

УДК 004.422.613

**А.Я. Анопrienко** (канд. тех. наук, проф.),  
**С.В. Иваница** (аспирант), **Е.И. Котов** (магистрант)  
Донецкий национальный технический университет  
[anoprien@cs.dgtu.donetsk.ua](mailto:anoprien@cs.dgtu.donetsk.ua), [isv@cs.dgtu.donetsk.ua](mailto:isv@cs.dgtu.donetsk.ua), [bimarck555@gmail.com](mailto:bimarck555@gmail.com)

## **ПРЕОБРАЗОВАТЕЛЬ ВЕЩЕСТВЕННЫХ ДЕСЯТИЧНЫХ ЧИСЕЛ В ПОСТБИНАРНЫЕ ФОРМАТЫ С ПЛАВАЮЩЕЙ ЗАПЯТОЙ**

Предложена и описана программа-преобразователь для представления вещественного числа в постбинарных форматах чисел с плавающей запятой. Описаны интерфейс и рабочее пространство программы, рассмотрена ее структурная модель. Выполнена оценка времени работы и показаны примеры преобразования иррационального числа в различных вариантах приближения.

**Ключевые слова:** число с плавающей запятой, постбинарные форматы, постбинарное кодирование, погрешность, округление, длинная арифметика

### ***Введение***

В связи с совершенствованием вычислительных технологий и постоянно повышающимися требованиями к точности вычислений, возникает необходимость оперировать числами повышенной точности. В то же время существующие форматы чисел стандарта IEEE 754–2008 не в полной мере отвечают этим требованиям. IEEE 754 — широко распространённый стандарт формата представления чисел с плавающей точкой, используемый как в программных реализациях арифметических действий, так и во многих аппаратных (CPU и FPU) реализациях [1]. В рамках перехода от бинарного к постбинарному компьютерингу [2] предложены новые форматы данных — постбинарные форматы чисел с плавающей запятой [3], которые призваны повысить точность вычислений и частично устранить недостатки стандарта IEEE 754–2008 [4].

Использование постбинарных форматов как основного носителя количественной информации для постбинарного компьютеринга возможно и в современных бинарных компьютерных системах на программно-аппаратном уровне. Программное внедрение постбинарных форматов достижимо путем создания как специализированных библиотек для языков программирования, ориентированных на императивную модель вычислений, задаваемую фон-неймановской архитектурой ЭВМ, так и в виде специальных библиотек для большинства существующих математических пакетов. На аппаратном уровне использование

постбинарных форматов и вычислений возможно с применением специализированного процессора — постбинарного математического сопроцессора (pFPU — postbinary Floating-Point Unit), оформленного в виде отдельного функционального модуля и позволяющего выполнять постбинарные арифметико-логические операции над постбинарными операндами.

В статье рассмотрена программная модель представления чисел в постбинарных форматах, которая является незаменимым инструментом на этапе разработки постбинарной плавающей арифметики и средством контроля при моделировании компонентов постбинарного сопроцессора. Данная работа является развитием проекта, апробированного на VII международной научно-технической конференции «Информатика и компьютерные технологии», прошедшей в Донецком национальном техническом университете 22–23 ноября 2011 года [5].

### ***К вопросу актуальности создания программы-преобразователя для представления чисел в постбинарных форматах***

Постбинарные форматы чисел с плавающей запятой (pbinary, или pb) впервые были описаны в [9] как модификация существующих форматов стандарта IEEE 754–2008. Фактически, постбинарный формат — это формат с плавающей запятой, поддерживающий постбинарное кодирование и содержащий на месте младших разрядов мантиссы служебную информационную, определяющую точность и тип числа, представленного в данном формате.

Актуальность перехода от бинарных форматов к постбинарным обусловлена, прежде всего, существующими и неустранимыми недостатками современного представления вещественных чисел в форматах с плавающей запятой и выполнением арифметических операций над ними. Ключевыми недостатками бинарных форматов являются как неизбежные округления чисел (при кодировании и во время вычислений), так и отсутствие информации о точности величины, представленной битовым полем формата. Пример того, как эти недочеты приводят к получению неверных результатов, был продемонстрирован З. Румпом, представившим полином (названный по фамилии автора), который при определенном сочетании значений переменных дает заведомо неправильный результат [7, с. 173]. Постбинарные форматы лишены подобных недостатков, поскольку, во-первых, ориентированы на динамическое наращивание разрядности, во-вторых, способны работать с интервальными и дробными значениями, и, в-третьих, использование постбинарного кодирования дает возможность избежать грубых округлений и потерь значимых разрядов мантиссы.

Особенности реализации постбинарных форматов, способы представления чисел этими форматами, оценки погрешностей и полное описание служебных и информационных полей подробно рассмотрены в монографии «Постбинарный компьютеринг и интервальные вычисления в контексте кодо-логической эволюции» [7, с. 196–217].

В частности, в указанной работе введено следующее обозначение форматов: постбинарный формат определен словом «rbinary» или сочетанием «rb», после которого стоит цифра, определяющая длину поля (в битах), занимаемую форматом. Следующая цифра (отделенная от первой слешем «/») определяет разрядность тетракода или составного слова (дроби или интервала), «упакованного» в основном формате. Появление второй цифры требует уточнения, за которое отвечает своеобразный «суффикс» формата:  $r$  — постбинарное число (тетракод);  $f$  — дробь;  $i$  — интервал. Например:

- 1) формат `rbinary128` представляет собой число счетверенной точности (формат представлен 128 битами);
- 2) формат `rb256/64fr` — постбинарный формат, в 256-ти двоичных разрядах которого хранятся два 64-разрядных числа — числитель и знаменатель дроби (на что указывает «f»), при этом каждое 64-разрядное число занимает 128 двоичных разрядов, поскольку является постбинарным (на что указывает «r») каждый разряд которого (тетрит) кодируется парой битов. Следовательно, 2 поля по 64 разряда тетракода занимают  $2 \times (2 \times 64) = 256$  бит.

Идея создания программного продукта, работающего с постбинарными форматами чисел с плавающей запятой, берет свое начало в целом цикле исследований, подтверждающих эффективность перехода к постбинарным форматам [3, 6–8]. В ходе работы над постбинарной арифметикой встали задачи, в которых необходимо представлять вещественные числа в постбинарных форматах в виде набора двоичных полей знака, порядка и мантииссы, и, наоборот, из полей постбинарных форматов получать точное (без округления) представление вещественных чисел в виде десятичной дроби.

Разработанная для этого программа «Преобразователь постбинарных форматов» (Postbinary Format Converter) способна корректно обрабатывать вещественные числа, дроби (определяемые вещественными числителем и знаменателем) и интервалы (определяемые его границами — вещественными числами). Входные числа можно задавать практически любой длины — установленный по умолчанию предел в 1 024 десятичных цифр можно расширять, ограничиваясь лишь объемом свободной оперативной памяти компьютера.

## Интерфейс и рабочее пространство программы Postbinary Format Converter

На рис. 1 представлено рабочее пространство программы Postbinary Format Converter (далее PFC) на котором выделены основные компоненты (к.):

1. **Ввод исходных данных** — поле для ввода десятичного числа, подлежащего преобразованию (поддерживается ввод чисел в нормализованном виде). Возможны два варианта ввода исходных данных в текстовое поле: из клавиатуры; из файла (кнопка «Загрузить...»).

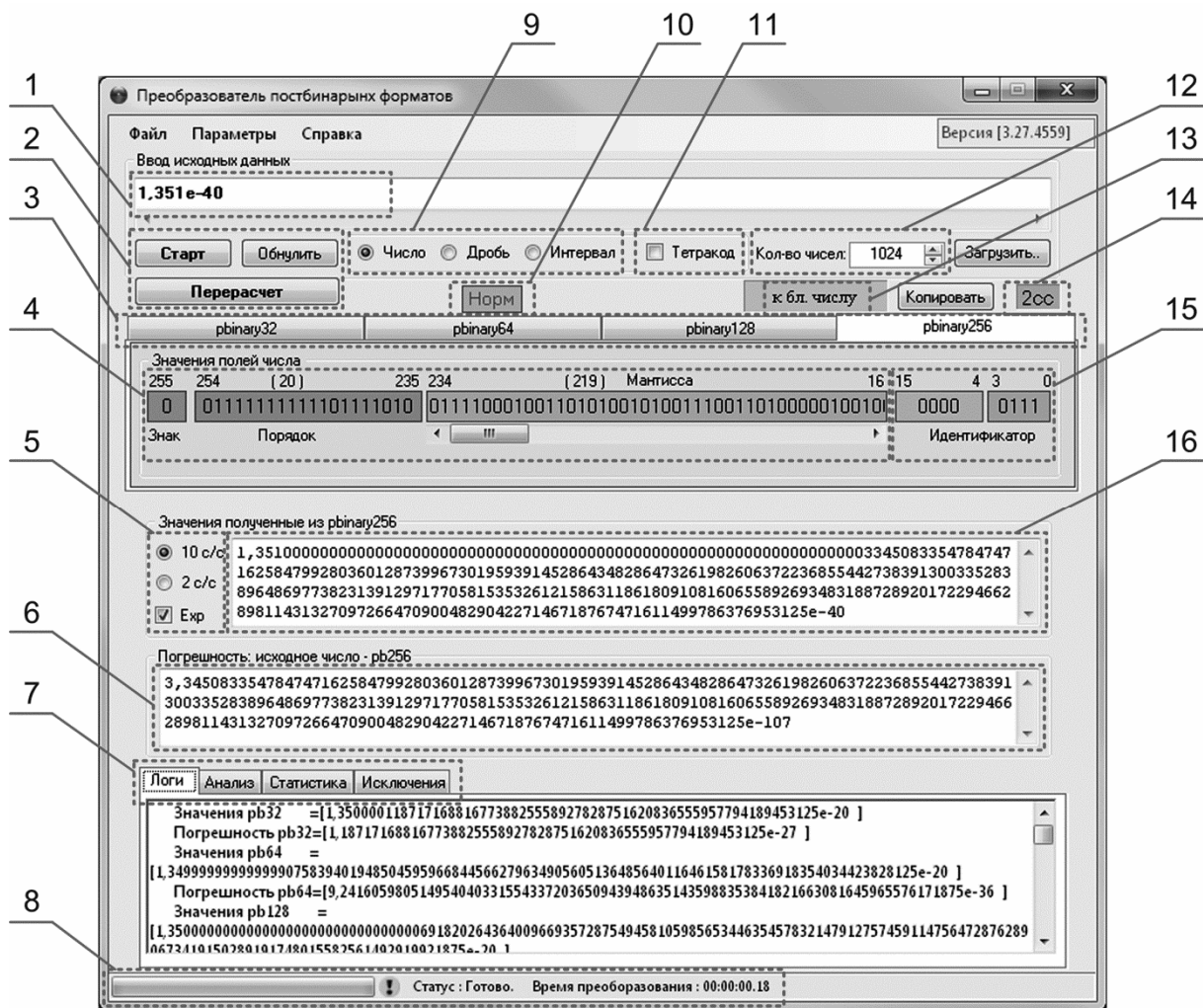


Рисунок 1 — Интерфейс программы PFC в режиме представления десятичного числа  $1,351 \times 10^{-40}$  в 256-разрядном постбинарном формате

2. **Блок управления**, состоящий из трех кнопок: «Старт» — начало преобразования входного числа и извлечение его точного значения из поля формата (формирование результата в направлении *Исходное число* (к.1) →



7. **Служебная информация** о работе программы: «Логи» — ведение истории конвертирования с возможностью ее сохранения в файле; «Статистика» — представление сравнительных характеристик (погрешности и временных затрат) для всех форматов; «Анализ» и «Исключения» — служебная информация для разработчиков (в пользовательской версии программы заменена вкладкой «Сведения о формате» — справочные данные и характеристика текущего формата).

8. **Строка состояния**, отображающая ход выполнения операции (индикатор прогресса), статуса программы, а также времени, затраченного на последнее преобразование.

9. **Переключатель формата входных данных** (табл. 1).

Таблица 1 — Постбинарные форматы в зависимости от выбора типа входных данных и способа кодирования

Число	Дробь	Интервал	Тетракод	Разрядность постбинарного формата			
				32	64	128	256
<input checked="" type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="checkbox"/>	pbinary32	pbinary64	pbinary128	pbinary256
			<input checked="" type="checkbox"/>	pbinary32/16p	pbinary64/32p	pbinary128/64p	pbinary256/128p
<input type="radio"/>	<input checked="" type="radio"/>	<input type="radio"/>	<input type="checkbox"/>	–	pbinary64/32f	pbinary128/64f	pbinary256/128f
			<input checked="" type="checkbox"/>	–	–	pbinary128/32fp	pbinary256/64fp
<input type="radio"/>	<input type="radio"/>	<input checked="" type="radio"/>	<input type="checkbox"/>	–	pbinary64/32i	pbinary128/64i	pbinary256/128i
			<input checked="" type="checkbox"/>	–	–	pbinary128/32ip	pbinary256/64ip

10. **Тип числа в текущем формате**: нормализованное (Норм/Norm), денормализованное (Денорм/Denorm), не число (NaN), бесконечность (Inf). Слово «Переполнение/Overflow» появляется в данном поле, если исходное число невозможно представить в текущем формате.

11. **Компонент для постбинарного кодирования**. Его включение означает, что информационные поля формата (к.4) содержат тетракод (в названии формата появляется суффикс «р») и каждая пара бит поля представляет один четверичный разряд — тетрит.

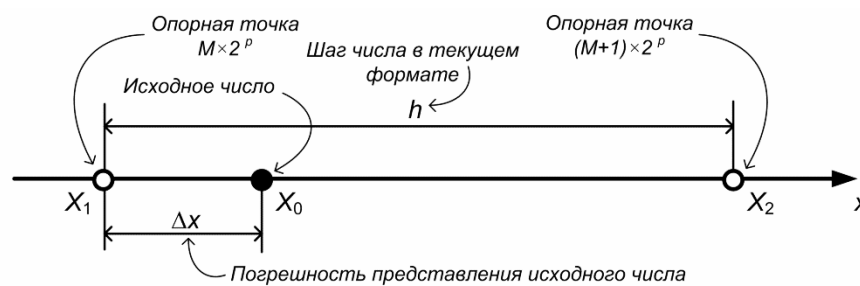
12. **Счетчик ограничения цифр для входного числа** в поле компонента 1. Цифры входного числа, превышающие установленное количество, в дальнейших преобразованиях не учувствуют и просто отбрасываются.

13. **Текущий способ округления**, установленный на момент преобразования. Поддерживаются как стандартные способы округления чисел стандарта IEEE754 (к нулю, к ближайшему числу, к положительной и отрицательной бесконечности), так и специализированный способ (постбинарное округление, рис. 3).

14. **Кнопка-индикатор текущей системы счисления.** Указывает и позволяет выбрать двоичную или 16-ричную систему счисления для отображения полей формата (к.4).

15. **Поле идентификатора текущего формата** — служебная часть постбинарных форматов, состоящая из полей модификатора и кода формата [7, с. 203].

16. **Поле точного значения.** Отображает результат извлечения из полей (к.4) текущего формата (к.3) без потери точности (рис. 2). В данном поле отображается результат преобразования двоичного числа (поля формата к.4) в десятичную дробь без ограничений вычислительных возможностей процессора и округления чисел операционной системой.



Округление к ближайшему числу

$$\text{Число IEEE754} \Rightarrow \begin{cases} X_0 \rightarrow X_1, & \text{если } \Delta x < \frac{1}{2}h \\ X_0 \rightarrow X_2, & \text{если } \Delta x \geq \frac{1}{2}h \end{cases}$$

Постбинарное округление

$$\text{Число в постбинарном формате} \Rightarrow \begin{cases} X_0 \rightarrow X_1, & \text{если } \Delta x < \frac{1}{4}h \\ X_0 \rightarrow [X_1; X_2], & \text{если } \frac{1}{4}h \leq \Delta x < \frac{3}{4}h \\ X_0 \rightarrow X_2, & \text{если } \Delta x \geq \frac{3}{4}h \end{cases}$$

Рисунок 3 — Принцип постбинарного округления для постбинарных форматов чисел с плавающей запятой (опорная точка — точка числовой оси, которая может быть точно представлена текущим плавающим форматом, поэтому множество опорных точек является конечным множеством чисел, представленных в данном формате)

### **Структурная модель и диаграмма деятельности программы Postbinary Format Converter**

На рис. 4 представлена структурная модель преобразователя постбинарных форматов PFC. Ниже приведено краткое описание функциональности блоков модели.

**Интерфейс пользователя (User Interface, UI)** включает в себя набор функций, составляющих каркас программы, а также связывает все остальные блоки с основной формой программы (рис. 1):

- **Програмная оболочка** — набор предоставляемых платформой .NET Framework классов и функций для визуализации данных в форме Windows Forms. Инкапсулирует в себе все необходимые компоненты для работы и манипуляции над данными всех типов.
- **Проверка ошибок** — блок отладки, содержащий набор функций, направленных на самопроверку и поиск ошибок в вычислениях. Запускается при изменении версии программы и не пустом «Файле ошибок». Записывает возникшие в процессе работы названия исключений в «Файл исключений», а числа в результате, которых были сгенерированы эти исключения и текущую версию в «Файл ошибок».
- **Контроль ввода числа** — набор функций, направленных на логическую обработку числа (ввод не больше одного знака «-» и «+» недопустимость ввода букв (кроме экспоненты «e» и «E») и др.) для его корректного представления.

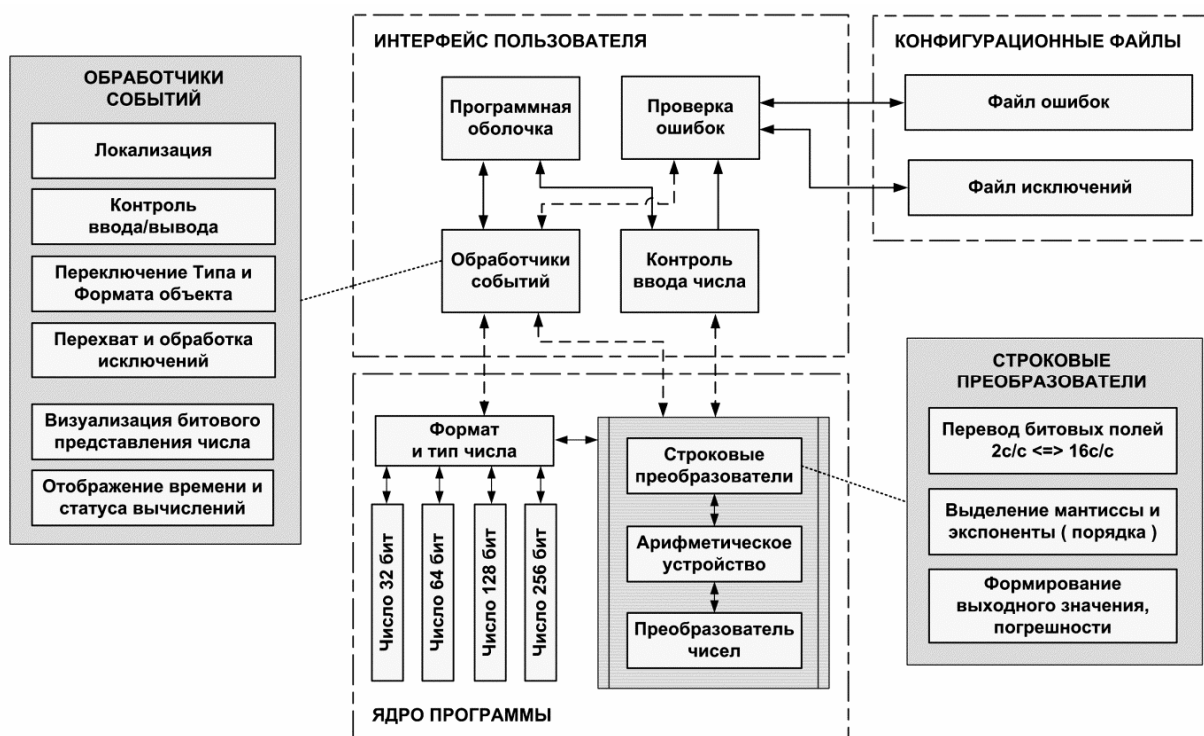


Рисунок 4 — Структурная модель PFC

(сплошная стрелка – передача входных и выходных параметров, пунктирная стрелка – только выходных параметров)

- **Обработчики событий** — набор стандартных «слушателей событий» (event listeners), которые прикреплены к какому-то событию. Генерацию связывания объектов с событием берет на себя среда разработки. В модели представлены следующие обработчики событий:



- *Локалізація* — обробочик, який слідить за подією переключення мови інтерфейсу;
- *Контроль вводу-виводу* направлено на перевірку (валідацію) входного значення перед виконанням перетворення;
- *Переключення типу і формату об'єкта* — обробочики подій, відслідковують налаштування модифікатора формату: тип числа (речовинне, дробне, інтервальне) і спосіб кодування (бінарне, постбінарне);
- *Перехват і обробка виключень* — комплекс перехватчиків виключень, повертає як найменування викликаного виключення, так і викликану його функцію;
- *Візуалізація бітового представлення числа* — обробочик, який викликається після коректного завершення перетворення числа, в результаті чого відбувається заповнення полів на вкладках форматів (к.3, к.4, рис. 1);
- *Відображення часу і статусу обчислень* — обробочик, який служить для розрахунку часу виконання певних частин алгоритму.

**Ядро програми** — комплекс класів і функцій, які є основою всіх обробок, перевірок і перетворень входного числа. Містить математичний апарат, здатний проводити арифметичні операції над числами, представленими в вигляді рядків (для реалізації довгої арифметики [10]). При цьому для реалізації такої арифметики використовується числовий тип `BigInteger` простору імен `System.Numerics` [11], який є складним цілочисельним типом, що підтримує довільне число великих цілих чисел (не має нижньої або верхньої межі і може містити будь-яке цілочисельне значення). Ядро програми включає в себе наступні блоки:

- **Формат і тип числа** — набір змінних, що містять необхідну інформацію про оброблюване число (знак, порядок, мантиса, тип округлення, формат і тип).
- **Числа 32, 64, 128 і 256 біт** — екземпляри спеціального класу (`FSCore.Number`), які зберігають інформацію про представлення числа для кожного формату: статус обробки числа; бітові поля; тип числа; зміщення порядку; витягнуте точне значення; похибка представлення.
- **Рядкові перетворювачі** — набір функцій, які працюють з рядковим представленням числа, перевіряючи і розділяючи його на бітові поля мантиси і порядку, готуючи до арифметичних операцій і виконуючи взаємний переклад з 2-ричної в 16-ричну систему числення.

- **Арифметическое устройство** — блок функций оперирующих строковыми и целочисленными данными. При этом в строках содержатся записи входных и выходных числовых данных блока, а с целочисленными переменными (тип данных System.Numerics.BigInteger) производятся все арифметические операции над целой и дробной частью числа. Эти части обрабатываются по отдельности с последующим сведением результатов как к двоичным полям мантиссы, порядка и знака, так и к точному значению, извлеченному из сформированных полей постбинарного формата.
- **Преобразователь чисел** — группа функций осуществляющих преобразование десятичного вещественного числа в двоичную дробь, а также приведение числа (двоичного и десятичного) к нормализованному виду.

На рис. 5 приведена диаграмма деятельности (activity diagram) в нотации языка UML [12], призванная детализировать особенности алгоритмической и логической реализации выполняемых программой операций.

### ***Оценка времени работы программы Postbinary Format Converter***

На рис. 6 приведена гистограмма общего времени обработки программой PFC значений из диапазона  $1,5E \mp 200$  (с логарифмическим шагом) и преобразования их в постбинарные форматы pb32 ÷ pb256. Нулевые значения времени для формата rbinary32 показывают, что исходное число не вошло в область представления этого формата, и дальнейшая обработка не производилась. Общее время для каждого формата состоит из суммы затраченного времени на формирование битовых полей порядка, мантиссы и знака, извлечение точного значения из этих полей и расчета погрешности представления исходного числа. Измерения выполнены с точностью до миллисекунды.

Для нахождения причинно-следственной связи между входными данными и временем их обработки, были получены коэффициенты логарифмической и степенной аппроксимирующих функций (кривые на рис. 5), численный вид которых представлен в табл. 2. Логарифмическая аппроксимирующая функция применена только для формата rbinary32 из-за наличия нулевых значений аргумента, при которых невозможно применить степенную функцию. В табл. 2 также приведены значения величины  $R^2$ , которые характеризуют достоверность аппроксимации: чем ближе  $R^2$  к единице, тем полученная функция точнее аппроксимирует исследуемый процесс.

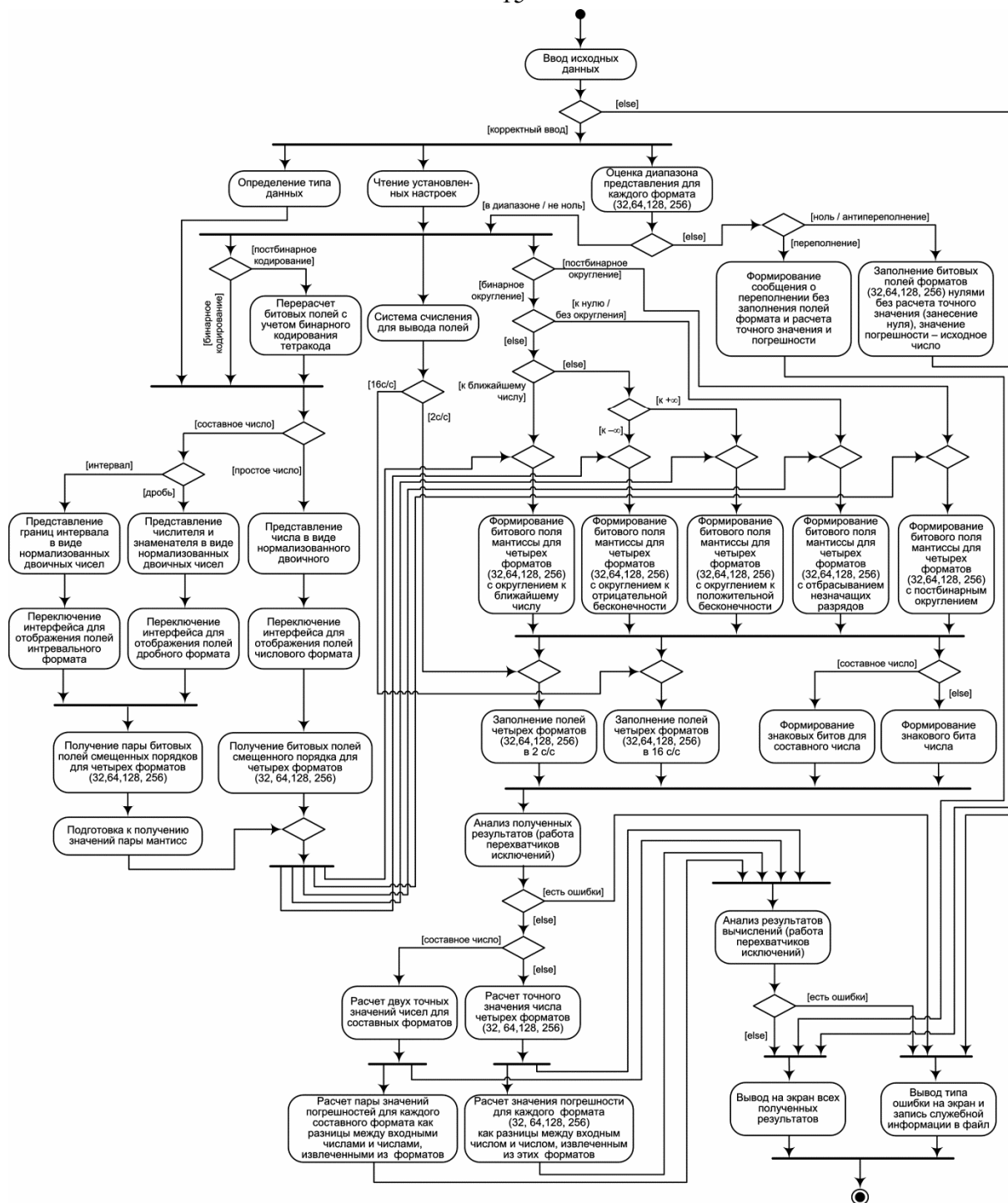


Рисунок 5 — Диаграмма деятельности программы PFC

На основании полученных результатов можно сделать ряд заключений:

1) Временные затраты существенно увеличиваются при обработке чисел с возрастающей отрицательной степенью десятичной экспоненты. Иными словами, чем исходное число «ближе к нулю», тем больше времени требуется для его обработки. Это связано с операциями над числами, представляющими большие отрицательные степени двойки. При этом

большую часть времени занимает вычисление точного значения этих чисел.

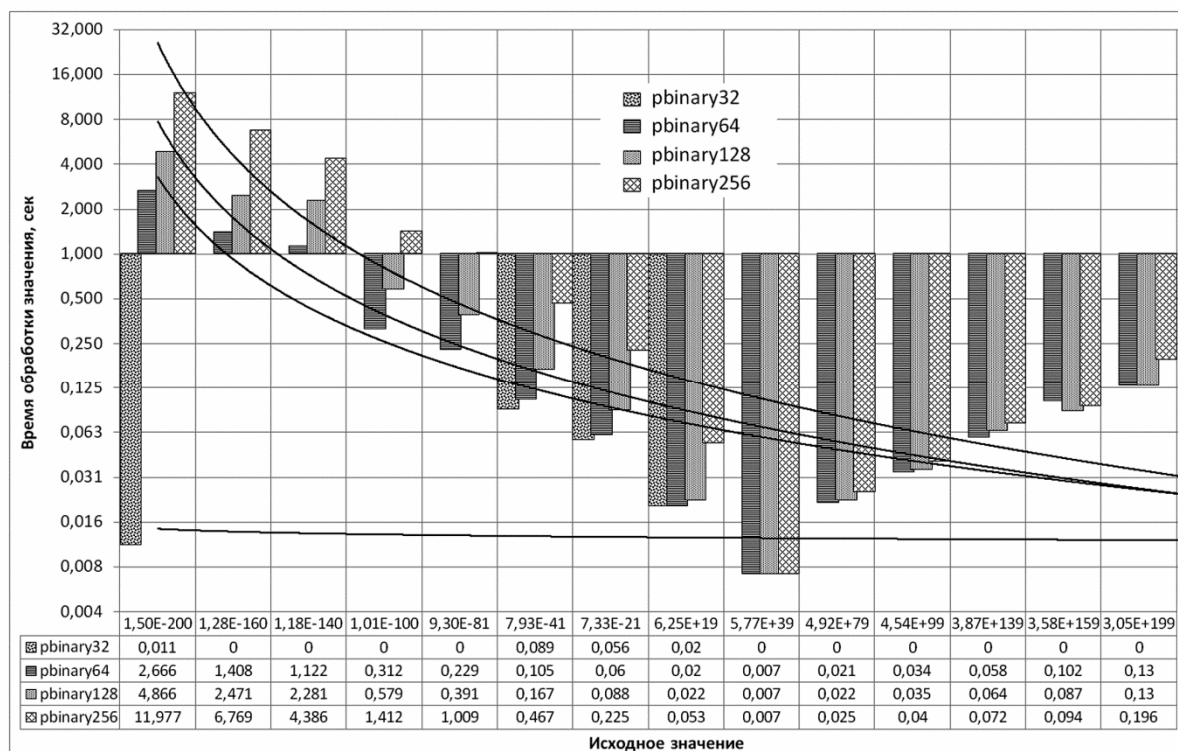


Рисунок 6 — Гистограмма временных затрат на обработку данных для постбинарных форматов и графики аппроксимирующих функций

Таблица 2 — Аппроксимирующие функции и величина достоверности аппроксимации  $R^2$  для исследуемых форматов

Формат	Аппроксимация	Аппроксимирующая функция $f(x)$	$R^2$
pbinary32	Логарифмическая	$-0,0009 \ln(x) + 0,0141$	0,206
pbinary64	Степенная	$3,2991 x^{-1,887}$	0,7742
pbinary128	Степенная	$7,8003 x^{-2,157}$	0,821
pbinary256	Степенная	$26,207 x^{-2,508}$	0,8476

Например, при исходном числе  $1,5e-200$  для получения точного значения, извлеченного из полей форматов, производится преобразование следующих двоичных дробей

– для pbinary64:

1,001001011110111011101100011111111011001110011100e-664;

– для pbinary128:

1,00100101111011101110110001111111101100111001110000010001011110110001111001001001111110100110011000011010e-664;

– для pbinary256:

1,0010010111101110111011000111111110110011100111000001000101111011000111100100100111111010110111001101011111111010101101101111110010011011011010001010110010001000110111100100111e-664

### В соответствии десятичные дроби

— для `pbinary64`:

```
1,499999999999999968306684161266835577666944638341229562731698860453551871  
7844272789199436274191888833431534392330820717096774434630570658306838242  
9364958665499574673400370351047036375195939993158407568317077135113315489  
9405913582310844055191206758665801192845812540228026407155213222260120937  
2805671713096801257681381538068544002032645617134735324147799751279359770  
3292517866507460278193283416442987674273509551655239607672702679192194789  
1777325038451909839856799460126781514679450424409878905862569808959960937  
5e-200;
```

— для `pbinary128`:

```
1,50000000000000000000000000000002973692538587960351298165733639388618084  
4159622543222587667348491274409135991825637900579782014114689514001626100  
1143506979310342989473996627064111397337154954957172014452948033230007408  
9490040497641138505891107781915339730730122004187267471296687505786255426  
8072845031235122527745649807629541176195615183891700838589512934935300732  
0625859618749202185623287739900289560778943568321912184166307694524843418  
9615533733210307997605675491153811312261043637454539408031964865867053505  
4051412306221091087776731143321740091778337955474853515625e-200;
```

— для `pbinary256`:

```
1,499999999999999999999999999999999999999999999999999999999999999999982196  
7047652181113934752028303675459856871231994234596814474668480669488635112  
8204388895402354966486996532712730302031359621343676043630585708062069058  
0672100607213236194344171866467422297439560461273736196341507418363558985  
5823824782492989251610675835595136078541253255811501681316340443103364912  
9137409609661868933161985846499153382903671732114442091153684476770409963  
2875001045926275065877616845818337228719704425692510666990330049456470295  
6012758733006658953658980815049654657346855052917417383997508337133480568  
3947205272444329516574199206532545407733667786819627656860320213949577805  
5788832716643810272216796875e-200.
```

Данный пример демонстрирует вычислительные возможности программы. Причем для получения значения погрешности представления чисел организована операция вычитания по модулю между рассчитанным «длинным» и исходным числами для каждого формата.

2) На основании предыдущего пункта: денормализованные числа требуют для обработки и преобразования на порядок больше времени, чем нормализованные числа.

3) Моменты времени  $t$ , необходимые для полного цикла преобразований одного входного числа в форматах различной точности имеют экспоненциальную зависимость, пропорциональную разрядности формата. То есть, чем точнее формат (больше разрядность), тем больше времени необходимо для работы с ним. Очевидно, что

$$t(\text{pbinary}32) < t(\text{pbinary}64) < t(\text{pbinary}128) < t(\text{pbinary}256).$$

### **Пример представления числа в постбиннарных форматах с плавающей запятой**

Рассмотрим варианты преобразования числа на примере числового, дробного и интервального форматов с получением погрешности представления. В качестве исходного числа рассмотрим математическую константу  $e$  — иррациональное и трансцендентное число, основание

натурального логарифма, — играющее важную роль в дифференциальном и интегральном исчислении, а также во многих других разделах математики. Иррациональность данного числа указывает, что его невозможно точно представить в виде десятичной и рациональной дробей. Поэтому воспользуемся десятичным и рациональным приближениями с погрешностью, например, не более  $10^{-9}$ . Получаем значения входных данных в виде

– числа  $e \approx 2,718281828$ ;

– рациональной дроби  $e \approx \frac{271801}{99990}$ ;

– интервала  $e \in [2,718281828; 2,718281829]$ .

На рис. 7 приведены битовые значения полей соответствующих типов на примере 128-разрядных форматов (для данного примера выбраны типы форматов, использующих бинарное кодирование), а также значения, извлеченные из данных полей. В табл. 3 сведены погрешности представления исходных данных для каждого формата.

На основании полученных значений из полей форматов и величины погрешности можно сделать следующие заключения:

1) При представлении иррационального числа в числовом формате, погрешности исходного числа и его представления в виде числа с плавающей запятой накладываются, что может привести к еще большей потере точности (в нашем случае в формате rb32 погрешность исходного числа снизилась с  $10^{-9}$  до  $10^{-7}$ );

2) Целые числа числителя и знаменателя рациональной дроби представляются без погрешности (с учетом, что данные числа входят в диапазон представления формата). При этом сохраняется только исходная погрешность числа — погрешность рационального приближения.

3) Интервальное представление исходного числа имеет преимущество, поскольку заданный интервал содержит исходное число не зависимо от класса точности последнего. Однако, при задании границ интервала, следует учитывать относительную точность десятичных цифр, которую обеспечивают форматы с плавающей запятой. Эта точность определяется по формуле  $(\lg 2^{m+1}) \in \mathbf{Z}$ , где  $m$  — количество разрядов мантииссы. В нашем случае для 64-разрядного интервального формата количество разрядов мантииссы для каждой границы равно 21, следовательно, относительная точность равна  $\lg 2^{22} = 7$  десятичных цифр. Поэтому числа 2,718281828 и 2,718281829 (отличающиеся друг от друга 10-ой десятичной цифрой) для данного формата неразличимы (о чем свидетельствует одинаковое значение погрешности для каждой границы в табл. 3). В таких случаях необходим переход к формату большей разрядности (т. е. более высокого класса точности).

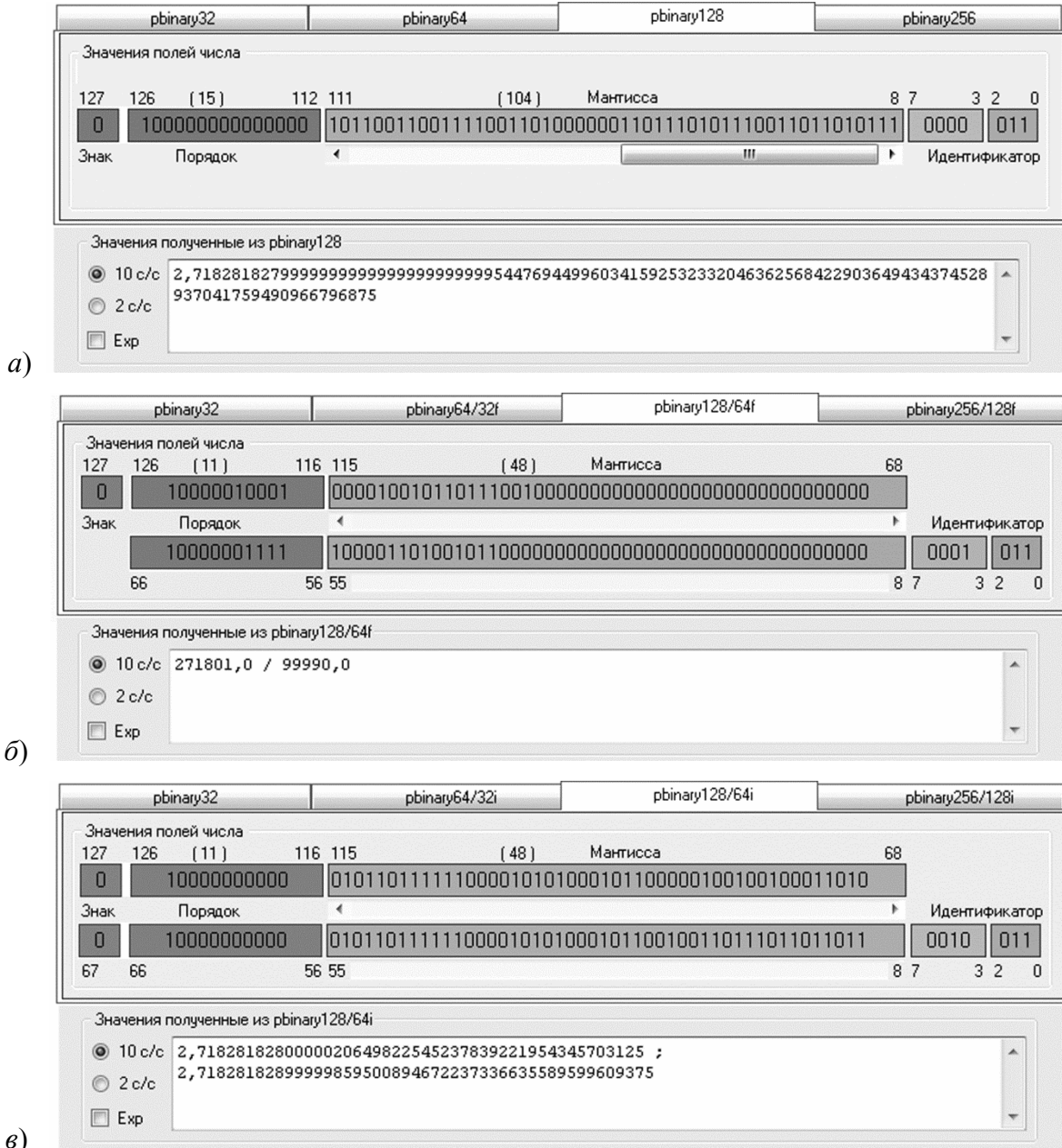


Рисунок 7 — Представление константы  $e$  в 128-разрядных постбинарных форматах в виде десятичной (а) и рациональной (б) дробей, а также в виде вещественного интервала (в)

Таблица 3 — Погрешность представления константы  $e$  в числовых дробных и интервальных форматах

Разрядность	Число	Дробь	Интервал
32	8,208935547e <sup>-7</sup>	*	—
64	2,064982255e <sup>-14</sup>	0,0 / 0,0	8,208935547e <sup>-7</sup> ; 8,308935547e <sup>-7</sup>
128	4,552305500e <sup>-31</sup>		2,064982255e <sup>-14</sup> ; 1,4049910533e <sup>-14</sup>
256	2,375392996e <sup>-66</sup>		4,552305500e <sup>-31</sup> ; 8,608605556e <sup>-32</sup>

\* — работа с дробями и интервалами не предусмотрена (см. табл. 1)

## **Выводы**

В данной работе рассмотрена программа-преобразователь данных в постбинарные форматы с плавающей запятой. Этот программный продукт является не имеющим аналогов инструментом при исследовании и разработке арифметики постбинарных форматов. В процессе отладки основных этапов алгоритмов выполнения арифметических операций над числами, представленными в постбинарных форматах (например, нормализация результата, выравнивание порядков, фиксирование переполнения, и др.) приходится многократно осуществлять взаимные переходы от вещественных десятичных чисел к двоичным полям мантиссы, порядка и знака постбинарного формата. Программа Postbinary Format Converter позволяет значительно снизить трудозатраты, связанные с преобразованием операндов и оценить погрешность их представления в постбинарных форматах чисел с плавающей запятой. Кроме того, программа снабжена обширной справочной информацией о представленных форматах: характеристикой, значениями минимальных и максимальных числовых диапазонов, относительной точности десятичных цифр и значениями исключительных чисел.

Представленный программный продукт может быть также полезен студентам, магистрантам и аспирантам компьютерных специальностей для ознакомления с основами постбинарного компьютеринга, а также для расширения научного кругозора при работе со стандартными форматами чисел с плавающей запятой.

## **Список литературы**

1. IEEE floating point — Wikipedia, the free encyclopedia. [Электронный ресурс] — Режим доступа: [http://en.wikipedia.org/wiki/IEEE\\_floating\\_point](http://en.wikipedia.org/wiki/IEEE_floating_point).
2. Аноприенко А.Я. Вызовы времени и постбинарный компьютеринг // Информатика и компьютерные технологии / Материалы VI международной научно-технической конференции — 23–25 ноября 2010 г. Т. 1. Донецк, ДонНТУ. — 2010. С. 13–31.
3. Аноприенко А.Я., Иваница С.В. Гибкая разрядность и постбинарные форматы представления вещественных чисел // Вестник Инженерной Академии Украины. Теоретический и научно-практический журнал Инженерной Академии Украины. Выпуск 1. — Киев, 2012. С. 92–98.
4. Яшкардин В.Л. IEEE 754 — стандарт двоичной арифметики с плавающей точкой. [Электронный ресурс] — Режим доступа: <http://softelectro.ru/ieee754.html>.
5. Кулибаба С.В., Иваница С.В. Программная модель преобразователя вещественных чисел в постбинарные форматы // Информатика и компьютерные технологии / Материалы VII международной научно-технической конференции студентов, аспирантов и молодых ученых — 22–23 ноября 2011 г. Т.1. Донецк, ДонНТУ. — 2011. С. 214–219.
6. Аноприенко А.Я. Особенности представления постбинарных вещественных чисел в контексте интервальных вычислений и развития аппаратного обеспечения



средств компьютерного моделирования. / А.Я. Аноприенко, С.В. Иваница, С.В. Кулибаба // Материалы четвертой международной научно-технической конференции «Моделирование и компьютерная графика» 5–8 октября 2011 года, Донецк, ДонНТУ, 2011. С. 13–19.

7. Аноприенко А.Я. Постбинарный компьютинг и интервальные вычисления в контексте кодо-логической эволюции. / А.Я. Аноприенко, С.В. Иваница — Донецк, ДонНТУ, УНИТЕХ, 2011. — 248 с.

8. Аноприенко А.Я. Представление постбинарных форматов чисел с плавающей запятой в контексте интервальных вычислений. / А.Я. Аноприенко, С.В. Иваница, С.В. Кулибаба // Наукові праці Донецького національного технічного університету, серія «Інформатика, кібернетика та обчислювальна техніка», вып. 14 (188). — Донецк, ДонНТУ, 2011. — С. 55–60.

9. Аноприенко А.Я., Гранковский В.А., Иваница С.В. Пример Румпа в контексте традиционных, интервальных и постбинарных вычислений / А.Я. Аноприенко, В.А. Гранковский, С.В. Иваница // Научные труды Донецкого национального технического университета. Серия «Проблемы моделирования и автоматизации проектирования динамических систем» (МАП-2011). Выпуск 9 (179): Донецк: ДонНТУ, 2011. С. 324–343.

10. Окулов С.М. Длинная арифметика. [Электронный ресурс] — Режим доступа: <http://comp-science.narod.ru/DL-AR/okulov.htm>.

11. System.Numerics Namespace. MSDN Library. .NET Framework 4. [Электронный ресурс] — Режим доступа: <http://msdn.microsoft.com/en-us/library/system.numerics.aspx>.

12. Леоненков А.В. Самоучитель по UML. [Электронный ресурс] — Режим доступа: <http://khpi-iip.mipk.kharkiv.edu/library/case/leon/index.html>.

*Надійшла до редакції 02.09.2012 г.*

*Рецензент: канд.тех.наук, доц. Зеленева И.Я.*

**А.Я. Аноприенко, С.В. Иваница, Є.І. Котов**  
Донецький національний технічний університет

**Перетворювач дійсних десяткових чисел у постбінарні формати з рухомою комою.** Запропонована та описана програма-перетворювач для подання дійсного числа в постбінарних форматах чисел з рухомою комою. Описано інтерфейс і робочий простір програми, розглянуто її структурна модель. Виконана оцінка часу роботи і показані приклади перетворення ірраціонального числа в різних варіантах наближення.

**Ключові слова:** число з плаваючою комою, постбінарні формати, постбінарне кодування, похибка, округлення, довга арифметика.

**A.J. Anopriyenko, S.V. Ivaniza, E.I. Kotov**  
Donetsk National Technical University

**Decimal number converter in postbinary floating-point formats.** We proposed and described a program converter for presentation of a real number in postbinary format floating-point numbers. We described the interface and the workspace of the program, described its structural model. We also measured calculation time and showed examples of the transformation of the irrational numbers in a variety of approximations.

**Keywords:** floating-point number, postbinary formats, postbinary coding, error of rounding, long arithmetic.