экзаменационного проекта и сдачей отчёта по нему: $E \leftrightarrow P$.

Итоговый балл, S	< 60	$60 \div 69$	$70 \div 89$	$90 \div 100$
Оценка	2	3	4	5

Таблица 3: Перевод баллов в оценки экзамена

Литература

Основная

1. *Петцольд Ч.* Код: тайный язык информатики.
2. *Гук М.Ю.* Аппаратные средства IBM PC: Энциклопедия.
3. *Максимов Н.В., Попов И.И., Партыка Т.П.* Архитектура ЭВМ и вычислительные системы.
4. *Брайант Р.Э., О'Халларон Д.Р.* Компьютерные системы: архитектура и программирование (Computer Systems: A Programmer's Perspective).
5. *Patt Y.N., Patel S.J.* Introduction to Computing Systems: From Bits & Gates to C/C++ & Beyond.
6. *Hyde R.* The Art of Assembly Language.
7. *Марек Р.* Ассемблер на примерах. Базовый курс.

Литература

Дополнительная — часть 1

1. *Столяров А.В.* Программирование: введение в профессию. Т. 1, 2.
2. *Джордейн Р.* Справочник программиста персональных компьютеров типа IBM PC, XT и AT.
3. *Нортон П., Уилтон Р.* IBM PC и PS/2. Руководство по программированию.
4. Разработка на платформе Эльбрус. `https://dev.mcst.ru/`
5. *Керниган Б., Ритчи Д.* Язык программирования C.
6. *Богатырёв А.* Хрестоматия по программированию на Си в Unix.

Литература

Дополнительная — часть 2

7. *Абель П.* Язык Ассемблера для IBM PC и программирования.
8. *Юров В.И.* Assembler. Учебник для вузов.
9. *Калашников О.* Ассемблер — это просто. Учимся программировать.
10. *Акчурин Э.А.* Машинно-зависимые языки программирования.
11. *Жуковский Д.А., Жуковский А.Г., Швиденко С.А.* Машинно-зависимые языки программирования (языки ассемблера).
12. *Бурдаев О.В., Иванов М.А., Тетерин И.И.* Ассемблер в задачах защиты информации.
13. *Квочкин Д.О., Макаров П.А., Устюгов В.А.* Микросхемотехника. Изд-во СГУ им. Питирима Сорокина, Ч.1 — 2016, 64 с; Ч.2 — 2017, 67 с.
14. *Устюгов В.А., Макаров П.А.* Практикум по ассемблеру x86. Примеры и задачи. Изд-во СГУ им. Питирима Сорокина — 2021, 54 с.
15. *Устюгов В.А., Чуфырев А.Е.* Программирование микроконтроллеров AVR. Изд-во СГУ им. Питирима Сорокина — 2017, 51 с.

Что такое компьютер?

Определение 1

*Под **компьютерами** будем подразумевать любые электронные устройства, способные исполнять программы, независимо от их конкретной архитектуры. Часто в качестве синонима используются понятия **вычислительная машина/система, ЭВМ**.*

Определение 2

***Архитектура компьютера** — это концептуальная модель компьютерной системы, определяемая её компонентами и их взаимодействиями между собой и окружением. Кроме того, в понятие архитекуры включаются принципы её проектирования и развития.*

Определение 3

Процессор *— специализированное электронное устройство, предназначенное для выполнения тех или иных операций над некоторой информацией по заданной программе.*

Определение 4

Контроллер *— это практически то же самое, что и процессор, с той разницей, что программа его работы, как правило, никогда не меняется в процессе функционирования системы.*

μЭВМ: процессор, ОЗУ (RAM), ПЗУ (ROM), системы обработки прерываний, таймеры и другие устройства.

- По используемым видам электрических сигналов:
  - аналоговые;
  - цифровые.
- По функциональному назначению:
  - CPU;
  - FPU (NPU);
  - GPU.

Основные понятия

Примеры стандартов цифровых микросхем

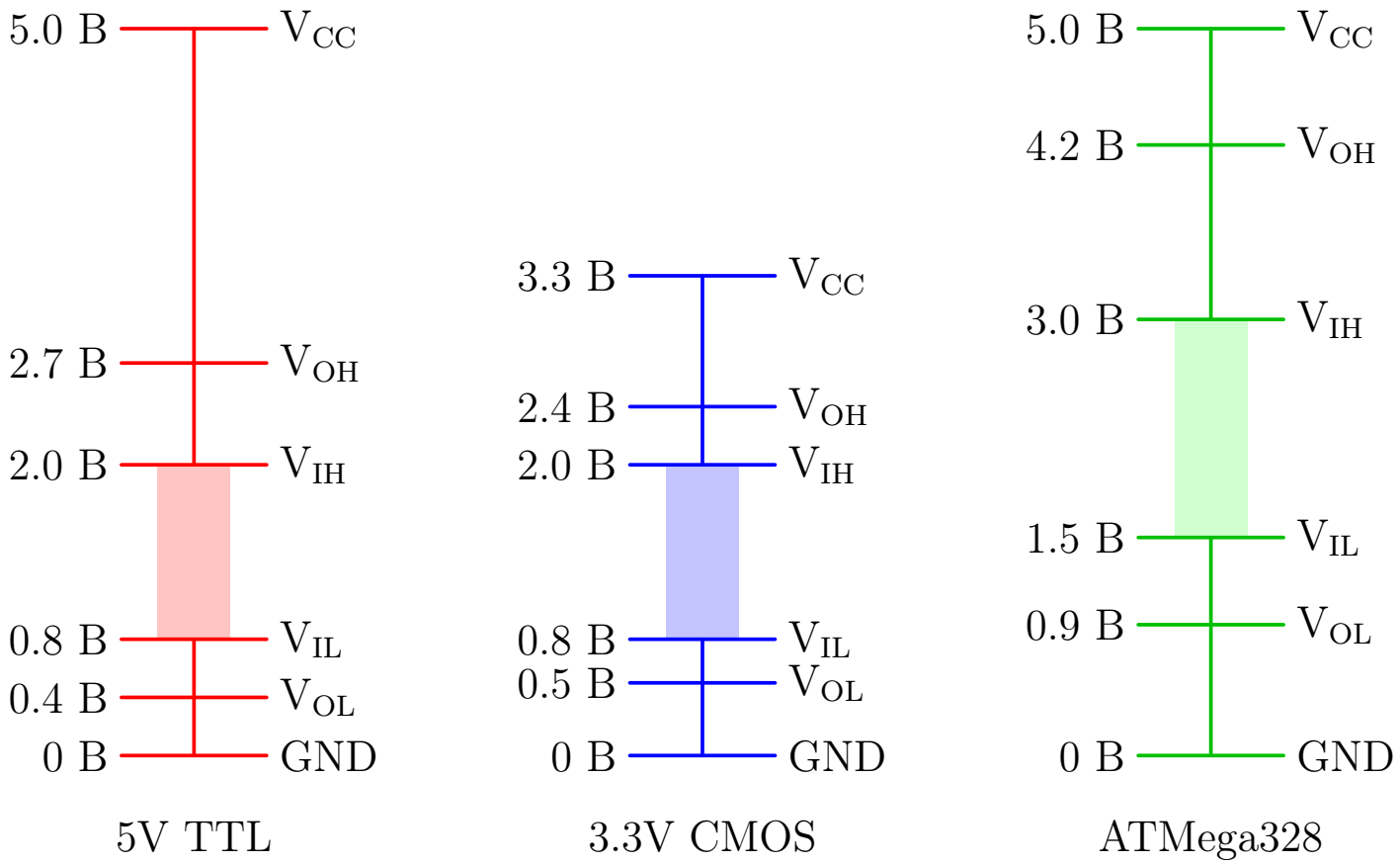


Рис. 1: Логические уровни некоторых типов дискретных сигналов

Что такое программа?

Определение 5

Программа *— это текст, представляющий собой исчерпывающее описание действий исполнителя и написанный на понятном ему языке.*

Определение 6

Язык программирования *— это формальный язык, предназначенный для записи компьютерных программ. Язык программирования представляет собой совокупность лексических, синтаксических и семантических правил.*

- По уровню применяемых абстракций.
- По режиму исполнения программ.
- По используемой типизации.
- По поддержке различных компьютерных архитектур.
- По назначению.
- По возможности следования парадигме программирования.

Основные понятия

Примеры текстов, написанных на языках программирования

```c
#include <stdio.h>

int main() {
  printf("Hello, world!\n");
  return 0;
}
```

Листинг 1: Пример текста, написанного на языке C

```python
#!/usr/bin/python3

print("Hello, world!")
```

Листинг 2: Пример текста, написанного на языке Python

Определение 7

Исходный код *— это текст компьютерной программы на каком-либо языке программирования или языке разметки, который может быть прочтён человеком. Исходный код транслируется в исполняемый код целиком до запуска программы при помощи транслятора либо исполняется непосредственно при помощи интерпретатора. В широком смысле, исходный код — это любые входные данные для транслятора.*

Определение 8

Транслятор *— это программа, выполняющая* ***трансляцию****, то есть преобразование программы, представленной на одном из языков программирования, в программу на другом языке.*

Процесс трансляции программ, написанных на языке C

1. Обработка исходного текста *препроцессором*.
2. *Компиляция*.
3. *Ассемблирование*.
4. *Компоновка*.

А является ли компьютерной программой текст, приведённый в следующем листинге?

```
1  ; nasm -f elf64 null.nasm && ld null.o -o null

2  %define SYSCALL_EXIT 60

3  global _start:
4  _start:
5    mov rax, SYSCALL_EXIT
6    mov rdi, 0
7    syscall
```

Листинг 3: Пример текста, написанного на языке NASM, для вычислительных машин, работающих под управлением ОС на ядре Linux

Определение 9

*Термин **ассемблер** применяется в информатике в двух значениях.*

1. *С одной стороны это слово **служит общим названием семейства низкоуровневых языков программирования.** В этом контексте различают множество различных **диалектов ассемблера** (в частности, вариации ассемблера x86, ARM, PIC и т. п.), а также два основных **синтаксиса:** Intel и AT&T.*
2. *С другой стороны, под этим подразумевают именно **транслятор, который преобразует исходный код программы,** написанный на конкретном языке ассемблера в машинный код некоторого формата (более точно — объектный код) для данной архитектуры компьютера.*

Определение 10

Интерпретатор *— программа, выполняющая* ***интерпретацию*** *— построчный анализ, обработку и выполнение исходного кода программы.*

Виды интерпретаторов

1. Простые.
2. Компилирующего типа.

Алгоритм работы простого интерпретатора

1. прочитать инструкцию;
2. проанализировать её и определить требуемые действия;
3. выполнить необходимые действия;
4. если не достигнуто условие завершения программы, прочитать следующую инструкцию и перейти к пункту 2.

Некоторые интерпретаторы могут работать не только в пакетном режиме (batch mode — основной режим для выполнения скриптов), но и в режиме диалога (interactive mode) или *цикла чтения-вычисления-печати* (REPL — read-eval-print loop).

Определение 11

Отладчиком *называется инструмент, необходимый для интерактивного анализа исходного кода программы с целью поиска и устранения ошибок или исследования принципа её работы.*

Замечание 1

Работа с некоторыми отладчиками (в том числе и с отладчиками самого низкого уровня) очень похожа на работу с интерпретатором в режиме диалога.

Инструменты разработчика

Какие инструменты нужны программисту?

Инструменты разработчика

- Текстовый редактор.
- Набор транслятора либо интерпретатор.
- Библиотека необходимых функций и компонентов.
- Система сборки.
- Отладчик.
- Статический анализатор (Static Analyzer).
- Профилировщик кода (Profiler).
- Система контроля версий.
- Генератор документации.

Определение 12

Интегрированная среда разработки *— это комплекс программных средств, используемый программистами для разработки программного обеспечения (IDE — Integrated Development Environment).*

Инструменты разработчика

- Текстовый редактор. `Notepad++, vim`
- Набор транслятора либо интерпретатор. `nasm, gcc`
- Библиотека необходимых функций и компонентов. `glib`
- Система сборки. `make`
- Отладчик. `debug, gdb`
- Статический анализатор (Static Analyzer).
- Профилировщик кода (Profiler).
- Система контроля версий. `git`
- Генератор документации.

Определение 12

Интегрированная среда разработки *— это комплекс программных средств, используемый программистами для разработки программного обеспечения (IDE — Integrated Development Environment).*

Определение 13

Операционная система *— это комплекс взаимосвязанных программ, предназначенных для управления ресурсами компьютера и организации взаимодействия с пользователем.*

Основные этапы развития компьютеров и операционных систем

- Пакетный режим.
- Разделение времени и многозадачность.
- Разделение полномочий.
- Системы реального времени.
- Гибридные системы.

Нужны ли операционные системы?

Операционные системы используются

- если нужен универсальный механизм хранения данных;
- для предоставления программам системных библиотек с часто используемыми подпрограммами;
- для распределения полномочий;
- необходима возможность имитации “одновременного” исполнения нескольких программ на одном компьютере;
- для управления процессами выполнения отдельных программ.

Современные универсальные ОС

- используют файловые системы;
- многопользовательские;
- многозадачные.

- Основные функции:
  - загрузка программ в оперативную память и их выполнение.
  - исполнение запросов программ.
  - стандартизованный доступ к периферийным устройствам.
  - управление оперативной памятью.
  - управление доступом к данным на носителях, организованным в определённой файловой системе.
  - обеспечение пользовательского интерфейса.
  - сохранение информации об ошибках системы.
- Дополнительные функции:
  - параллельное или псевдопараллельное выполнение задач.
  - эффективное распределение ресурсов вычислительной системы между процессами.
  - разграничение доступа различных процессов к ресурсам.
  - организация надёжных вычислений.
  - взаимодействие между процессами: обмен данными, взаимная синхронизация.
  - защита системы и данных/программ пользователя.

Группы компонентов ОС

- Ядро и планировщик. Драйверы устройств, непосредственно управляющие оборудованием. Сетевая и файловая системы.
- Системные библиотеки.
- Оболочка с утилитами.

Определение 14

Ядро *— центральная часть операционной системы, управляющая ресурсами компьютера, выполнением процессов и обеспечивающая координированный доступ процессов к ресурсам.*

Основные ресурсы системы

- процессорное время;
- память;
- устройства ввода-вывода;
- файловая система;
- сетевая система.

Основные объекты ядра

- Процессы.
- Файлы.
- События.
- Потоки.
- Семафоры.
- Мьютексы.
- Каналы.

Определение 15

Системный вызов *— это элемент пользовательского интерфейса ядра, представляющий собой обращение прикладной программы к ядру операционной системы для выполнения той или иной операции.*

Определение 16

Программный интерфейс приложения (API — Application Programming Interface) *— это описание способов взаимодействия одной компьютерной программы с другими.*

Определение 17

Двоичный интерфейс приложений (ABI — Application Binary Interface) *— это набор соглашений для доступа приложения к операционной системе и другим низкоуровневым сервисам, спроектированный для переносимости исполняемого кода между машинами, имеющими совместимые ABI.*

Задуманная и реализованная в 1969 году коллективом Кена Томпсона (включая Денниса Ритчи и Брайана Кернигана), операционная система UNIX (первоначально UNICS) вобрала в себя многие черты более ранних систем, но обладала целым рядом свойств, отличающих её от большинства предшественниц:

- простая метафорика (два ключевых понятия: вычислительный процесс и файл);
- компонентная архитектура: принцип "одна программа — одна функция" плюс мощные средства связывания различных программ для решения возникающих задач ("оболочка");
- минимизация ядра и количества системных вызовов;
- независимость от аппаратной архитектуры и реализация на машиннонезависимом языке программирования (язык программирования Си стал побочным продуктом разработки UNIX);
- унификация файлов.

— *любые данные или факты,* которые представляют какой-либо интерес.
— *сообщения,* передаваемые в форме знаков или сигналов.
— часть *знаний,* которая используется для ориентирования, активного действия, управления, т. е. в целях сохранения, совершенствования, развития системы.

Несколько определений информации:

- отрицание энтропии (Л. Бриллюэн);
- мера сложности структур (Моль);
- отражённое разнообразие (Урсул);
- содержание процесса отражения (Тузов);
- вероятность выбора (Яглом);
- снятая неопределенность наших знаний о чем-то (К. Шеннон);

Информация и кодирование

Классификации информации

- По уровням сложности:
  - сигнал;
  - сообщение/документ;
  - информационный массив;
  - информационный ресурс.
- По типу применяемых сигналов:
  - аналоговая (непрерывная);
  - цифровая (дискретная);
- По уровням доступа и организации:
  - данные в регистровой памяти;
  - данные в оперативной памяти;
  - файлы данных на внешних устройствах;
  - базы данных.
- По способам кодирования и представления:
  - цифровая (вычислительные данные, двоичные);
  - символьная или текстовая (алфавитно-цифровая, строчная);
  - графическая.

Определение 18

Информация *— это данные и контекст.*

Пример 1

```
$ hexdump -C data.dat

00000000  b8 3c 00 00 00 bf 00 00  00 00 0f 05              |.<..........|
0000000c

$ objdump -d null

null:     file format elf64-x86-64

Disassembly of section .text:

0000000000401000 <_start>:
  401000: b8 3c 00 00 00       mov    $0x3c,%eax
  401005: bf 00 00 00 00       mov    $0x0,%edi
  40100a: 0f 05                syscall
```

Определение 19

Битом *(bit) информации называется один двоичный разряд.* ***Байт*** *(byte) — это группа из восьми (октет) битов*[1]*.* ***Машинное слово*** *(word) — это двоичное число определенной разрядности, зависящей от архитектуры вычислительной машины.*

[1] В исторической ретроспективе, байт не всегда формировался именно восемью битами, но на данный момент иное уже не имеет места.

Определение 19

Бит *(от англ.* ***BI****nary digi****T*** *— двоичная единица) — единица измерения количества информации, равная количеству информации, содержащемуся в опыте, имеющем два равновероятных исхода. Это наименьшая единица информации в цифровом компьютере, принимающая значения «0» или «1».*

Определение 20

Машинное слово *(МС) – упорядоченное множество двоичных разрядов, используемое для хранения команд программы и обрабатываемых данных. Каждый разряд, называемый битом, — это двоичное число, принимающее значения только «0» или «1». Разряды в МС обычно нумеруются справа налево начиная с 0. Количество разрядов в МС называется размерностью МС или его разрядностью. Типовая длина МС — 16, 32 или 64 бита.*

Определение 21

Байт *— машинное слово минимальной размерности, адресуемое в процессе обработки данных.*

Определение 22

Код (code) *— совокупность знаков, символов и правил представления информации.*

Определение 23

Кодируемые (обозначаемые) *элементы входного алфавита обычно называют* ***символами.***

Определение 24

Кодирующие (обозначающие) *элементы выходного алфавита называются* ***знаками.*** *Количество различных знаков в выходном алфавите называют* ***значностью*** *(****-арностью, -ичностью,*** *например «бинарный» или «двоичный» код). Количество знаков в кодирующей последовательности для одного символа —* ***разрядностью кода.***

Определение 25

*При **последовательном коде** каждый временно́й такт предназначен для отображения одного разряда символа. Здесь все разряды символа фиксируются по очереди одним и тем же элементом и проходят через одну и ту же линию передачи (например, радио- или оптические сигналы либо передача данных по двум проводам, двухжильному кабелю).*

Определение 26

*При **параллельном коде** все знаки символа представляются в одном временном такте, каждый знак проходит через отдельную линию (например, по четырём проводам, четырёхжильному кабелю), образуя символ (т. е. символ передается в один приём, в один момент времени).*

Пример 2

Широко известным примером кода является азбука Морзе, в которой буквы латиницы (или кириллицы) и цифры кодируются сочетаниями из «точек» и «тире». Применительно к азбуке Морзе:

- *символами являются элементы языкового алфавита (буквы A–Z или А–Я) и цифровой алфавит (здесь — цифры 0–9);*
- *знаками являются «·» и «—» (или «+» и «−» либо «1» и «0», короче — любые два разных знака);*
- *поскольку знаков два, то азбука Морзе является двузначным (бинарным, двоичным) кодом;*
- *поскольку число знаков в азбуке Морзе колеблется от 1 (буквы E, T) до 5 (цифры), здесь имеет место код с переменной разрядностью (часто встречающиеся в тексте символы обозначены более короткими кодовыми комбинациями, нежели редкие символы);*

Наименование	Расшифровка	Другие названия	Разрядность
Baudot	Код Бодо	IA-1	5
M2	CCITT-2	IA-2	5
ASCII-7	American Standard Code for Information Interchange	ISO-7, IA-5, USASCII, ANSI X3.4	7
ASCII-8	—		8
EBCDIC	Expanded Binary Coded Decimal Information Code		8
Hollerith	Код Холлерита	Код перфокарт	12
UNICODE	UNIversal CODE		16

Таблица 4: Некоторые наиболее известные коды

Коды с обнаружением ошибок

- Контроль чётности.
- Циклический избыточный код обнаружения ошибок (CRC — Cyclic Redundance Check).

Корректирующие коды (ECC — Error Correction Code)

- Код Хемминга (деление по модулю 4).
- Помехоустойчивые коды Рида — Соломона (Reed — Solomon).

Определение 27

Система счисления *— это совокупность правил записи чисел цифровыми символами.*

Определение 28

Позиционными системами счисления *называются такие системы, в которых действительное значение цифры зависит от её позиции (порядкого номера) в записи числа. Другими словами, каждая позиция в записи имеет свой вес.*

Определение 29

Непозиционная система счисления *— это система, в которой значение символа не зависит от его положения в числе.*

Пример 3

Римская система счисления — это пример смешанной системы: IV, V, VI, XIV, ...

Основные характеристики позиционной системы счисления

1. основание системы — это число различных цифр в системе: B;
2. наибольшая цифра: $B - 1$.

Представление чисел в позиционной системе

$$M = \sum_{i=-k}^{n-1} A_i \cdot B^i = A_{n-1} \dots A_1 A_0 . A_{-1} A_{-2} \dots A_{-k}. \qquad (3)$$

Пример 4

$$\pi \approx 3.1415 = 3 \cdot 10^0 + 1 \cdot 10^{-1} + 4 \cdot 10^{-2} + 1 \cdot 10^{-3} + 5 \cdot 10^{-4}. \quad (4)$$

Система	B	Обозначение	С
Двоичная	2	b (binary)	Префикс `0b`
Восьмеричная	8	o (octal)	Ведущий `0`
Десятичная	10	d (decimal)	Default
Шестнадцатеричная	16	h (hexadecimal)	Префикс `0x`

Таблица 5: Основные позиционные системы счисления

Пример 5

$$25_{10} = 11001_2 = 31_8 = 19_{16}. \tag{5}$$

Замечание 2

Система счисления определяет только правила записи числа. Само число и его свойства от выбора системы не зависят, т. е. простое число всегда останется простым, чётное — чётным и т. д.

Теорема 1

Если основание одной системы счисления представляет собой натуральную степень n основания другой системы счисления, то одна цифра первой системы в точности соответствует n цифрам второй системы.

oct	bin	hex	bin	hex	bin
0	000	0	0000	8	1000
1	001	1	0001	9	1001
2	010	2	0010	A	1010
3	011	3	0011	B	1011
4	100	4	0100	C	1100
5	101	5	0101	D	1101
6	110	6	0110	E	1110
7	111	7	0111	F	1111

Таблица 6: Связь представлений чисел в основных системах

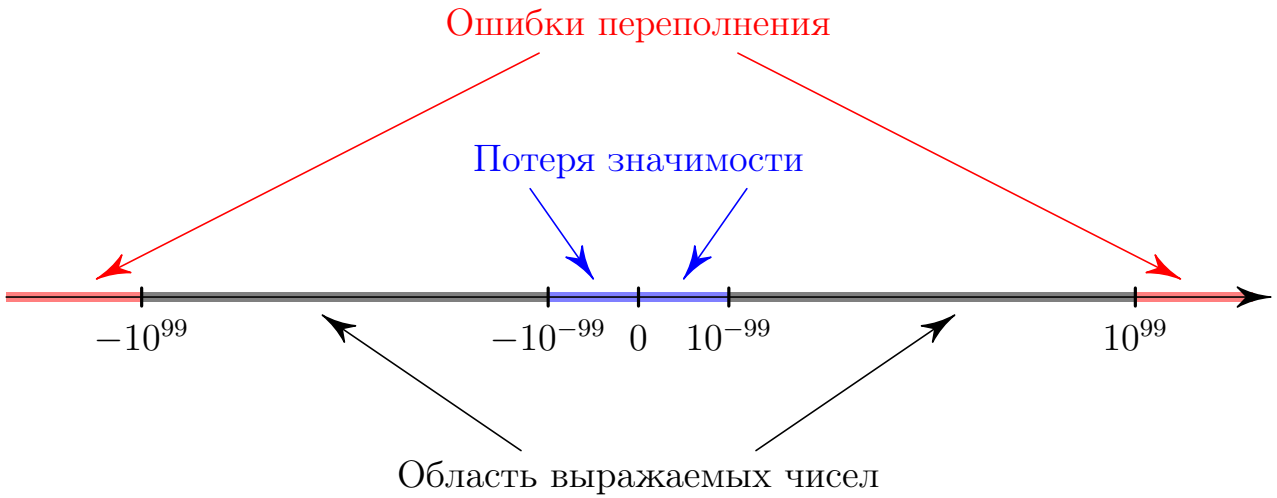


Рис. 2: Числа конечной точности как подмножество $\mathbb{R}$ на примере двузначного порядка

Последствия незамкнутости чисел конечной точности

1. Ошибки переполнения.
2. Ошибки потери значимости.
3. Результат операции: NaN — Not a Number.

Замечание 3

Для чисел конечной точности в общем случае не имеет места ассоциативность и дистрибутивность:

$$a + (b - c) \neq (a + b) - c, \tag{6}$$

$$a \cdot (b - c) \neq a \cdot b - a \cdot c. \tag{7}$$

Ещё одно отличие чисел конечной точности от действительных чисел состоит в том, что ***они заполняют числовую ось с переменной плотностью,*** *что может приводит к ошибкам округления и накоплению погрешности вычислений.*

Определение 30

Машинный ноль *— числовое значение с таким отрицательным порядком, которое воспринимается машиной как ноль.*

Определение 31

Машинный эпсилон ε *— это числовое значение, меньше которого невозможно задавать относительную точность для любого алгоритма, возвращающего вещественные числа. Формально* ε *определяют двумя альтернативными способами:*

1. *как минимальное из чисел* ε*, для которого* $1 + \varepsilon > 1$ *при машинных расчётах с числами данного типа;*
2. *либо как максимальное* ε*, для которого справедливо равенство* $1 + \varepsilon = 1$*.*

Пример 6

В языке Си машинный эпсилон определяют константы `FLT_EPSILON`*,* `DBL_EPSILON` *и* `LDBL_EPSILON`*.*

Пример 7

Пусть для представления чисел в основных системах счисления используется 5 разрядов в целой части числа и 3 — в дробной части.

```
+01221.075;
+000B0.125h;
-70301.123o.
```

Диапазон значащих чисел N в системе счисления с основанием B при наличии n разрядов в целой части и k разрядов в дробной части числа без учета знака:

$$B^{-k} \leqslant N \leqslant B^n - B^{-k}. \tag{8}$$

В ЭВМ числа с ФТ чаще всего хранятся в форматах:

- полуслово — это обычно 16 бит или 2 байта (`short, dw`);
- слово — 32 бита или 4 байта (`int, dd`);
- двойное слово — 64 бита или 8 байтов (`long int, dq`).

1. Знаковая система. Старший бит кодирует знак числа. Обычно считается, что нуль используется для записи положительных чисел, а единица обозначает отрицательные.
2. Дополнение до единицы (или обратный код). Знаковый бит + инверсия всех остальных разрядов.
3. Дополнение до двух (дополнительный код). Знаковый бит + инверсия + добавление 1.
4. Система со смещением (как правило, используются варианты Excess-2^{n-1} или Excess-$2^{n-1}-1$). Сумма исходного числа со смещением. Здесь предполагается, что для записи числа используется n двоичных разрядов.

Пример 8

Пусть требуется записать отрицательное число -27_{10} в $n = 8$ двоичных разрядах. Тогда в знаковой системе ему соответствует представление 10011011, а во второй, третьей и четвёртой (excess-128) — 11100100, 11100101 и 01100101, соответственно.

$$M = \pm f \cdot B^e. \tag{9}$$

$$M = \pm f \cdot B^e. \tag{9}$$

Пример 9

$$101.01101_2 = 4 + 1 + 1/4 + 1/8 + 1/32 = 5.40625_{10}. \tag{10}$$

$$M = \pm f \cdot B^e. \tag{9}$$

Пример 9

$$101.01101_2 = 4 + 1 + 1/4 + 1/8 + 1/32 = 5.40625_{10}. \tag{10}$$

Целая часть исходного числа в двоичной системе $5_{10} = 101_2$.

$$0.40625 \cdot 2 = 0.8125$$
$$0.8125 \cdot 2 = 1.625$$
$$0.625 \cdot 2 = 1.25$$
$$0.25 \cdot 2 = 0.5$$
$$0.5 \cdot 2 = 1.0$$

$$\Rightarrow 0.40625_{10} = 0.01101_2$$

Окончательно получаем $5.40625_{10} = 101.01101_2$.

Замечание 4

В виде конечной двоичной дроби представимы такие и только такие рациональные числа, знаменатель которых является степенью 2.

Определение 32

Нормализованными *называются такие числа с плавающей точкой M, для которых выполнено условие*

$$B^{-1} \leqslant |f| < 1. \tag{11}$$

Пример 10

Числа, использованные в Примере 7, в нормализованной форме будут выглядеть так:

$$\begin{aligned} +01221.075 &= +0.1221075_{10} \cdot 10^4; \\ +000B0.125h &= +0.B0125_{16} \cdot 16^2; \\ -70301.123o &= -0.70301123_8 \cdot 8^5. \end{aligned} \tag{12}$$

Утверждение 1

Правила представления чисел с плавающей точкой регламентирует международный стандарт IEEE 754-2008, согласно которому нормализованными двоичными числами называют такие числа для которых $1 \leqslant |f| < 2$.

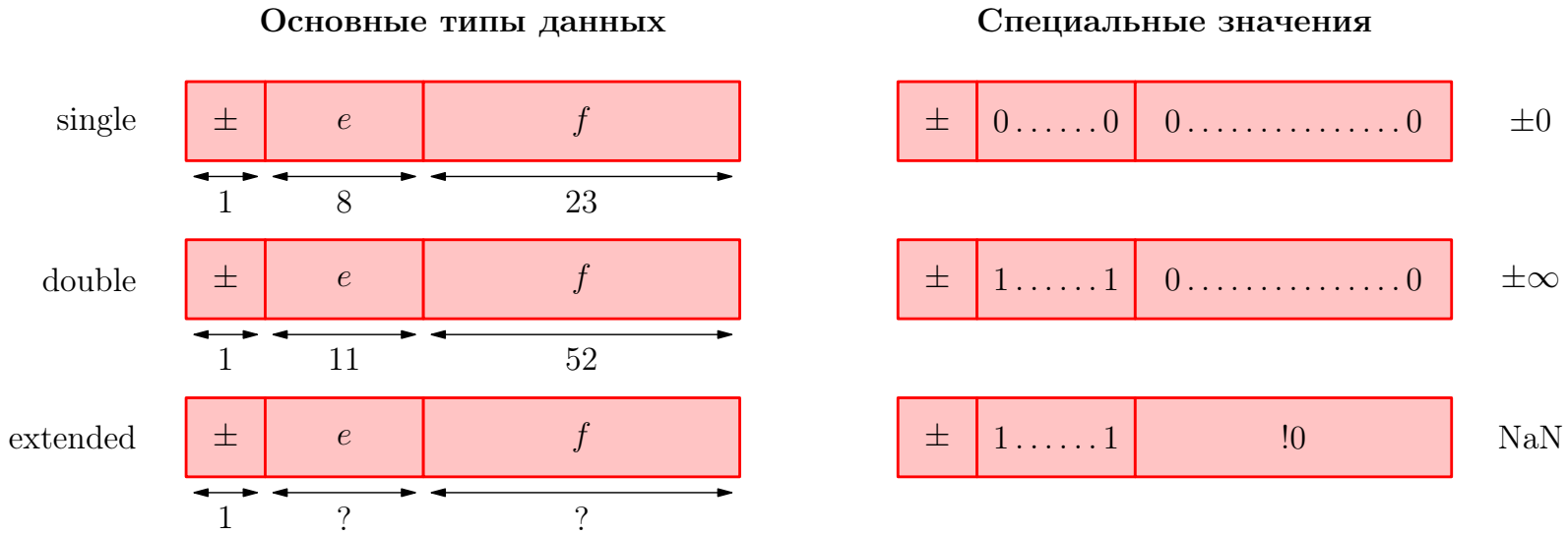


Рис. 3: Некоторые соглашения стандарта IEEE 754-2008

- Обычная двоичная арифметика.
- Двоично-десятичная арифметика.
- Циклическая арифметика (WrapAround).
- Арифметика с насыщением (Saturation Arithmetic — SArith).
- Векторные операции над упакованными данными («упакованная арифметика» — Packed Arithmetic).
- «Горизонтальная» и «вертикальная» арифметика.

$x = 0$, если $x \neq 1$	$x = 1$, если $x \neq 0$
$x = 0,\ \overline{x} = 1$	$x = 1,\ \overline{x} = 0$
$1 \cdot 0 = 0 \cdot 1 = 0$	$1 + 0 = 0 + 1 = 1$
$1 \cdot 1 = 1$	$0 + 0 = 0$
$0 \cdot 0 = 0$	$1 + 1 = 1$
$x \cdot 1 = x$	$x + 0 = x$
$x \cdot 0 = 0$	$x + 1 = 1$
$x \cdot x = x$	$x + x = x$
$x \cdot \overline{x} = 0$	$x + \overline{x} = 1$

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

$x \cdot y = y \cdot x$	$x + y = y + x$
$x + x \cdot y = x$	$x \cdot (x + y) = x$

$$(x + \overline{y}) \cdot y = x \cdot y$$

$x \cdot y + x \cdot \overline{y} = x$	$(x + y)(x + \overline{y}) = x$
$(x \cdot y) \cdot z = x \cdot (y \cdot z)$	$(x + y) + z = x + (y + z)$
$(x + y) \cdot (x + z) = x + y \cdot z$	$x \cdot y + x \cdot z = x \cdot (y + z)$
$\overline{x \cdot y \cdot z} = \overline{x} + \overline{y} + \overline{z}$	$\overline{x + y + z} = \overline{x} \cdot \overline{y} \cdot \overline{z}$

Таблица 7: Основы булевой алгебры

Контрольные вопросы и задания

1. С какими архитектурами компьютеров вы работали? Каковы их основные особенности? Наидите и изучите информацию о неизвестных вам архитектурах.
2. Каковы, на ваш взгляд, преимущества и недостатки использования IDE для разработки ПО? С какими инструментами разработчика у вас есть опыт работы?
3. Назовите существующие интерфейсы взаимодействия пользователя с операционной системой. Каковы их преимущества и недостатки?
4. Ознакомьтесь более подробно с теми определениями информации, которые были перечислены на слайде 31. Какой смысл в понятие «информация» вкладывали различные авторы?
5. Найдите и изучите материалы об использовании десятичных и двоичных единиц измерения информации в разных областях компьютерных наук. Что вы можете сказать о двоично-десятичных единицах измерения информации?