Введение

Производительность является самой важной характеристикой компьютера. Производительность – это способность компьютера выполнять определенные задачи за определенные промежутки времени.

Компьютер, выполняющий тот же объем работы за меньшее время является более быстрым. Время выполнения любой программы измеряется в секундах. Часто производительность измеряется как скорость появления некоторого числа событий в секунду, так что меньшее время подразумевает большую производительность.

Однако в зависимости от того, что мы считаем, время может быть определено различными способами. Наиболее простой способ определения времени называется астрономическим временем, временем ответа (response time), временем выполнения (execution time) или прошедшим временем (elapsed time). Это задержка выполнения задания, включающая буквально все: работу процессора, обращения к диску, обращения к памяти, ввод/вывод и накладные расходы операционной системы. Однако при работе в мультипрограммном режиме во время ожидания ввода/вывода для одной программы, процессор может выполнять другую программу, и система не обязательно будет минимизировать время выполнения данной конкретной программы.

Для измерения времени работы процессора на данной программе используется специальный параметр - время ЦП (CPU time), которое не включает время ожидания ввода/вывода или время выполнения другой программы. Очевидно, что время ответа, видимое пользователем, является полным временем выполнения программы, а не временем ЦП. Время ЦП может далее делиться на время, потраченное ЦП непосредственно на выполнение программы пользователя и называемое пользовательским временем ЦП, и время ЦП, затраченное операционной системой на выполнение заданий, затребованных программой, и называемое системным временем ЦП.

В ряде случаев системное время ЦП игнорируется из-за возможной неточности измерений, выполняемых самой операционной системой, а также из-за проблем, связанных со сравнением производительности машин с разными операционными системами. С другой стороны, системный код на некоторых машинах является пользовательским кодом на других и, кроме того, практически никакая программа не может работать без некоторой операционной системы. Поэтому при измерениях производительности процессора часто используется сумма пользовательского и системного времени ЦП.

В большинстве современных процессоров скорость протекания процессов взаимодействия внутренних функциональных устройств определяется не естественными задержками в этих устройствах, а задается единой системой синхросигналов, вырабатываемых некоторым генератором тактовых импульсов, как правило, работающим с постоянной скоростью. Дискретные временные события называются тактами синхронизации (clock ticks), просто тактами (ticks), периодами синхронизации (clock periods), циклами (cycles) или циклами синхронизации (clock cycles). Разработчики компьютеров обычно говорят о периоде синхронизации, который определяется либо своей длительностью (например, 10 наносекунд), либо частотой (например, 100 МГц). Длительность периода синхронизации есть величина, обратная к частоте синхронизации.

Таким образом, время ЦП для некоторой программы может быть выражено двумя способами: количеством тактов синхронизации для данной программы, умноженным на длительность такта синхронизации, либо количеством тактов синхронизации для данной программы, деленным на частоту синхронизации.

Важной характеристикой, часто публикуемой в отчетах по процессорам, является среднее количество тактов синхронизации на одну команду - CPI (clock cycles per instruction). При известном количестве выполняемых команд в

программе этот параметр позволяет быстро оценить время ЦП для данной программы.

Таким образом, производительность ЦП зависит от трех параметров: такта (или частоты) синхронизации, среднего количества тактов на команду и количества выполняемых команд. Невозможно изменить ни один из указанных параметров изолированно от другого, поскольку базовые технологии, используемые для изменения каждого из этих параметров, взаимосвязаны: частота синхронизации определяется технологией аппаратных средств и функциональной организацией процессора; среднее количество тактов на команду зависит от функциональной организации и архитектуры системы команд; а количество выполняемых в программе команд определяется архитектурой системы команд и технологией компиляторов. Когда сравниваются две машины, необходимо рассматривать все три компоненты, чтобы понять относительную производительность.

В процессе поиска стандартной единицы измерения производительности компьютеров, было принято несколько популярных единиц измерения. Они подробно рассмотрены в первой главе.

1. Обзор методов и средств оценки производительности вычислительных систем. Постановка задачи.

1.1 Показатели оценки производительности вычислительных систем

MIPS

Одной из альтернативных единиц измерения производительности процессора (по отношению к времени выполнения) является MIPS - (миллион команд в секунду). Имеется несколько различных вариантов интерпретации определения MIPS.

В общем случае MIPS есть скорость операций в единицу времени, т.е. для любой данной программы MIPS есть просто отношение количества команд в программе к времени ее выполнения. Таким образом, производительность может быть определена как обратная к времени выполнения величина, причем более быстрые машины при этом будут иметь более высокий рейтинг MIPS.

Положительными сторонами MIPS является то, что эту характеристику легко понять, особенно покупателю, и что более быстрая машина характеризуется большим числом MIPS, что соответствует нашим интуитивным представлениям. Однако использование MIPS в качестве метрики для сравнения наталкивается на три проблемы. Во-первых, MIPS зависит от набора команд процессора, что затрудняет сравнение по MIPS компьютеров, имеющих разные системы команд. Во-вторых, MIPS даже на одном и том же компьютере меняется от программы к программе. В-третьих, MIPS может меняться по отношению к производительности в противоположенную сторону.

Классическим примером для последнего случая является рейтинг MIPS для машины, в состав которой входит сопроцессор плавающей точки. Поскольку в общем случае на каждую команду с плавающей точкой требуется большее количество тактов синхронизации, чем на целочисленную команду, то программы, используя сопроцессор плавающей точки вместо соответствующих подпрограмм из состава программного обеспечения, выполняются за меньшее время, но имеют меньший рейтинг MIPS. При отсутствии сопроцессора операции над числами с плавающей точкой реализуются с помощью подпрограмм, использующих более простые команды целочисленной арифметики и, как следствие, такие машины имеют более высокий рейтинг MIPS, но выполняют настолько большее количество команд, что общее время выполнения значительно увеличивается. Подобные аномалии наблюдаются и при использовании оптимизирующих компиляторов, когда в результате оптимизации сокращается количество выполняемых в программе команд, рейтинг MIPS уменьшается, а производительность увеличивается [4].

Другое определение MIPS связано с очень популярным когда-то компьютером VAX 11/780 компании DEC. Именно этот компьютер был принят в качестве эталона для сравнения производительности различных машин. Считалось, что производительность VAX 11/780 равна 1MIPS (одному миллиону команд в секунду).

В то время широкое распространение получил синтетический тест Dhrystone, который позволял оценивать эффективность процессоров и компиляторов с языка С для программ нечисловой обработки. Он представлял собой тестовую смесь, 53% которой составляли операторы присваивания, 32% - операторы управления и 15% - вызовы функций. Это был очень короткий тест: общее число команд равнялось 100. Скорость выполнения программы из этих 100 команд измерялась в Dhrystone в секунду. Быстродействие VAX 11/780 на этом синтетическом тесте составляло 1757 Dhrystone в секунду. Таким образом 1 MIPS равен 1757 Dhrystone в секунду [4].

Третье определение MIPS связано с IBM RS/6000 MIPS. Дело в том, что ряд производителей и пользователей (последователей фирмы IBM) предпочитают сравнивать производительность своих компьютеров с производительностью современных компьютеров IBM, а не со старой машиной компании DEC. Соотношение между VAX MIPS и RS/6000 MIPS никогда широко не публиковались, но 1 RS/6000 MIPS примерно равен 1.6 VAX 11/780 MIPS [4].

MFLOPS

Измерение производительности компьютеров при решении научно-технических задач, в которых существенно используется арифметика с плавающей точкой, всегда вызывало особый интерес. Именно для таких вычислений впервые встал вопрос об измерении производительности, а по достигнутым показателям часто делались выводы об общем уровне разработок компьютеров. Обычно для научно-технических задач производительность процессора оценивается в MFLOPS (миллионах чисел-результатов вычислений с плавающей точкой в секунду, или миллионах элементарных арифметических операций над числами с плавающей точкой, выполненных в секунду).

Как единица измерения, MFLOPS, предназначена для оценки производительности только операций с плавающей точкой, и поэтому не применима вне этой ограниченной области. Например, программы компиляторов имеют рейтинг MFLOPS близкий к нулю вне зависимости от того, насколько быстра машина, поскольку компиляторы редко используют арифметику с плавающей точкой.

MFLOPS базируется на количестве выполняемых операций, а не на количестве выполняемых команд. По мнению многих программистов, одна и та же программа, работающая на различных компьютерах, будет выполнять различное количество команд, но одно и то же количество операций с плавающей точкой. Именно поэтому рейтинг MFLOPS предназначался для справедливого сравнения различных машин между собой [4].

1.2. Стандартные тесты измерения производительности

В этом параграфе рассматриваются наиболее часто встречающиеся стандартные контрольно-оценочные тесты - Whetstone, Dhrystone, Linpack [1]. Linpack, и Whetstone характеризуют обработку вещественных чисел, а Dhrystone - целочисленную обработку. Более современные тесты SPEC являются, по сути, множеством пакетов для получения комплекса оценок производительности пакетной обработки и потому описаны отдельно.

1.2.1. Whetstone (общее описание)

В 1976 году Х.Курноу (H.J.Curnow) и Б.Вичманн (B.A.Wichmann) из Британской национальной физической лаборатории (National Physical Laboratory) представили комплект программ для измерения производительности, написанных на языке Алгол-60. Это был первый случай публикации контрольно-оценочных тестов, тем более примечательный, что пакет Whetstone составлен из синтетических тестов, разработанных с использованием статистики распределения инструкций промежуточного уровня (Whetstone-инструкций) компилятора Whetstone Algol (от которого и пошло название этого измерительного пакета), собранной на основе большого количества вычислительных задач. Более подробное описание данного тестового пакета приведено в следующей главе.

1.2.2. Dhrystone (общее описание)

Тесты Dhrystone основаны на типовом распределении языковых конструкций. В состав Dhrystone включено 12 модулей, представляющих различные типовые режимы обработки. Тесты Dhrystone предназначены для

оценки производительности относящейся к функционированию конкретных видов системного и прикладного ПО (операционные системы, компиляторы, редакторы и т.д.). Более подробное описание данного тестового пакета приведено в следующей главе.

1.2.3. Linpack (общее описание)

Linpack представляет собой набор функций линейной алгебры. Программы пакета выполняют обработку двумерных матриц, размер которых является основным параметром тестирования (чаще всего применяются матрицы 100x100 или 1000x1000): чем больше элементов в матрице, тем выше параллелизм операций при тестировании производительности. Особую значимость этот параметр имеет для компьютеров с векторной архитектурой (в этом случае он характеризует длину обрабатываемых векторов), однако было бы большой ошибкой не считаться с ним при тестировании систем других классов. Дело в том, что практически все современные компьютеры широко используют средства параллельной обработки (конвейеризованная и/или суперскалярная арифметика, VLIW-архитектура процессора, MPP-организация системы и т. д.), поэтому оценка производительности при разной глубине программного параллелизма весьма показательна для любой современной системы.

Интерпретация результатов:

- Оценка **Linpack** характеризует главным образом производительность обработки чисел с плавающей точкой; при задании большого размера матриц (1000x1000) влияние целочисленных операций и команд управления (операторы типа IF) на эту оценку невелико.
- Наибольший вес в результирующей оценке производительности имеет оператор вида Y[i]=Y[i]+a*X[i], представленный тестовой процедурой SAXPY/DAXPY (SAXPY - одиночная точность; DAXPY - двойная точность). По данным Вейкера, эта процедура занимает свыше 75% времени выполнения всех тестов Linpack, а Донгарра приводит еще большее

значение - 90%. Следовательно, Linpack-оценка показательна не для всего множества операций с плавающей точкой, а в основном для команд сложения и умножения.

- Малый объем исполнительного кода Linpack (примерно 4,5 Кбайт) и незначительное число операций перехода практически не создают сколько-нибудь значимой нагрузки на средства буферизации команд в процессоре: большая часть модулей пакета целиком размещается в кэше инструкций и не требует динамической подкачки команд в процессе выполнения (например, самая “весомая” процедура SAXPY/DAXPY представлена всего 234 байтами кода). Однако нагрузка на тракт взаимодействия процессора с памятью достаточно высока: тесты одинарной точности с матрицами 100x100 обрабатывают 40 Кбайт данных, а двойной точности - 80 Кбайт. Конечно, для большинства современных компьютеров весь объем данных Linpack, скорее всего, будет локализован во вторичной кэш-памяти, и тем не менее результаты выполнения теста, особенно для матриц размера 1000x1000, в большей степени соответствуют понятию “системная производительность пакетной обработки”, чем оценки, полученные с помощью Whetstone и Dhrystone и отражающие преимущественно производительность процессора.
- Отсутствие в тестах Linpack обращений к библиотечным функциям исключает возможность оптимизации результатов поставщиками компьютеров и позволяет трактовать полученные оценки почти как “чистую” характеристику производительности системы.
- Методика Linpack требует обязательной публикации названия компилятора, выполнявшего трансляцию исходного кода (во время этой процедуры запрещается любое ручное вмешательство в действия компилятора, не разрешается даже убирать комментарии из текста программ), и операционной системы, под управлением которой производилось тестирование. Отсутствие этих данных, а также сведений об установленных атрибутах тестирования (Single/Double, Rolled/Unrolled, Coded BLAS/Fortran BLAS) и размере матриц должно служить предупреждением о возможном

нарушении стандартных условий измерения производительности по методике Linpack**.**

1.3 Основные тестовые пакеты корпорации SPEC

1.3.1. SPECint92, SPECfp92

Важность создания пакетов тестов, базирующихся на реальных прикладных программах широкого круга пользователей и обеспечивающих эффективную оценку производительности процессоров, была осознана большинством крупнейших производителей компьютерного оборудования, которые в 1988 году учредили бесприбыльную корпорацию SPEC (Standard Performance Evaluation Corporation) [5]. Основной целью этой организации является разработка и поддержка стандартизованного набора специально подобранных тестовых программ для оценки производительности новейших поколений высокопроизводительных компьютеров. Членом SPEC может стать любая организация, уплатившая вступительный взнос.

Главными видами деятельности SPEC являются: разработка и публикация наборов тестов, предназначенных для измерения производительности компьютеров. Перед публикацией объектные коды этих наборов вместе с исходными текстами и инструментальными средствами интенсивно проверяются на предмет возможности импортирования на разные платформы. Они доступны для широкого круга пользователей за плату, покрывающую расходы на разработку и административные издержки. Специальное лицензионное соглашение регулирует вопросы выполнения тестирования и публикации результатов в соответствии с документацией на каждый тестовый набор. SPEC публикует ежеквартальный отчет о новостях SPEC и результатах тестирования: "The SPEC Newsletter", что обеспечивает централизованный источник информации для результатов тестирования на тестах SPEC.

Основным результатом работы SPEC являются наборы тестов. Эти наборы разрабатываются SPEC с использованием кодов, поступающих из разных источников. SPEC работает над импортированием этих кодов на разные платформы, а также создает инструментальные средства для формирования из кодов, выбранных в качестве тестов, осмысленных рабочих нагрузок. Поэтому тесты SPEC отличаются от свободно распространяемых программ. Хотя они могут существовать под похожими или теми же самыми именами, время их выполнения в общем случае будет отличаться.

В настоящее время имеется два базовых набора тестов SPEC, ориентированных на интенсивные расчеты и измеряющих производительность процессора, системы памяти, а также эффективность генерации кода компилятором. Как правило, эти тесты ориентированы на операционную систему UNIX, но они также импортированы и на другие платформы. Процент времени, расходуемого на работу операционной системы и функции ввода/вывода, в общем случае ничтожно мал.

Набор тестов CINT92, измеряющий производительность процессора при обработке целых чисел, состоит из шести программ, написанных на языке Си и выбранных из различных прикладных областей: теория цепей, интерпретатор языка Лисп, разработка логических схем, упаковка текстовых файлов, электронные таблицы и компиляция программ.

Набор тестов CFP92, измеряющий производительность процессора при обработке чисел с плавающей точкой, состоит из 14 программ, также выбранных из различных прикладных областей: разработка аналоговых схем, моделирование методом Монте-Карло, квантовая химия, оптика, робототехника, квантовая физика, астрофизика, прогноз погоды и другие научные и инженерные задачи. Две программы из этого набора написаны на языке Си, а остальные 12 - на Фортране. В пяти программах используется одинарная, а в остальных - двойная точность.

Результаты прогона каждого индивидуального теста из этих двух наборов выражаются отношением времени выполнения одной копии теста на

тестируемой машине к времени ее выполнения на эталонной машине. В качестве эталонной машины используется VAX 11/780. SPEC публикует результаты прогона каждого отдельного теста, а также две составные оценки: SPECint92 - среднее геометрическое 6 результатов индивидуальных тестов из набора CINT92 и SPECfp92 - среднее геометрическое 14 результатов индивидуальных тестов из набора CFP92.

Следует отметить, что результаты тестирования на наборах CINT92 и CFT92 сильно зависят от качества применяемых оптимизирующих компиляторов. Для более точного выяснения возможностей аппаратных средств с середины 1994 года SPEC ввел две дополнительные составные оценки: SPECbase_int92 и SPECbase_fp92, которые накладывает определенные ограничения на используемые компиляторы поставщиками компьютеров при проведении испытаний.

1.3.2. SPECrate_int92, SPECrate_fp92

Составные оценки SPECint92 и SPECfp92 достаточно хорошо характеризуют производительность процессора и системы памяти при работе в однозадачном режиме, но они совершенно не подходят для оценки производительности многопроцессорных и однопроцессорных систем, работающих в многозадачном режиме. Для этого нужна оценка пропускной способности системы или ее емкости, показывающая количество заданий, которое система может выполнить в течение заданного интервала времени. Пропускная способность системы определяется прежде всего количеством ресурсов (числом процессоров, емкостью оперативной и кэш-памяти, пропускной способностью шины), которые система может предоставить в распоряжение пользователя в каждый момент времени. Именно такую оценку, названную SPECrate и заменившую ранее применявшуюся оценку SPECthruput89, SPEC предложила в качестве единицы измерения производительности многопроцессорных систем [3].

При этом для измерения выбран метод "однородной нагрузки" (homogenous capacity metod), заключающийся в том, что одновременно выполняются несколько копий одной и той же тестовой программы. Результаты этих тестов показывают, как много задач конкретного типа могут быть выполнены в указанное время, а их средние геометрические значения (SPECrate_int92 - на наборе тестов, измеряющих производительность целочисленных операций и SPECrate_fp92 - на наборе тестов, измеряющих производительность на операциях с плавающей точкой) наглядно отражают пропускную способность однопроцессорных и многопроцессорных конфигураций при работе в многозадачном режиме в системах коллективного пользования. В качестве тестовых программ для проведения испытаний на пропускную способность выбраны те же наборы CINT92 и CFT92.

При прогоне тестового пакета делаются независимые измерения по каждому отдельному тесту. Обычно такой параметр, как количество запускаемых копий каждого отдельного теста, выбирается исходя из соображений оптимального использования ресурсов, что зависит от архитектурных особенностей конкретной системы. Одной из очевидных возможностей является установка этого параметра равным количеству процессоров в системе. При этом все копии отдельной тестовой программы запускаются одновременно, и фиксируется время завершения последней из всех запущенных программ.

1.3.3. TPC

По мере расширения использования компьютеров при обработке транзакций в сфере бизнеса все более важной становится возможность справедливого сравнения систем между собой. С этой целью в 1988 году был создан Совет по оценке производительности обработки транзакций (TPC - Transaction Processing Performance Council), который представляет собой бесприбыльную организацию. Любая компания или организация может стать членом TPC после уплаты соответствующего взноса. На сегодня членами TPC

являются практически все крупнейшие производители аппаратных платформ и программного обеспечения для автоматизации коммерческой деятельности. К настоящему времени TPC создал три тестовых пакета для обеспечения объективного сравнения различных систем обработки транзакций и планирует создать новые оценочные тесты [1].

В компьютерной индустрии термин транзакция (transaction) может означать почти любой вид взаимодействия или обмена информацией. Однако в мире бизнеса "транзакция" имеет вполне определенный смысл: коммерческий обмен товарами, услугами или деньгами. В настоящее время практически все бизнес-транзакции выполняются с помощью компьютеров. Наиболее характерными примерами систем обработки транзакций являются системы управления учетом, системы резервирования авиабилетов и банковские системы. Таким образом, необходимость стандартов и тестовых пакетов для оценки таких систем все больше усиливается.

До 1988 года отсутствовало общее согласие относительно методики оценки систем обработки транзакций. Широко использовались два тестовых пакета: Дебет/Кредит и TPI. Однако эти пакеты не позволяли осуществлять адекватную оценку систем: они не имели полных, основательных спецификаций; не давали объективных, проверяемых результатов; не содержали полного описания конфигурации системы, ее стоимости и методологии тестирования; не обеспечивали объективного, беспристрастного сравнения одной системы с другой.

Чтобы решить эти проблемы, и была создана организация TPC, основной задачей которой является точное определение тестовых пакетов для оценки систем обработки транзакций и баз данных, а также для распространения объективных, проверяемых данных в промышленности.

TPC публикует спецификации тестовых пакетов, которые регулируют вопросы, связанные с работой тестов. Эти спецификации гарантируют, что покупатели имеют объективные значения данных для сравнения производительности различных вычислительных систем. Хотя реализация

спецификаций оценочных тестов оставлена на усмотрение индивидуальных спонсоров тестов, сами спонсоры, объявляя результаты TPC, должны представить TPC детальные отчеты, документирующие соответствие всем спецификациям. Эти отчеты, в частности, включают конфигурацию системы, методику калькуляции цены, диаграммы значений производительности и документацию, показывающую, что тест соответствует требованиям атомарности, согласованности, изолированности и долговечности (ACID - atomicity, consistency, isolation, and durability), которые гарантируют, что все транзакции из оценочного теста обрабатываются должным образом.

TPC определяет и управляет форматом нескольких тестов для оценки производительности OLTP (On-Line Transaction Processing), включая тесты TPC-A, TPC-B и TPC-C. Как уже отмечалось, создание оценочного теста является ответственностью организации, выполняющей этот тест. TPC требует только, чтобы при создании оценочного теста выполнялись определенные условия. Хотя упомянутые тесты TPC не являются характерными тестами для оценки производительности баз данных, системы реляционных баз данных являются ключевыми компонентами любой системы обработки транзакций.

Следует отметить, что как и любой другой тест, ни один тест TPC не может измерить производительность системы, которая применима для всех возможных сред обработки транзакций, но эти тесты действительно могут помочь пользователю справедливо сравнивать похожие системы. Однако, когда пользователь делает покупку или планирует решение о покупке, он должен понимать, что никакой тест не может заменить его конкретную прикладную задачу.

1.3.3.1. Тест TPC-A

Выпущенный в ноябре 1989 года, тест TCP-A предназначался для оценки производительности систем, работающих в среде интенсивно обновляемых баз данных, типичной для приложений интерактивной обработки данных (OLDP - on-line data processing). Такая среда характеризуется:

- множеством терминальных сессий в режиме on-line

- значительным объемом ввода/вывода при работе с дисками
- умеренным временем работы системы и приложений
- целостностью транзакций.

Практически при выполнении теста эмулируется типичная вычислительная среда банка, включающая сервер базы данных, терминалы и линии связи. Этот тест использует одиночные, простые транзакции, интенсивно обновляющие базу данных. Одиночная транзакция (подобная обычной операции обновления счета клиента) обеспечивает простую, повторяемую единицу работы, которая проверяет ключевые компоненты системы OLTP.

Тест TPC-A определяет пропускную способность системы, измеряемую количеством транзакций в секунду (tps A), которые система может выполнить при работе с множеством терминалов. Хотя спецификация TPC-A не определяет точное количество терминалов, компании-поставщики систем должны увеличивать или уменьшать их количество в соответствии с нормой пропускной способности. Тест TPC-A может выполняться в локальных или региональных вычислительных сетях. В этом случае его результаты определяют либо "локальную" пропускную способность(TPC-A-local Throughput), либо "региональную" пропускную способность (TPC-A wide Throughput). Очевидно, эти два тестовых показателя нельзя непосредственно сравнивать. Спецификация теста TPC-A требует, чтобы все компании полностью раскрывали детали работы своего теста, свою конфигурацию системы и ее стоимость (с учетом пятилетнего срока обслуживания). Это позволяет определить нормализованную стоимость системы ($/tpsA).

1.3.3.2. Тест TPC-B

В августе 1990 года TPC одобрил TPC-B, интенсивный тест базы данных, характеризующийся следующими элементами:

- значительный объем дискового ввода/вывода
- умеренное время работы системы и приложений
- целостность транзакций.

TPC-B измеряет пропускную способность системы в транзакциях в секунду (tpsB). Поскольку имеются существенные различия между двумя тестами TPC-A и TPC-B (в частности, в TPC-B не выполняется эмуляция терминалов и линий связи), их нельзя прямо сравнивать.

1.3.3.2. Тест TPC-C

Тестовый пакет TPC-C моделирует прикладную задачу обработки заказов. Он моделирует достаточно сложную систему OLTP, которая должна управлять приемом заказов, управлением учетом товаров и распространением товаров и услуг. Тест TPC-C осуществляет тестирование всех основных компонентов системы: терминалов, линий связи, ЦП, дискового в/в и базы данных.

TPC-C требует, чтобы выполнялись пять типов транзакций:

- новый заказ, вводимый с помощью сложной экранной формы
- простое обновление базы данных, связанное с платежом
- простое обновление базы данных, связанное с поставкой
- справка о состоянии заказов

справка по учету товаров

Среди этих пяти типов транзакций по крайней мере 43% должны составлять платежи. Транзакции, связанные со справками о состоянии заказов, состоянии поставки и учета, должны составлять по 4%. Затем измеряется скорость транзакций по новым заказам, обрабатываемых совместно со смесью других транзакций, выполняющихся в фоновом режиме.

База данных TPC-C основана на модели оптового поставщика с удаленными районами и товарными складами. База данных содержит девять таблиц: товарные склады, район, покупатель, заказ, порядок заказов, новый заказ, статья счета, складские запасы и история.

Обычно публикуются два результата. Один из них, tpm-C, представляет пиковую скорость выполнения транзакций (выражается в количестве транзакций в минуту). Второй результат, $/tpm-C, представляет собой нормализованную стоимость системы. Стоимость системы включает все

аппаратные средства и программное обеспечение, используемые в тесте, плюс стоимость обслуживания в течение пяти лет.

Синтетические ядра и натуральные тесты не могут служить в качестве настоящих тестовых пакетов для оценки систем: они не могут моделировать точно среду конечного пользователя и оценивать производительность всех относящихся к делу компонентов системы. Без такой гарантии результаты измерения производительности остаются под вопросом.

1.4. Современные базовые конфигурации компьютеров

Подбор конфигурации компьютера исходя из предполагаемых материальных затрат на покупку и/или списка задач, которые будут на нем решаться - это то, чем вынужден занимаются практически любой человек, решивший приобрести компьютер. Чтобы правильно сделать свой выбор необходимо рассмотреть критерии, по которым он производится [7]:

Производительность

Ограничивать производительность ПК "сверху", в принципе, нет никакого смысла: "если можно в 100 раз быстрее, чем у всех, но за те же деньги, то почему бы и нет?". А вот нижний предел обусловлен, прежде всего, требованиями наиболее распространенного в данный момент в

пользовательской среде программного обеспечения. К примеру, стандартом де-факто для офисного компьютера на данный момент является ОС семейства Microsoft Windows не младше Windows 98SE, то, соответственно, офисный компьютер, на котором эта операционная система работать не сможет, вряд ли сможет удовлетворить покупателя, даже если будет стоить 100 рублей. При этом подразумевается, что пользователь получит возможность не только наблюдать за песочными часами, но и производить осмысленные полезные действия.

Надежность

Современные технологии в области компьютерного "железа" идут вперед десятимильными шагами, но при составлении конфигурации ПК, предназначенного для широкого круга пользователей, включать в нее продукты, которым от роду пару недель, вряд ли разумно. Да, это некое устройство может казаться верхом совершенства. Да, пока что отзывы самые положительные, и никто не заметил никаких проблем. В конце концов, да, может быть, их и не будет вообще! А может, и будут... Это попросту неизвестно. Обратите внимание на продукцию ведущих западных брэндов - HP, Dell, IBM. Поначалу может показаться, что их компьютерный модельный ряд несколько консервативен. Однако именно поэтому пользователи, покупающие их компьютеры, могут быть уверены, что установленные там комплектующие не окажутся через полгода "выброшенными на свалку истории", не останутся без технической поддержки и обновлений драйверов и т. д.

Модернизируемость

К сожалению, поддержка аппаратной конфигурацией компьютера дальнейшего наращивания мощности путем замены части (а не всех) основных составляющих по-прежнему остается уделом довольно дорогих моделей. Т.е. за возможность последующего апгрейда платить необходимо сразу при покупке. Соответственно, наиболее "модернизируемыми" являются модели среднего и высшего ценового диапазона. Однако минимальной способностью к апгрейду любой, даже самый дешевый ПК должен обладать. Хотя бы, к примеру,

допускать установку процессора с в полтора-два раза более высокой тактовой частотой и удвоение (желательно - утроение) объема памяти.

В таблице 1.1. приведено описание современных компьютерных конфигураций с указанием, в каких областях жизни они применяются.

Таблица 1.1.

Описание современных компьютерных конфигураций

Компонент	Office low-end	Office middle-end	Office high-end (1)	Office high-end (2)	Entertainment low-end	Entertainment middle-end	Entertainment high-end
Платформа Intel							
Процессор	Celeron 700 МГц	Celeron 800 МГц	Pentium III 866 МГц	Pentium 4 1500 МГц	Celeron 700 МГц	Celeron 1000 МГц	Pentium 4 2000 МГц
Чипсет (системная плата)	mATX i810 или VIA PLE133	ATX VIA Apollo Pro133T	ATX i815EP B-step	ATX i845	ATX VIA Apollo Pro133T	ATX i815EP B-step	ATX VIA Apollo P4X266
Объем ОЗУ, Мбайт	64 PC133	128 PC133	256 PC133	256 PC133	128 PC133	256 PC133	512 PC2100
Ориентировочная цена, $*	100-120	145-155	270-290	330-380	130-140	200-220	680-730
Платформа AMD							
Процессор	Duron 700 МГц	Duron 800 МГц	Duron 1000 МГц	Athlon XP 1500+	Duron 700 МГц	Duron 1000 МГц	Athlon XP 1800+
Чипсет (системная плата)	mATX VIA Apollo KLE133	ATX VIA Apollo KT133A	ATX VIA Apollo KT266A	ATX VIA Apollo KT266A	mATX VIA Apollo KT133A	ATX VIA Apollo KT266A	ATX VIA Apollo KT266A
Объем ОЗУ, Мбайт	64 PC133	128 PC133	256 PC2100	256 PC2100	128 PC133	256 PC2100	512 PC2100
Ориентировочная цена, $	110-120	150-160	235-255	290-320	135-145	235-255	430-470
Независимо от платформы							
Видеокарта (графический чип)	Встроенная	ATI Xpert 2000 Pro	ATI Radeon VE 32 DDR	Matrox G450 32 DDR	GeForce2 MX	GeForce2 Pro	GeForce3
Звуковая карта	AC'97	AC'97	PCI Yamaha 744	PCI Yamaha 744	PCI Yamaha 744	SB Live! Player	Hercules Game Theatre XP
Жесткий диск	10 GB, 5400 об/мин	20 GB, 5400 об/мин	30 GB, 5400 об/мин	30 GB, 5400 об/мин	20 GB, 5400 об/мин	40 GB, 7200 об/мин	60 GB, 7200 об/мин
CD-ROM	-	+	+	+	+	+	DVD-ROM
Ориентировочная цена, $	75	155-165	220-230	250-260	190-200	335-355	550-580
Общая цена Intel/AMD, $**	250-270/ 260-270	375-395/ 380-400	565-595/ 530-560	655-715/ 615-655	395-415/ 400-420	610-650/ 645-685	1305-1385/ 1055-1125

Office low-end

Минимальная конфигурация, которая формируется по принципу "главное - цена, со всем остальным будем как-то мириться". Соответственно, используется материнская плата даже без слота AGP, т. е. возможности дальнейшего апгрейда системы ограничены фактически только установкой более мощного CPU и наращиванием объема памяти. В то же время такой ПК вполне позволяет выполнять ограниченный набор офисных приложений - текстовый редактор, электронная таблица, Internet-броузер и почтовый клиент.

При увеличении объема оперативной памяти работа становится более комфортной, но в большинстве случаев можно обойтись и без этого. Windows 2000 (и тем более Windows XP) такой машине "категорически противопоказаны", впрочем, как и офисные пакеты последнего поколения.

Office middle-end

Полновесная рабочая машина, оснащенная мощным процессором и достаточным объемом оперативной памяти, позволяет удовлетворить потребности практически любого офисного работника. Кроме использования в качестве "электронной печатной машинки", такой ПК может стать вполне удобным рабочим местом как для бухгалтера, так и для "внутриофисного дизайнера", работающего с несложной бизнес-графикой. Благо, видеокарта от ATI обеспечивает превосходное качество изображения даже в очень высоких разрешениях.

Наличие 128 Мбайт ОЗУ, в принципе, позволяет при желании установить на такой компьютер даже Windows 2000, хотя рекомендовать эту ОС для обычного офисного ПК пока что вряд ли разумно. Оптимальным путем увеличения производительности является установка большего объема памяти - процессора с частотой 800 МГц, работающего на 100-мегагерцевой шине, вряд ли станет "не хватать" для офисной работы в течение еще, как минимум, года-полутора.

Office high-end

В категорию офисных эти конфигурации попали исключительно по той причине, что вводить разделение ПК более чем на две группы нецелесообразно. Секретаря, работающего с электронными документами и при этом испытывающего острую необходимость в Pentium 4 1.5 ГГц, представить сложно. То есть если первые две конфигурации носили чисто офисный характер - документооборот, бухгалтерия и работа в Сети, то "офисный high-end" - это рабочее место некоего "продвинутого пользователя", который тоже находится в офисе, но занимается не только составлением и просмотром

документов, но и верстkой, дизайном, работой со звуком или видео либо, к примеру, написанием несложных программ "для внутреннего использования". В этом случае мощный процессор и большой объем оперативной памяти будет востребован. К тому же эти ПК имеют прекрасные показатели модернизируемости - поддержка современных высокочастотных CPU и DDR-памяти (в случае платформы AMD) позволит "достраивать" эти системы без кардинальной переделки еще довольно продолжительное время.

Entertainment low-end

Часто компьютер используется не только (а нередко даже и не столько) для работы: домашние ПК также весьма распространены. Но требования, предъявляемые к этим двум разновидностям одного и того же компьютера, совершенно разные. Для пользователя такой системы основным ее предназначением являются игры и развлечения, что накладывает определенные ограничения "снизу" на аппаратную конфигурацию - звук AC'97 и слабый по производительности 3D-акселератор в домашнем компьютере уже совершенно неуместны. Entertainment low - если финансовые ограничения очень жесткие, то он позволит за скромную сумму получить устройство, с которого пользователь начнет путешествие в мир компьютерных развлечений. К тому же возможности дальнейшего апгрейда этой системы достаточно широки, что при наличии финансов, желания или необходимости позволит постепенно наращивать ее мощность, не прибегая к большим одноразовым денежным затратам.

Entertainment middle-end

Достаточно мощная конфигурация, позволяющая особо не задумываться при покупке каждой новой игры над вопросами: "А хватит ли ей мощности моего ПК? А не превратится ли процесс игры в созерцание слайд-шоу?". Кроме того, наличие аудиокарты, поддерживающей пятиканальный звук, даст возможность организовать на базе этого ПК домашний кинотеатр или любительскую звуковую студию. В конфигурацию на базе процессора AMD Duron 1 ГГц установили материнскую плату на чипсете VIA Apollo KT266A с

поддержкой DDR-памяти, даже, несмотря на то, что по результатам тестирования Duron в комбинации с PC2100 DDR существенного прироста производительности по сравнению с PC133 не наблюдается.

Entertainment high-end

Обладатель такой конфигурации может забыть обо всех возможных неудобствах, возникающих в процессе игр. "Все на максимум" - вот оптимальный набор опций для практически любой из существующих сегодня игр, если она запускается на этом ПК. Звуковая карта обеспечивает прекрасное качество звука и поддержку всех современных стандартов трехмерного аудио. При наличии высококачественной акустики эта система вполне может стать домашним центром развлечений. Говорить же о рабочем ее применении совершенно бессмысленно, так как всем уже давно известно, что требования игровых приложений к мощности ПК на порядок превосходят требования прикладного ПО.

1.5. Вычисление рекурсивных функций

Рекурсия – процесс определения либо выражения функции, процедуры, языковой конструкции, решения задачи через них самих.

Рекурсия в вычислениях - это ситуация, при которой для вычисления некоторого значения величины (функции) используется та же самая величина (функция), вычисленная при других условиях (аргументах функции). Свойство рекурсивности не ограничивается вычислительными задачами. Это общее свойство, которым могут обладать алгоритмы и программы любой природы.

Рекурсия в программе (алгоритме) - это способность программы (процедуры, алгоритма) обращаться к самой себе для выполнения однотипной последовательности операций при различных внешних условиях (параметрах). Признак наличия рекурсии - это конструкция вызова программы (процедуры), которая присутствует в тексте этой программы.

Рекурсии в алгоритмах и программах являются мощным и эффективным средством программирования. С одной стороны, они отражают внутреннюю

природу задачи и метода ее решения, при котором различные части алгоритма подобны ему самому. Можно сказать, что рекурсивные алгоритмы - это алгоритмическое проявление свойств равенства и подобия, реально существующих в мире предметов. С другой стороны, рекурсии позволяют в максимальной степени использовать преимущества процедурного программирования - процедура или функция использует вызов своих собственных операций. В результате текст программы получается наиболее компактным и удобным для восприятия [6].

Каждый рекурсивный вызов процедуры и функции использует часть оперативной памяти компьютера, организованную специальным образом - *стек*. Стек сохраняет информацию, необходимую для возврата к выполнению вызывающей программы. Поскольку размер стека, как и любой памяти, ограничен, то слишком большое количество конструкций вызова в реальных условиях оказывается невозможным. Это важно иметь в виду, когда предполагаемое число рекурсивных обращений (глубина рекурсии) может измеряться сотнями.

Наиболее известной из полностью рекурсивных функций является функция Аккермана. Целесообразно проверить возможность использования этой функции при тестировании различных компьютерных конфигураций.

1.6. Постановка задачи

Исходя из общих требований, представленных в задании на дипломный проект, сформулируем более подробное описание разрабатываемого пакета тестовых программ.

Требуется разработать алгоритм программы, вычисляющей функцию Аккермана, и реализовать по этому алгоритму тест производительности вычислительных систем. Программа должна обладать простым и удобным пользовательским интерфейсом: пользователю необходимо лишь выбрать требуемый тест и указать его параметры. При помощи разрабатываемого теста необходимо сравнить последовательности компьютерных конфигураций. Чтобы сравнение было объективным те же конфигурации должны быть протестированы стандартными тестами измерения производительности. Полученные результаты необходимо проанализировать, и на их основе должны быть сделаны соответствующие выводы.

Завершающим этапом проектирования является составление программной документации, включающей в себя помимо технического задания тексты программы, описание программы, а также программу и методику испытаний.

2. Проектирование и разработка программного продукта

Разрабатывемый программный продукт содержит три тестовых программы: одна основная и две дополнительные. В качестве дополнительных, были выбраны стандартные тесты измерения производительности компьютеров: Whetstone и Dhrystone. Эти программы характеризуют обработку вещественных чисел и целочисленную обработку данных, соответственно. Основным тестом является программа, основанная на вычислении функции Аккермана.

2.1. Описание дополнительных тестов

2.1.1. Whetstone (описание процедуры)

Описание алгоритма работы теста

Комплект тестов Whetstone состоит из нескольких модулей, имитирующих программную нагрузку в наиболее типичных режимах исполнения вычислительных задач (арифметика с плавающей точкой, операторы типа IF, вызовы функций и т.д.). Каждый модуль выполняется многократно, в соответствии с исходной статистикой Whetstone-инструкций (практически это реализуется с помощью заключения модулей в циклические конструкции с разным числом “оборотов” цикла), а производительность рассчитывается как отношение числа Whetstone-инструкций к суммарному времени выполнения всех модулей пакета. Этот результат представляется в KWIPS (Kilo Whetstone Instructions Per Second) или в MWIPS (Mega Whetstone Instructions Per Second). Существенный плюс данных оценок состоит в том, что Whetstone-инструкции не привязаны к системе команд какого-либо компьютера, т. е. оценка производительности в MWIPS является моделенезависимой [1].

Программа была адаптирована следующим образам: дополнительно был введен параметр «время выполнения теста» - этот параметр задается пользователем. Это было сделано для того, чтобы любой функции Аккермана (с любыми параметрами) можно было сопоставить число проходов (количество выполненных внешних циклов) теста Whetstone, что позволяет оценить количество выполненных процессором операций за время вычисления заданной функции Аккермана.

По приведенному выше описанию работы теста, была составлена общая схема программы Whetstone, изображенная на рисунке 2.1.

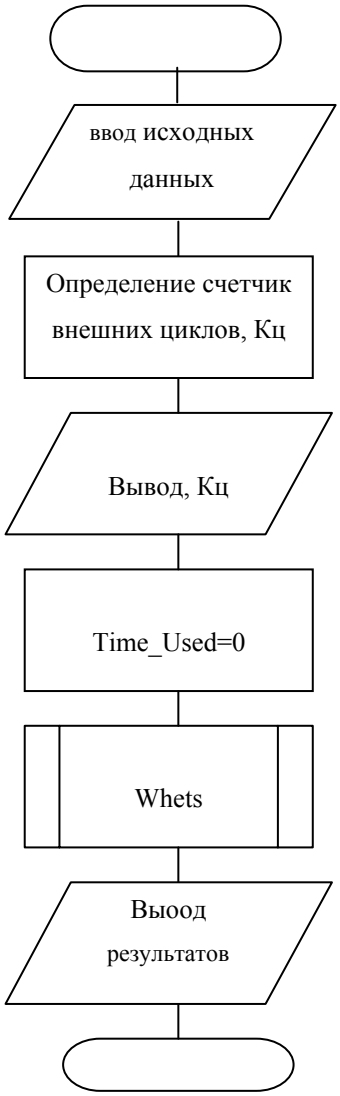

Рис.2.1. Общая схема программы Whetstone

Основная логика программы включена в функцию Whets. В рамках этой функции реализована программная нагрузка, представляющая собой восемь последовательно выполняемых модулей:

- вычисление элементов массива
- вычисление элементов массива (массив задается как параметр функции)
- операции ветвления (if – else)
- целочисленная арифметика (опреции вычитания, сложения и умножения)
- тригонометрические функции (sin, cos, atan)
- вызов процедур, оперирующих с указателями

- обработка массива ссылок
- вызов стандартных функций (sqrt, log, exp)

схема алгоритма процедуры тестирования представлена на рисунке 2.2.

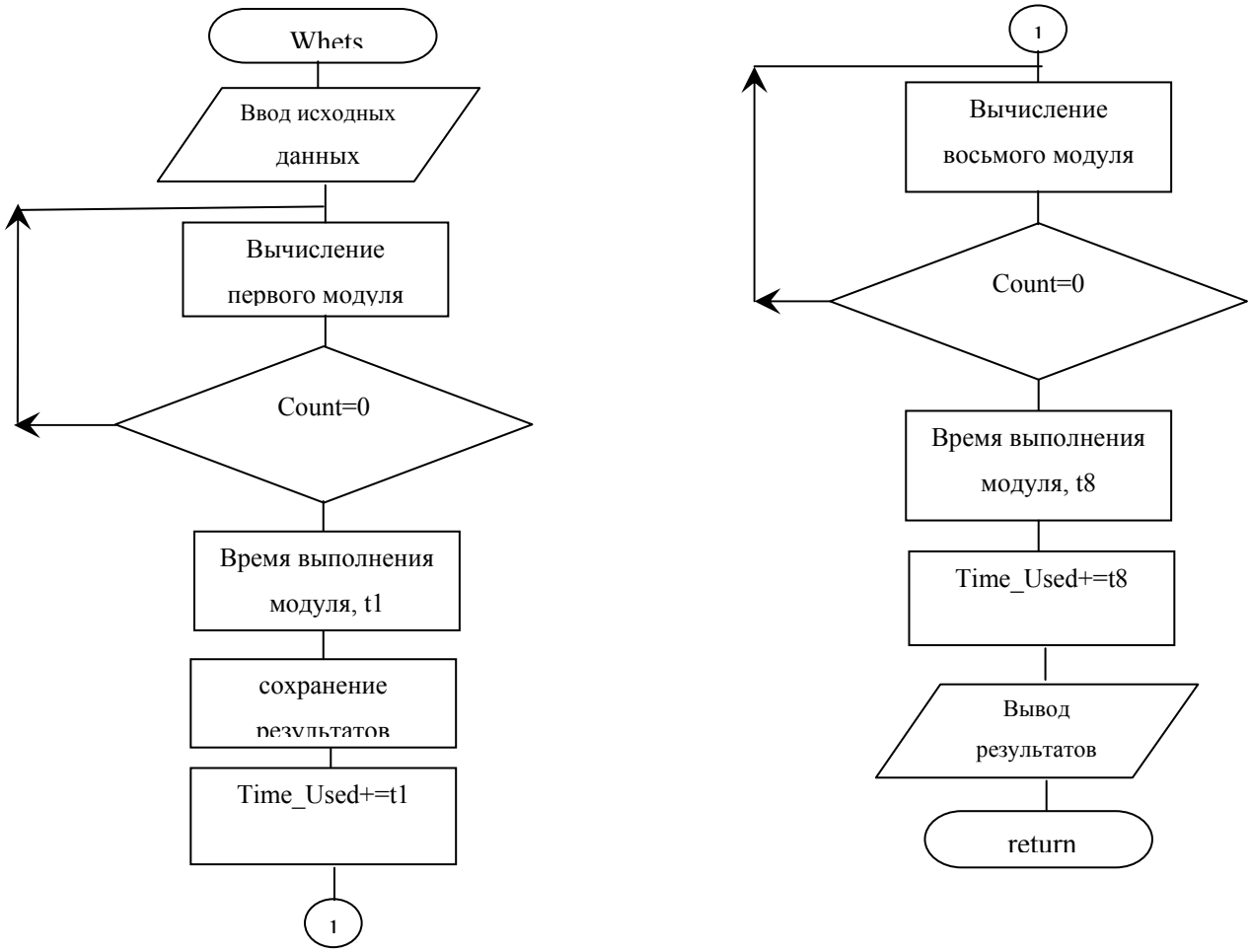

Рис. 2.2. Схема алгоритма процедуры тестирования

Интерпретация результатов

- Пакет Whetstone ориентирован на оценку производительности обработки чисел с плавающей точкой: почти 70% времени выполнения приходится на “плавающую” арифметику и исполнение библиотечных математических функций.
- Большое число обращений к библиотеке математических функций, заложенное в тесты Whetstone, требует особой осторожности при сравнении результатов, полученных для разных компьютеров: фирмы-изготовители имеют возможность оптимизировать оценку Whetstone, внося изменения в библиотеку.
- Поскольку тестовые модули Whetstone представлены очень компактным исполнительным кодом, для современных процессоров они не позволяют оценить эффективность механизма динамической подкачки команд в кэш инструкций: любой модуль Whetstone целиком размещается в кэш-памяти даже самой малой емкости.
- Особенностью рассматриваемых тестов является почти полное отсутствие локальных переменных. В результате оценки Whetstone в значительной степени зависят от эффективности функционирования ресурсов компьютера, обеспечивающих доступ к оперативной памяти и буферизацию данных в процессоре (включая количество регистров, емкость кэш-памяти данных и механизм ее замещения). Однако это же обстоятельство делает тесты Whetstone практически нечувствительными к средствам повышения эффективности работы с локальными переменными.

2.1.2. Dhrystone (описание процедуры)

Описание алгоритма работы теста

Тесты Dhrystone предназначены для оценки производительности функционирования конкретных видов системного и прикладного ПО (операционные системы, компиляторы, редакторы и т.д.). Это наложило заметный отпечаток на структуру данных и исполнительного кода: в тестах Dhrystone отсутствует обработка чисел с плавающей точкой, зато преобладают операции над другими типами данных (символы, строки, логические переменные, указатели и т. п.). Кроме того, по сравнению с тестами Whetstone уменьшено количество циклических конструкций, используются более простые вычислительные выражения, но возросло число операторов IF и вызовов процедур.

Тестовые процедуры Dhrystone объединены в один измерительный цикл, который содержит 103 оператора в С-версии. Этот глобальный цикл принят за единицу работы (один Dhrystone), а производительность измеряется в количестве измерительных циклов, выполненных за секунду (Dhrystone s/s). Правда, в последнее время при публикациях оценок Dhrystone стали применяться и другие единицы измерения - MIPS VAX. Такое отклонение от стандартных правил продиктовано двумя обстоятельствами: во-первых, единицы Dhrystone s/s выглядят слишком экзотично; во-вторых, оценка в MIPS VAX по смыслу совпадает с условными единицами очень распространенных тестов SPEC, характеризующих производительность компьютера относительно системы VAX 11/780 (например, 1,5 MIPS VAX означает, что тестируемая система работает в полтора раза быстрее VAX 11/780) [1].

2.2. Разработка теста на базе функции Аккермана

2.2.1.Применение рекурсии для оценки производительности ВС

Рекурсия выражает характерное свойство объекта, которое заключается в том, что объект ссылается на самого себя. В программировании термином «рекурсия» описывают ситуацию, когда программа продолжает вызывать саму себя до тех пор, пока выполняется некоторое заданное условие. Для решения таких задач необходимо оценить эффективность работы вычислительной системы на которой решается данная задача при выполнении рекурсии. Ниже рассмотрены некоторые типичные ситуации использования рекурсивных алгоритмов и программ для решения реальных задач [12].

Факториал

Рассмотрим некоторую функцию, определенную следующим образом: F(n) = 1 * 2 * 3 * ... * n, где n - заданное натуральное число. Такое произведение натуральных чисел от 1 до n называется в математике факториалом числа n и обозначается следующим образом: «n!». В общем случае значение «n!» определяется и для нулевого значения. Таким образом, полное определение факториала неотрицательного числа:

1. 0! = 1
2. n! = 1 * 2 * 3 * ... * n, для n > 0

Функция «n!» имеет большое значение в тех задачах математики, которые имеют дело с получением и исследованием различных вариантов. Так, например, количество различных способов расположения группы из n различных предметов равно «n!». Здесь свойство рекурсивности состоит в том, что значение факториала любого числа n может быть получено на основе значения факториала для предшествующего числа: (n-1).

Действительно, нетрудно заметить, что:

n! = (n-1)! * n

4! = (1 * 2 * 3) * 4 = 3! * 4

Другими словами, для определения алгоритма вычисления факториала можно задать следующие соотношения:

0! = 1 (2.1)

n! = (n-1)! * n, для n > 0 (2.2)

Обратим внимание на выражение под номером 2.2. И в правой, и в левой частях выражения присутствует одна и та же вычисляемая функция! Чтобы вычислить факториал n, необходимо вычислить (n-1)!, для вычисления (n-1)! необходимо знать, чему равно (n-2)! и т.д.

Задача о Ханойских башнях

Эта интересная задача была сформулирована в 1883 году математиком Э.Люка. Многие учебные пособия по программированию используют ее как классический пример рекурсивного алгоритма. Суть задачи состоит в следующем. Где-то далеко в окрестностях города Ханой стоят три башни. Одна из башен состоит из 64 дисков, положенных друг на друга - как кольца в детской пирамиде. Диаметры всех дисков различны и уменьшаются от нижнего к верхнему. Необходимо перенести все диски с одной башни на другую. При этом диски можно переносить только по одному с любой башни на другую. Ставить диск на землю или надевать больший на меньший нельзя.

Рекурсивность алгоритма решения задачи заметить очень легко. Пусть наши башни будут обозначены буквами A, B, C. Диски, надетые на башню A, следует перенести на башню C. Для упрощения будем рассматривать 3 ситуации - когда башня A состоит из 1, 2 и 3 дисков. Тогда алгоритмы решения будут следующими:

1 ДИСК ***2 ДИСКА*** ***3 ДИСКА***

A -> C	A -> B	A -> C
	A -> C	A -> B
	B -> C	C -> B
		A -> C
		B -> A
		B -> C
		A -> C

Для одного диска операция сводится к переносу этого диска с башни A на башню C. Для двух дисков мы переносим сначала диск с A на B, потом с A на(как в предыдущем случае!), потом с B на C. Для трех дисков предварительно выполняется процедура переноса двух дисков с A на B (A->C,A->B,C->B), потом с A на C и, наконец, перенос двух дисков с B на C (B->A,B->C,A->C). Нетрудно заметить, что для переноса башни из n дисков алгоритм решения имеет следующий вид: сначала башня из (n-1) дисков в верхней части переносится на рабочую башню (B). Потом один нижний диск переносится на башню C. И, наконец, башня из (n-1) дисков переносится с башни B на башню C.

Сортировка одномерного массива

Яркий пример рекурсии - это алгоритм сортировки данных, разработанный К.Хоором и известный под названием метода быстрой сортировки. По своему быстродействию метод существенно превосходит все остальные. Суть алгоритма для сортировки элементов одномерного массива чисел по не убыванию (возрастанию) их значений можно определить следующим образом:

- Пусть x - элемент, расположенный в середине массива. Будем просматривать массив слева направо до тех пор, пока не обнаружим элемент a[i], для которого a[i] > x. Аналогично, просматривая массив, справа налево, ищем элемент a[j], для которого a[j] < x
- Меняем элементы a[i] и a[j] местами. Продолжаем процесс просмотра массива до его середины

- Таким образом, массив будет разбит на две части, в одной из которых будут собраны элементы большие, чем x, а в другой - меньшие, чем x
- Процесс, описанный в пунктах 1-3, следует повторить для каждой из получаемых таким образом частей. Когда размеры частей станут равными одному элементу, массив будет содержать упорядоченную последовательность

Рекурсивность алгоритма сортировки проявляется в том, что для каждой части массива в процессе его разделения на группы элементов осуществляется вызов одной и той же процедуры - фактически эта процедура вызывает саму себя после выделения определенной группы элементов массива и выполняет для этой группы заданные операции.

Программы, вычисляющие функции, обладающие свойством рекурсивности, очень удобно использовать для оценки производительности вычислительных систем, так как вычисление таких функций требует больших затрат вычислительных ресурсов: процессорного времени, оперативной памяти, и др. Ярким примером таких функций является функция, названная в честь У. Аккермана.

2.2.2. Описание функции Аккермана

На сегодняшний день функция Аккермана является одной из самых трудно вычислимых функций, так как она дважды рекурсивна, т.е. она не только определяется через себя, но и является одним из параметров вызываемой функции.

Существует несколько способов задания функции Аккермана: например, они могут различаться количеством параметров. Ниже приведена функция Аккермана от двух и трех параметров.

- функция Аккермана (А) от трех параметров, индуктивно задается на трех неотрицательных целых числах и выглядит следующим образом :

$$A(N,X,Y)=\begin{cases} X+1, & \text{если } N=0 \\ X, & \text{если } N=1,\ Y=0, \\ 0, & \text{если } N=2,\ Y=0, \\ 1, & \text{если } N=3,\ Y=0, \\ 2, & \text{если } N=>4,\ Y=0, \\ A(N-1,A(N,X,Y-1),X), & \text{если } N\#0,\ Y\#0; \end{cases}$$

где N,X,Y - целые неотрицательные числа

- функция Аккермана (А) от двух параметров, индуктивно задается на паре неотрицательных целых числах [9]:

$$A(0,n) = n + 1$$
$$A(m,0) = A(m-1,1)$$
$$A(m,n) = A(m-1, A(m,n-1))$$
$$где \quad m, n \geq 0$$

Значение этой функции очень быстро возрастает по мере увеличения первого из ее параметров(“m”). Рассмотрим функцию Аккермана первого порядка A(1,n): зафиксируем первый параметр и будем постепенно увеличивать второй.

A(1,0)=A(0,1)=1+1=2

A(1,1)=A(0,A(1,0))=1+A(1,0)=1+2=3

A(1,2)=A(0,A(1,1))=1+A(1,1)=1+1+A(1,0)=2+2=4

A(1,3)=A(0,A(1,2))=1+A(1,2)=1+1+A(1,1)=1+1+1+A(1,0)=3+2=5

…

A(1,n)=n+2

Наблюдается линейная зависимость между значением функции и параметром “n”. Аналогично рассмотрим функцию Аккермана второго порядка:

A(2,0)=A(1,1)=1+2=3

A(2,1)=A(1,A(2,0))=A(0,A(1,(A(2,0)-1)))=2+3=5

A(2,2)=A(1,A(2,1))=A(1,5)=A(0,A(1,4))=A(0,A(0,A(1,3)))=…=1+1+1+1+3=4+3=7

…

A(2,n)=2*n+3

Таким образом, были получены формулы для вычисления функции Аккермана со степенью m=2,3,4.

$A(1,n)=n+2$

$A(2,n)=2*n+3$

$A(3,n)=$

$A(4,n)=$

Анализируя полученные выражения, делаем вывод, что функция Аккермана уже третей степени обладает степенной зависимостью от параметра «m», а для четвертой степени – характер зависимости многостепенной. На рисунке 2.1. наглядно продемонстрирован рост функции Аккермана при возрастании ее параметров.

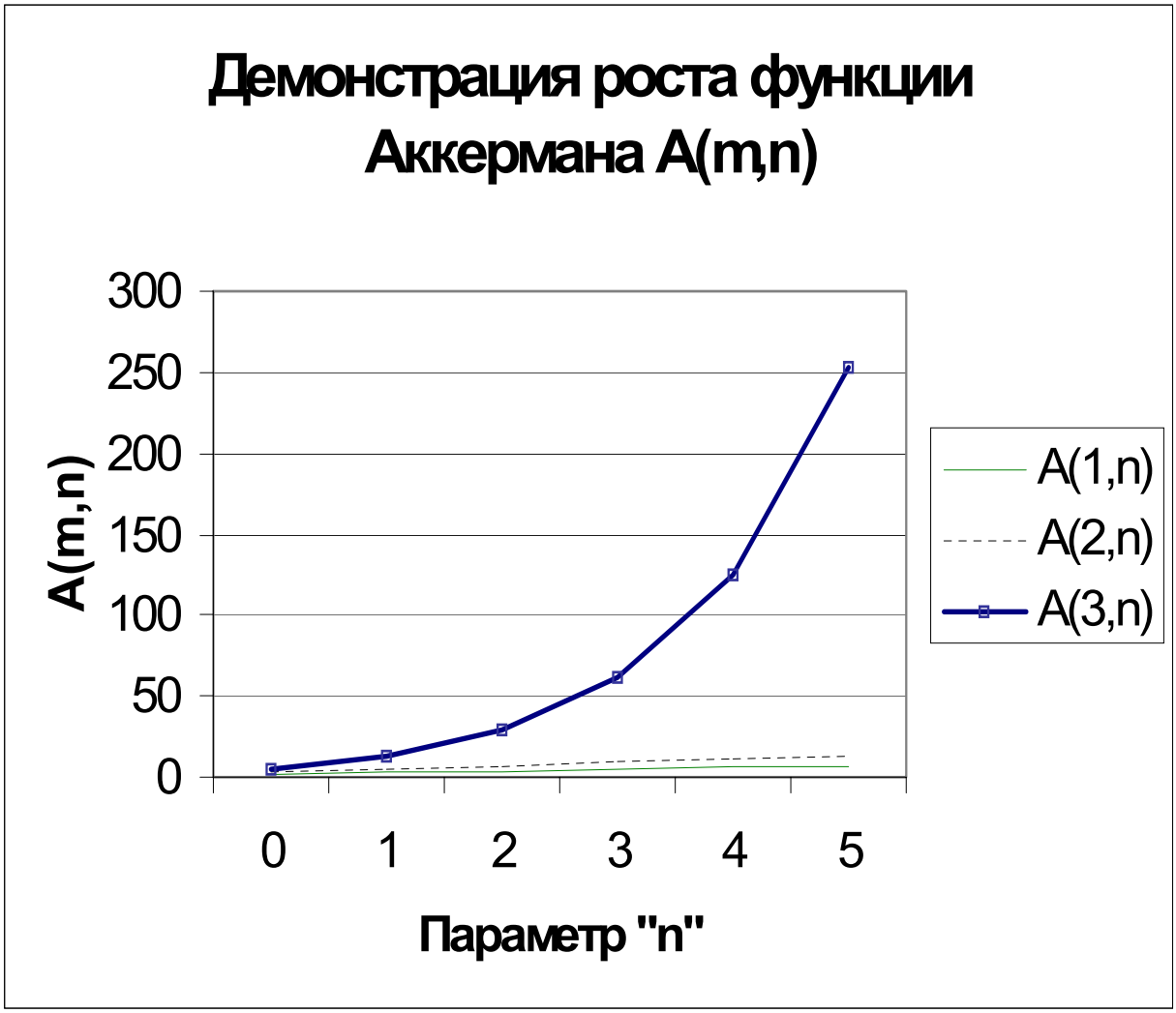

Рис. 2.3. Зависимость значения функции Аккермана от параметра “n”

Анализируя рассмотренные свойства функции Аккермана, делаем вывод, что тест, написанный на базе вычисления этой функции позволит оценить производительность вычислительных систем и даст возможность проверить эффективность выполнения рекурсии на современных вычислительных машинах. Варьируя параметры функции можно регулировать объем вычислений требуемый для получения результата, таким образом, не меняя исходного текста программы, можно сравнить производительность ВС различных поколений.

2.2.3. Разработка алгоритма программы

Для построения алгоритма программы используем сведения, изложенные в разделе 1 о рекурсивных функциях, а также описание свойств функции Аккермана из предыдущего параграфа.

Необходимо разработать программу, вычисляющую рекурсивную функцию, т.е. из функции main вызывается функция Akk:

```
main()          /* вызывающая  функция  */
{... Akk()...}
Akk()           /* рекурсивная функция  */
{ ...Akk()...}
```

Рекурсивная функция Akk имеет параметры P1,P2,...,Ps, внутренние переменные V1,V2,...,Vt и в функциях main и Akk имеется k обращений к функции Akk. Для реализации такой функции требуются следующие дополнительные объекты:

- переменные AR1,AR2,...,ARs, содержащие значения фактических параметров при вызове функции Akk (типы переменных должны соответствовать типам параметров P1,P2,...,Ps);
- переменная rz для вычисляемого функцией Akk результата (тип переменных совпадает с типом возвращаемого значения функции Akk);
- стек, содержащий в себе все параметры и все внутренние переменные функции Akk, а также переменную lr типа int, для хранения точки возврата, и переменную pst типа указатель, для хранения адреса предыдущего элемента стека;
- указатель dl для хранения адреса вершин стека;
- промежуточный указатель u для операций над стеком;
- k новых меток L1,...,Lk для обозначенных точек возврата;
- промежуточная переменная l типа int для передачи номера точки возврата.

Очевидно, что помимо непосредственного вычисления функции Аккермана, программа должна содержать некоторый подготовительный этап и завершающий этап. На подготовительном этапе необходимо реализовать считывание параметров функции, а также проверку их принадлежности множеству допустимых данных функции. Завершающий этап должен содержать функцию вывода и сохранения полученных результатов.

В результате получим следующий набор действий, выполняемый программой:

1) считывание исходных данных;

2) выбор теста;

3) выполнение выбранного теста;

4) вывод результатов тестирования на экран;

5) запись результатов в файл

Полученная обобщенная схема алгоритма программы изображена на рисунке 2.4.

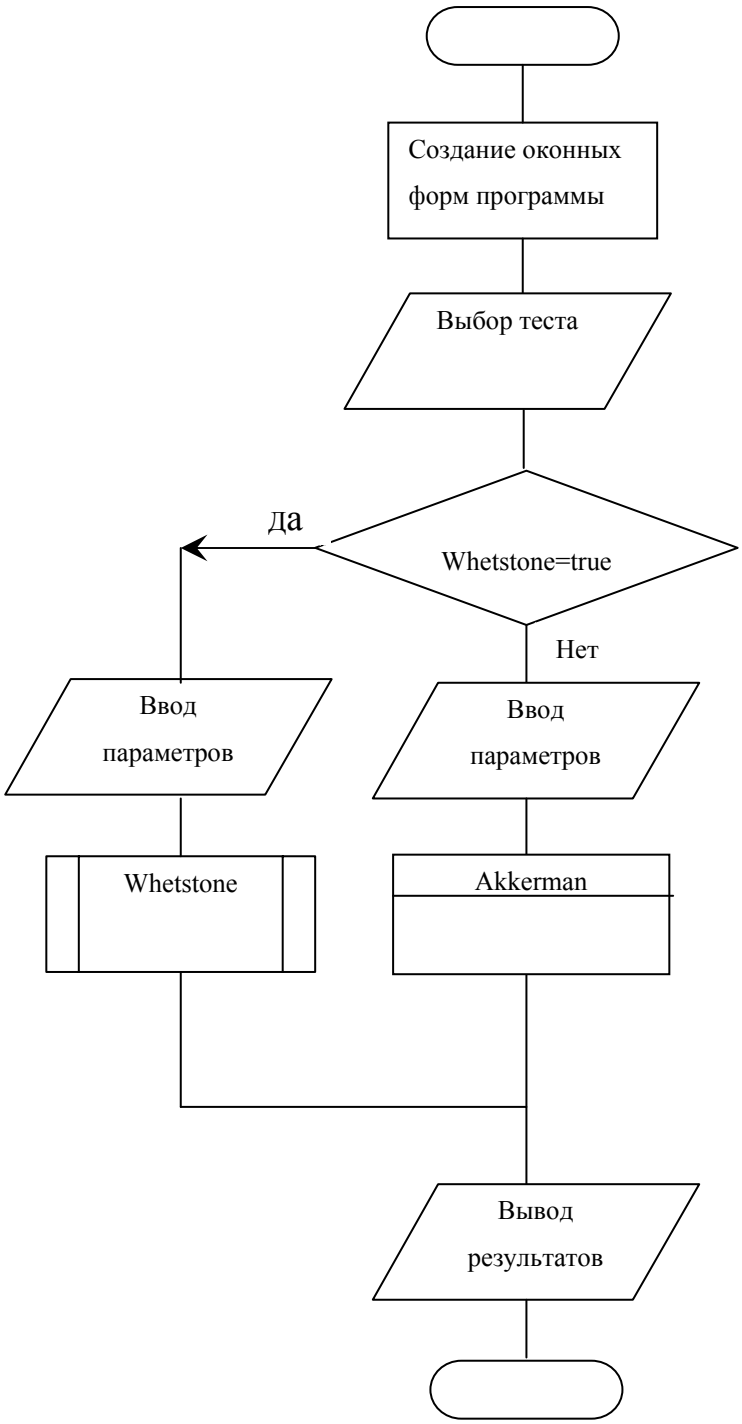

Рис. 2.4. Общая схема алгоритма процесса тестирования

Основная логика программы, реализующая вычисление функции Аккермана, будет включена в функцию Akkerman. Алгоритм функции Akkerman представлен на рисунке 2.5.

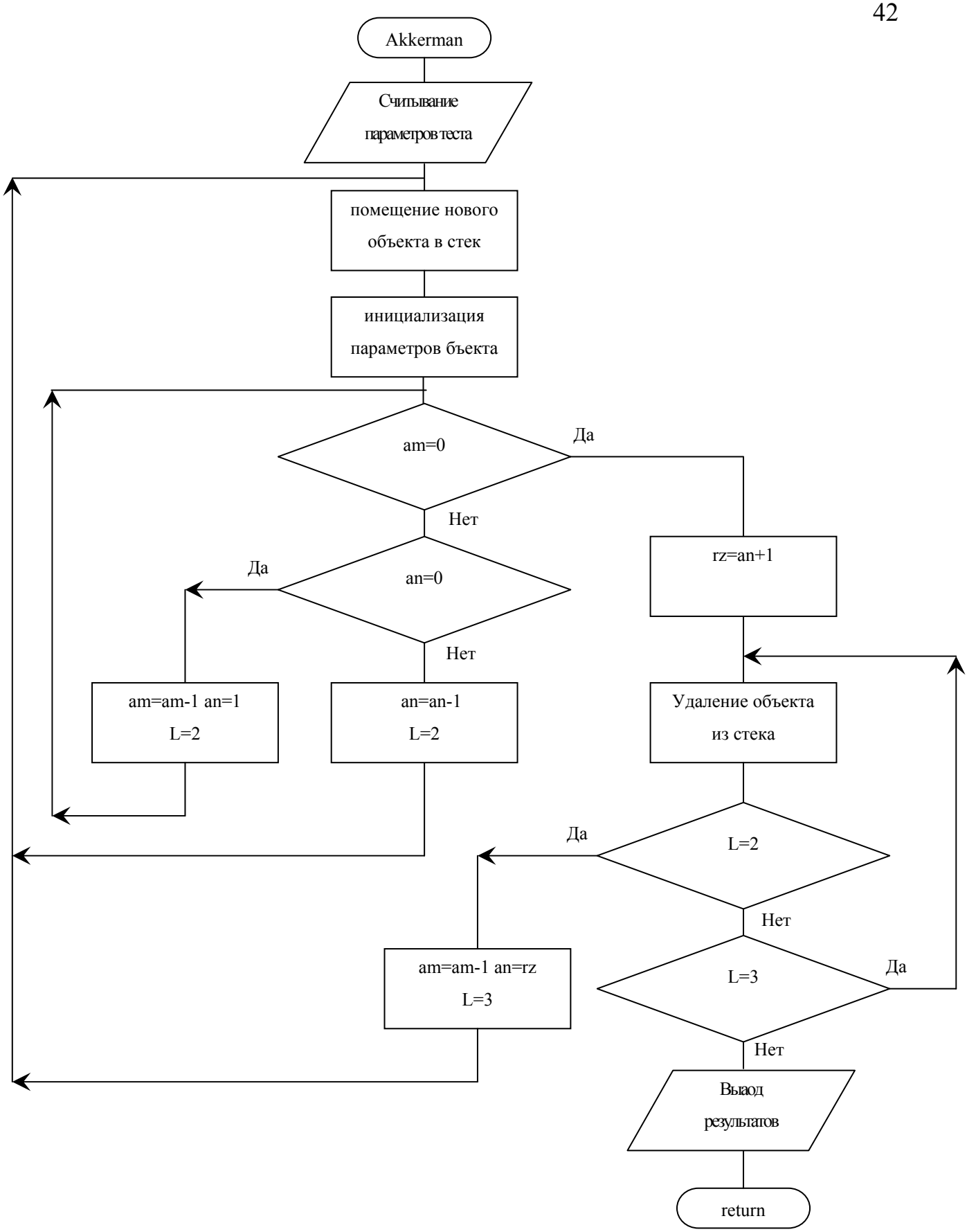

Рис. 2.5. Процедура вычисления функции Аккермана

В алгоритм введено понятие «объект» и анализ полей «объекта». Объект представляет собой структуру со следующими полями:

- Am – хранит первый параметр функции Аккермана
- An – хранит второй параметр функции Аккермана
- Rz – в эту переменную записывается результат вычислению данной функции
- L – уровень текущей функции, подробно это поле будет описано ниже
- Указатель на такую же структуру – это поле необходимо для включения вновь созданного объекта в стек

Таким образом, каждый объект представляет собой функцию Аккермана со своими параметрами. При рекурсивном вызове функции, создается новый объект и помещается в начало стека. Каждому, вновь созданному, объекту присваивается определенный уровень “L”: L=1 – означает, что данная функция Аккермана последняя в стеке и вычисление закончено; L=2 – означает, что данная функция порождает новую функцию Аккермана со следующими параметрами: A(am, an-1); L=3 – означает, что надо вызывать новую функцию Аккермана с параметрами A(am-1,rz), где rz – это результат вычисления функции Аккермана на предыдущем шаге.

2.3. Реализация программного продукта

2.3.1. Выбор технологии программирования

Система C++ Builder производства корпорации Borland предназначена для операционных систем Windows 95 и NT. Интегрированная среда C++ Builder обеспечивает скорость визуальной разработки, продуктивность повторно используемых компонент в сочетании с мощью языковых средств C++, усовершенствованными инструментами и разномасштабными средствами доступа к базам данных.

К несомненным достоинствам C++ Builder можно отнести следующие особенности:

1)Интегрированная среда разработки объединяет Редактор форм, Инспектор объектов, Палитру компонент, Администратор проекта и

полностью интегрированные Редактор кода и Отладчик - инструменты быстрой разработки программных приложений, обеспечивающие полный контроль над кодом и ресурсами.

2)Профессиональные средства языка C++ интегрированы в визуальную среду разработки. C++Builder предоставляет быстродействующий компилятор с языка Borland C++, эффективный инкрементальный загрузчик и гибкие средства отладки как на уровне исходных инструкций, так и на уровне ассемблерных команд - в расчете удовлетворить высокие требования программистов-профессионалов.

3) Конструирование по способу "drag-and-drop " позволяет создавать приложение простым перетаскиванием захваченных мышью визуальных компонент из Палитры на форму приложения. Инспектор объектов предоставляет возможность оперировать со свойствами и событиями компонент, автоматически создавая заготовки функций обработки событий, которые наполняются кодом и редактируются в процессе разработки.

4)Механизмы двунаправленной разработки (two-way-tools) устраняют барьеры между программистом и его кодом. Технология двунаправленной разработки обеспечивает контроль над кодом посредством гибкого, интегрированного и синхронизированного взаимодействия между инструментами визуального проектирования и Редактором кода.

5)Мастер инсталляции руководит созданием унифицированных дистрибутивных пакетов для разработанных приложений.

6)Исходные тексты Библиотеки Визуальных Компонент облегчают разработку новых компонент на базе готовых примеров.

7)Отрытые инструменты API могут быть непосредственно интегрированы в визуальную среду системы. Есть возможность подключения привычного текстового редактора или создания собственного мастера для автоматизации выполнения повторяющихся процедур[11].

Помимо этих и многих других достоинств самой системы разработки C++ Builder можно упомянуть и некоторые удобства самого языка C++, перечисленные в [11]:

1)Оптимизирующий 32-разрядный компилятор построен по компиляторной технологии корпорации Borland, обеспечивающей исключительно падежную и быструю оптимизацию как длины выходного исполняемого кода, так и расходуемой памяти.

2)Инкрементальный линкер осуществляет быструю и надежную сборку приложении в формате EXE файлов сравнительно малого размера. Автоматически устраняя повторную сборку не изменившихся исходных объектных файлов и подключение неиспользуемых функций, инкрементальный линкер строит эффективную выполняемую программу с минимальными потерями времени.

3)Создание DLL, LIB, и EXE файлов предоставляет свободу выбора формата целевого приложения в соответствии с требованиями конкретного проекта.

4)Прямое обращение к системным функциям Windows 95 и NT дает возможность программистам, работающим в среде C++Builder, при необходимости воспользоваться всеми усовершенствованиями современных операционных систем.

5)Поддержка промышленных стандартов ActiveX, OLE, COM, MAPI, Windows Sockets TCP/IP, ISAPI. NSAPI, ODBC, Unicode и MBCS.

Все приведенные выше достоинства среды C++ Builder говорят в пользу его использования при написании пакета тестирующих программ.

2.3.2. Общие принципы реализации приложения

Вся программа включает в себя два файла программного кода: Unit1.cpp и Unit2.cpp, каждый из которых описывает отдельную экранную форму.

Главная экранная форма программы изображена на рисунке 2.6.:

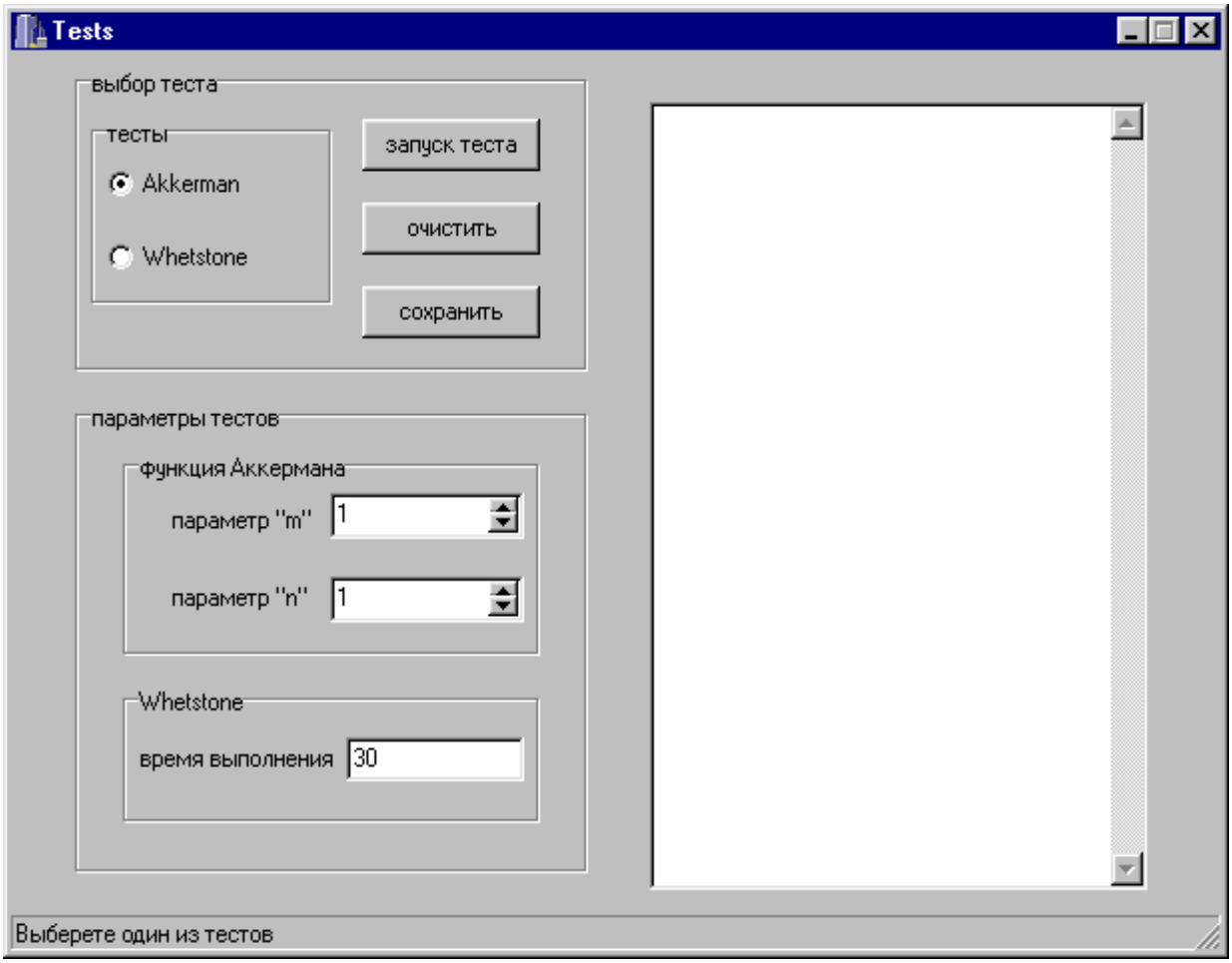

Рис. 2.6. Главная экранная форма программы

Как видно из рисунка 2.6., на главной форме программы присутствуют следующие элементы, реализованные с помощью библиотеки визуальных компонент системы Bulder:

- кнопки “запуск теста”, ”очистить”, ”сохранить” на основе компоненты TButton
- контейнеры, объединяющие логически связанные группы некоторых компонентов, на основе компоненты TgroupBox

- экран сообщений на основе TMemo
- окна редактируемого ввода на основе компоненты TEdit
- спаренные кнопки для установки целочисленного значения – компонента TCSpinEdit

- контейнер для отображения статусной информации - TStatusBar
- стандартный набор кнопок приложений Windows: “свернуть”, “развернуть”, “закрыть”, имеющихся во всех объектах класса TForm

Экран сообщений заполняется по мере выполнения различных действий и хранит в себе все сообщения, выданные за время работы программы. Для удобного просмотра сообщений можно использовать полосу вертикальной прокрутки.

При запуске программы, внизу оконной формы появляется подсказка– указание пользователю: «выберите один из тестов», т.е. пользователю предлагается выбрать один из тестов который он хочет использовать для тестирования системы. Нужный тест указывается в окне «тесты» с помощью кнопок переключения. Для выбора доступно два теста: тест, на базе вычисления функции Аккермана и Whetstone. После выбора, пользователю необходимо задать параметры этого теста. Для задания параметров используется специальное окно: «параметры теста», в нем задаются следующие значения: два целых неотрицательных числа – параметры функции Аккермана, и время выполнения – для Whetstone.

Проделав описанные выше действия, можно переходить к запуску тестов. Для запуска процесса тестирования служит кнопка “запуск теста”. При ее нажатии, программа анализирует выбранный тест и запускает его на выполнение, результаты тестирования выводятся на экран сообщений. Результаты работы тестов продемонстрированы на рисунке 2.7.

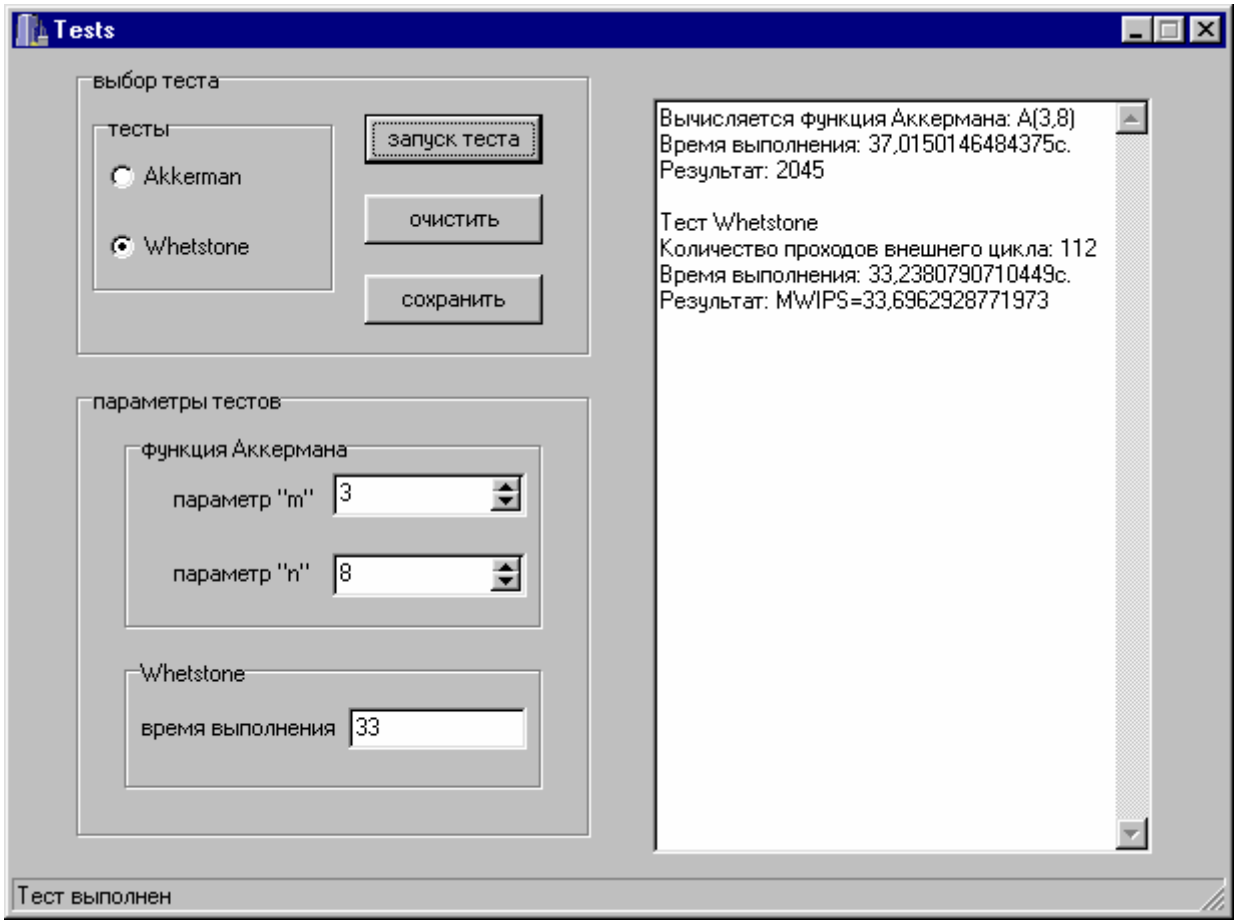

Рис. 2.7. Результаты работы тестирующей программы

На приведенном выше рисунке видно, что последовательно были запущены два теста: вначале функция Аккермана с параметрами A(3,8), а затем Whetstone для которого время выполнения было задано равным тридцати трем секундам. Результатом выполнения первого теста являются следующие полученные данные: время вычисления функции Аккермана, выраженное в секундах и число – ответ, полученный при вычислении функции. Результатом выполнения теста Whetstone является оценка производительности данной вычислительной системы, выраженная в специальных единицах - MWIPS. Подробнее об этом показателе производительности написано в второй главе, в первом параграфе.

После того как пользователь провел все необходимые эксперименты, результаты тестирования можно записать в файл. Для этой цели служит кнопка “сохранить”, при ее нажатии, на экран выводится вторая оконная форма, приведенная на рисунке 2.8.

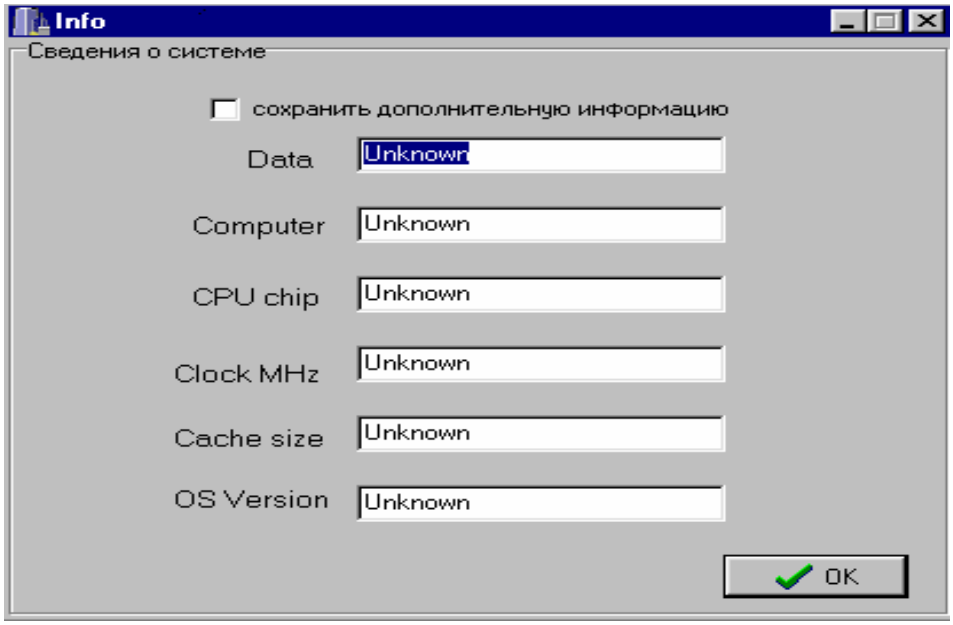

Рис. 2.8. Вспомогательная форма программы

При сохранении результатов тестирования пользователь решает: надо ли ему помещать в файл результатов описание тестируемой системы. Если этот вопрос решается положительно, то пользователю необходимо заполнить следующие информационные поля:

- Data – дата тестирования (например 01.01.02)
- Computer – какие-либо дополнительные сведения о компьютере
- CPU Chip – тип тестируемого процессора
- Clock MHz – тактовая частота процессора
- Cache size – объем кэш памяти
- OS Version – операционная система, поставленная на тестируемом компьютере

После заполнения полей необходимо поставить галку в окне “сохранить дополнительную информацию”, если же галку не поставить, это будет означать отказ пользователя от записи в файл дополнительной информации. При нажатии на кнопку “ОК” пользователю будет предложено указать файл в который будет производиться запись, после чего и произойдет запись в файл результатов тестирования. После этого работа программы

будет закончена и пользователь сможет приступать к проведению нового цикла тестирования.

Текст программы приведен в пункте 2 приложения, структурная схема программы представлена на чертежах 1, 2.

3. Тестирование компьютерных конфигураций

3.1. Описание условий проведения экспериментов

Разработанный пакет тестовых программ использовался для анализа производительности последовательности компьютерных конфигураций. Последовательно, из различных комплектующих, собиралась вычислительная система и на ней производилась серия запусков тестового пакета.

Каждая компьютерная конфигурация тестировалась тремя программами: двумя стандартными тестами производительности вычислительной системы: Whetstone и Dhrystone и тестом, разработанным в рамках дипломного проекта, основанным на вычислении функции Аккермана.

Компьютерные комплектующие были условно разбиты на две группы:

- не изменяемые
- изменяемые

К первой группе относятся те комплектующие, которые не влияют на производительность компьютера при решении инженерных (вычислительных) задач: жесткий диск, видео карта и различные периферийные устройства. Эти комплектующие при переходе от одной компьютерной конфигурации к другой, оставались неизменными.

Ко второй группе отнесли те комплектующие, замена которых может существенно сказаться на производительности системы. Что же необходимо отнести ко второй группе? В первую очередь это, несомненно, процессор – частота работы процессора является одним из определяющих параметров производительности всей системы. Но производительность компьютера

определяется не только быстродействием процессора и объемом оперативной памяти, но в значительной мере зависит от возможностей остальных узлов и подсистем. Прежде всего, производительность компьютера зависит от возможностей материнской платы – основной системной платы, объединяющей процессор, чипсет и все остальные электронные компоненты. От качества элементов, входящих в состав материнской платы, от тщательности проработки ее архитектуры и качества изготовления этой платы зависят в дальнейшем производительность и потенциальные возможности всей системы компьютера. Не секрет, что разные материнские платы, ориентированные на один и тот же сектор рынка, могут отличаться по своей производительности на десятки процентов, что не редко значительно превышает разницу в производительности соседних выпусков процессоров [8].

Исходя из вышесказанного, была разработана схема эксперимента: выбиралась и фиксировалась определенная материнская плата, при этом последовательно менялся процессор, причем для каждого набора материнская плата – процессор изменялся объем оперативной памяти. После тестирования всех исследуемых процессоров на данной платформе, материнская плата менялась, и процесс тестирования начинался заново.

Перечень тестируемых комплектующих

В приведённой ниже таблице представлены основные характеристики исследуемых микросхем оперативной памяти.

Таблица 3.1.

Микросхемы оперативной памяти

объем памяти, Мб	рабочая частота, MHz
64	100
128	100
256	100

В таблице 3.2. представлены основные параметры исследуемых процессоров.

Таблица 3.2.

Параметры исследуемых процессоров

Название	Рабочая частота процессора, MHz	частота шины, MHz	объем кэша, Кб
IP Celeron	733	66	128
IP Celeron	800	100	128
IP Celeron	900	100	128
IP Celeron	1000	100	128
Intel PentiumIII	733	133	256

Материнские платы на основе которых собирались тестируемые компьютерные конфигурации:

- Производитель материнской платы - ACORP, chipset – VIA 693A(Apollo Pro133)
- Производитель материнской платы - ACORP, chipset – VIA 694(Apollo Pro133A)
- Производитель материнской платы - EPOX, название материнской платы – 3PTA, chipset – Intel 815EP.

3.2. Результаты тестирования

Представление результатов тестирования разбиты на параграфы: в первых трех будут отображены результаты, полученные на представленных выше, материнских платах, а в четвертом пункте эти платформы будут сравниваться между собой.

3.2.1. Результаты тестирования компьютерных конфигураций на платформе ACORP, чипсет VIA 693A

В приведённой ниже таблице представлены результаты тестирования различных процессоров на стандартных тестах производительности Whetstone и Dhrystone.

Таблица 3.3.

Результаты тестирования процессоров
на тестах Whetstone и Dhrystone

Объем оперативной памяти, Мб	Частота,Mhz	Прирост, %	Рейтинг, VAX MIPS rat.	Прирост, %	Рейтинг, MWIPS	Прирост, %
64	733	0	339,79	0	266,03	0
256	733	0	339,79	0	266,03	0
64	800	9,14	370,93	9,16	290,45	9,18
64	900	12,50	417,25	12,49	326,62	12,45
64	1000	11,11	463,43	11,07	363,43	11,27

В двух верхних строчках таблицы приведены результаты тестирования конфигураций с одинаковым процессором (IP Celeron 733MHz), но с различными объемами оперативной памяти.

По данным, представленным в таблице 3.3., были построены графики, приведенные на рисунке 3.1.

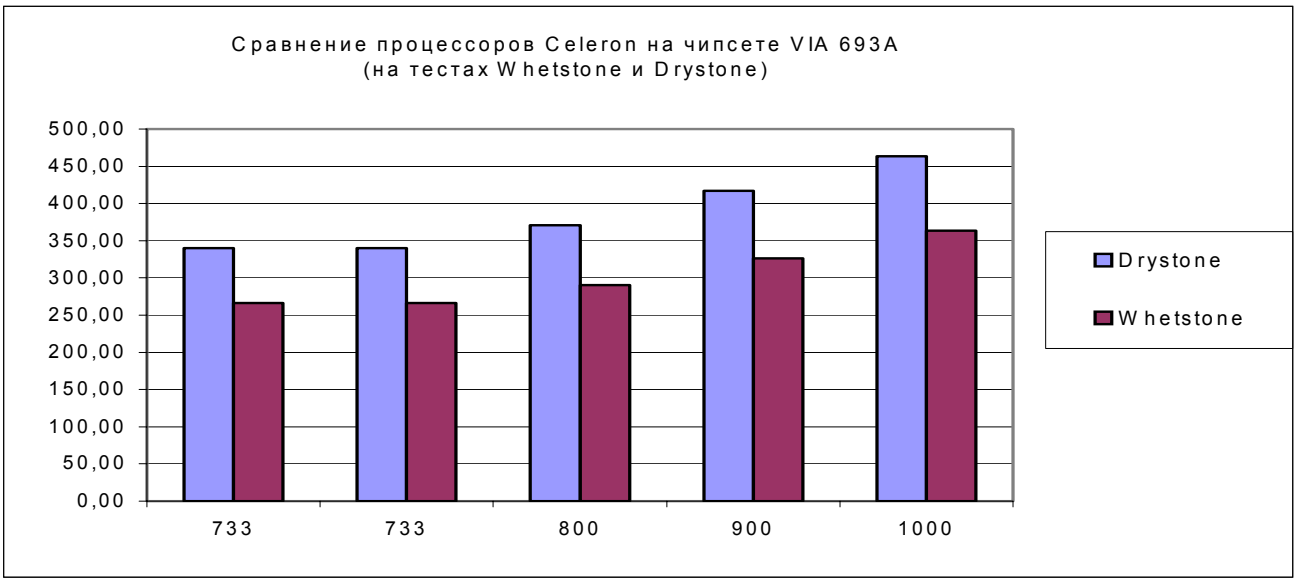

Рис.3.1.

Из графика, представленного выше, видно, что производительность системы растет пропорционально увеличению тактовой частоты процессора, причем коэффициент пропорциональности приблизительно равен единице. При этом увеличение объема оперативной памяти в четыре раза не отразилось на полученных оценках, т.е. вычислительные системы с одинаковым процессором, но разными объемами оперативной памяти получили на тестах Whetstone и Dhrystone одинаковые оценки.

В таблице 3.4. представлены результаты тестирования процессоров на тесте, основанном на базе вычисления функции Аккермана.

Таблица 3.4.

Результаты тестирования процессоров
на базе вычисления функции Аккермана

Процессоры класса Celeron, Mhz	**733**					**800**				
Параметр функции Аккермана	7	8	9	10	11	7	8	9	10	11
Время выполнения теста, сек	1,75	6,1	24,99	106,06	441,1	1,65	5,39	21,64	89,91	370,81
Процессоры класса Celeron, Mhz	**900**					**1000**				
Параметр функции Аккермана	7	8	9	10	11	7	8	9	10	11
Время выполнения теста, сек	1,54	4,83	19,44	81,51	337,9	1,32	4,45	17,8	74,75	310,44

По данным, представленным в таблице 3.4., были построены графики, приведенные на рисунке 3.2.

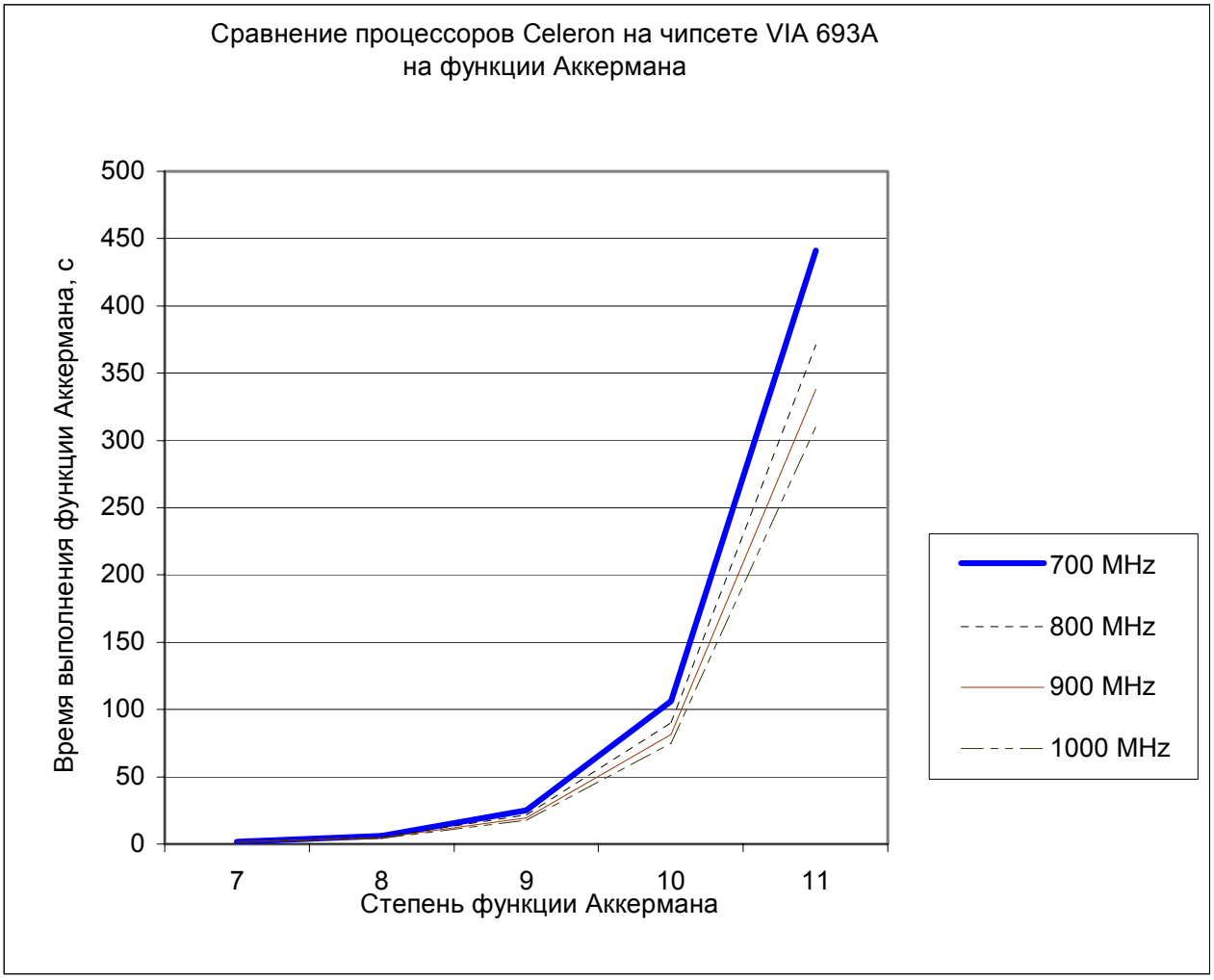

Рис. 3.2.

Анализируя построенный график, можно сделать выводы, что результаты сравнения компьютерных конфигураций, полученные при использовании программы вычисления функции Аккермана очень близки к результатам, полученным стандартными тестами.

На данной платформе был проведен эксперимент по сравнению следующих процессоров: IP Celeron 733MHz и IPentiumIII-733MHz. Эти процессоры различаются объемом кэш памяти, а также частотой работы шины. Результаты, полученные на стандартных тестах, приведены в таблице 3.5.

Таблица 3.5.

Результаты сравнения процессоров IPIII-733 с Celeron-733 на стандартных тестах

Процессоры, Mhz	hrystone, VAX MIPS rat.	hetstone, MWIPS
Celeron-3 733	339,79	266,03
Celeron-3 800	370,93	290,45
Pentium3 733	337,90	265,17

Графически, результаты сравнения приведены на рисунке 3.3.

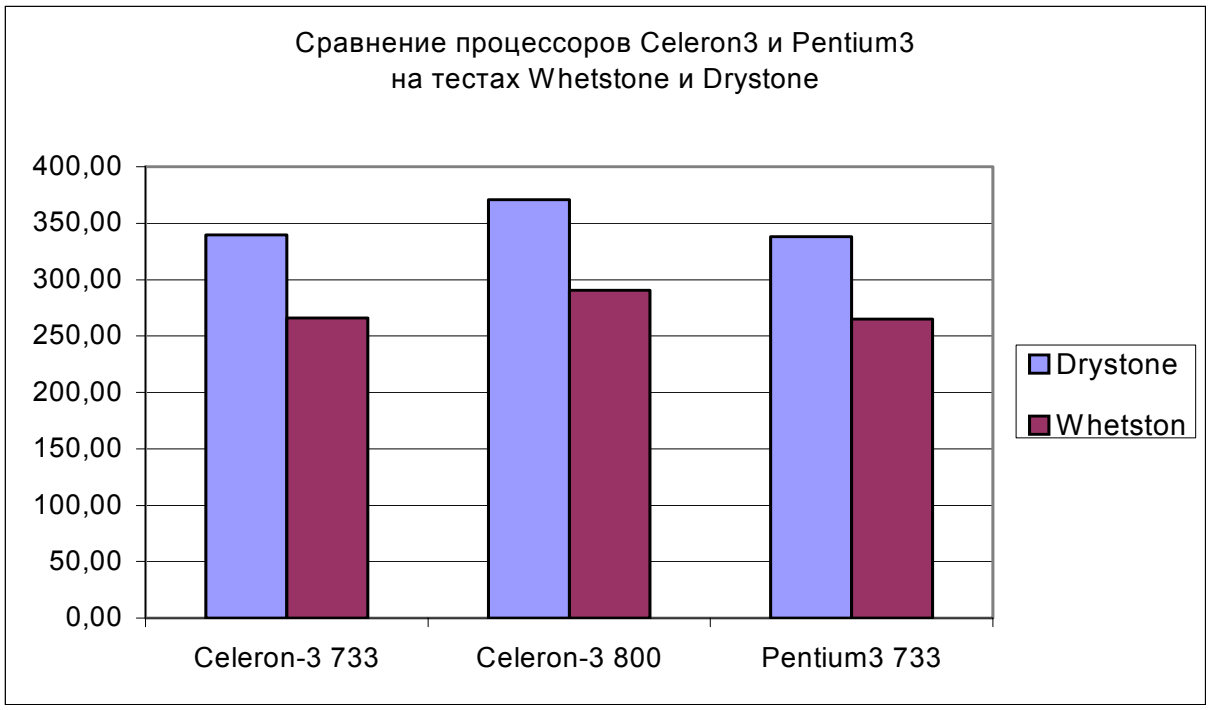

Рис. 3.3.

Результаты, полученные на тесте, основанном на вычислении функции Аккермана, представлены в таблице 3.6.

Таблица 3.6.

Результаты сравнения процессоров IPIII-733 и Celeron-733 на базе вычисления функции Аккермана

Процессоры, Mhz	Celeron-3 733					Pentium-3 733				
Параметр функции Аккермана	7	8	9	10	11	7	8	9	10	11
Время выполнения теста, сек	1,75	6,1	24,99	106,06	441,1	1,76	5,65	21,75	89,59	371,52

Графически, результаты сравнения приведены на рисунке 3.4.

Рис. 3.4.

По результатам проведенного сравнения можно отметить следующую деталь: стандартные тесты измерения производительности «не заметили» разницу между PentiumIII-733MHz и CeleronIII-733MHz. А по результатам тестирования разработанной программой, можно отметить, что PentiumIII-733MHz не только обогнал CeleronIII-733MHz, но и получил оценку производительности на уровне CeleronIII-800MHz.

3.2.2. Результаты тестирования компьютерных конфигураций на платформе ACORP, чипсет VIA 694

Алгоритм проведения экспериментов на данной материнской плате совпадает с алгоритмом, описанным в предыдущем параграфе.

В приведённой ниже таблице представлены результаты тестирования различных процессоров на стандартных тестах производительности Whetstone и Dhrystone.

Таблица 3.7.

Результаты тестирования процессоров
на тестах Whetstone и Dhrystone

Процессоры класса Celeron		Dhrystone		Whetstone	
Частота,Mhz	Прирост, %	Рейтинг, VAX MIPS rat.	Прирост, %	Рейтинг, MWIPS	Прирост, %
733	0,00	339,79	0,00	266,03	0,00
800	9,14	370,93	9,16	290,45	9,18
900	12,50	417,25	12,49	326,62	12,45
1000	11,11	463,43	11,07	363,43	11,27

По данным, представленным в таблице 3.7., были построены графики, приведенные на рисунке 3.5.

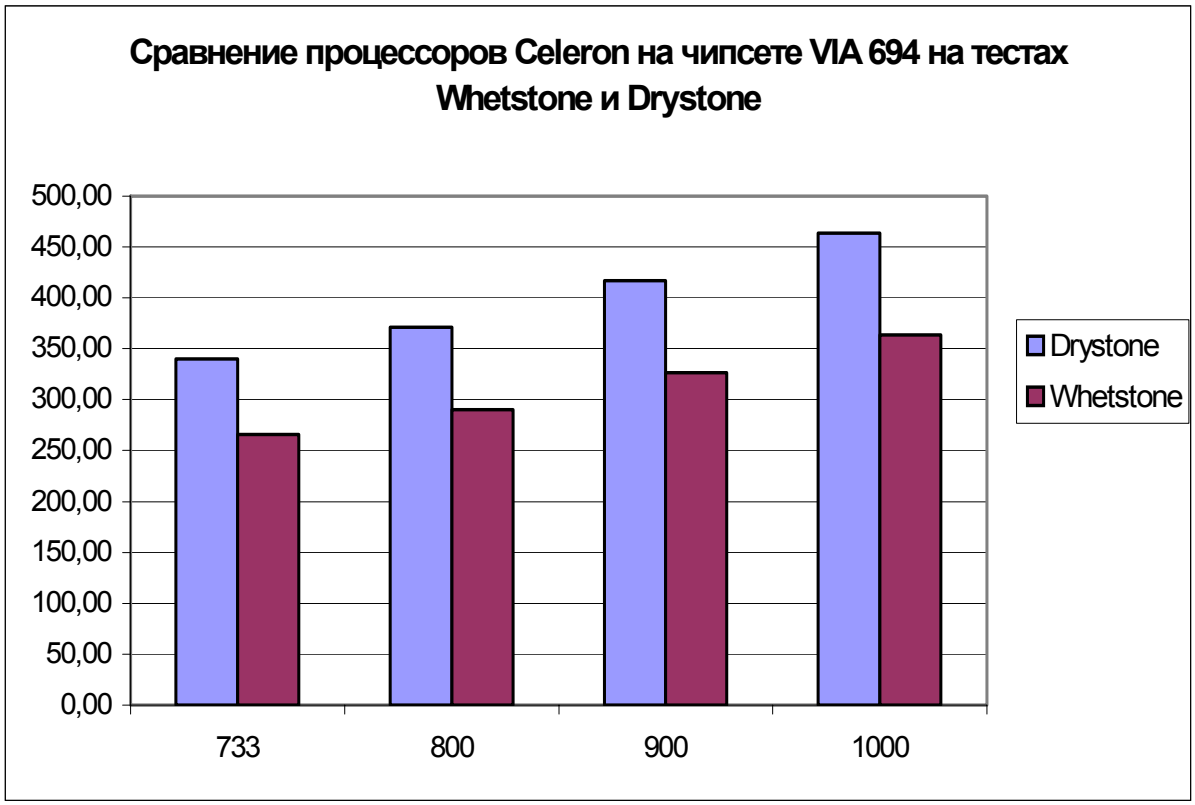

Рис. 3.5.

В таблице 3.8. представлены результаты тестирования процессоров на тесте, основанном на базе вычисления функции Аккермана.

Таблица 3.8.

Результаты тестирования процессоров
на базе вычисления функции Аккермана

Процессоры класса Celeron, Mhz	**733**					**800**				
Параметр функции Аккермана	7	8	9	10	11	7	8	9	10	11
Время выполнения теста, сек	1,76	5,82	22,85	93,65	382,45	1,7	5,28	20,59	83,84	339,55
Процессоры класса Celeron, Mhz	**900**					**1000**				
Параметр функции Аккермана	7	8	9	10	11	7	8	9	10	11
Время выполнения теста, сек	1,48	4,77	18,57	75,31	306,98	1,43	4,4	16,86	68,93	281,66

По данным, представленным в таблице 3.8., были построены графики, приведенные на рисунке 3.6.

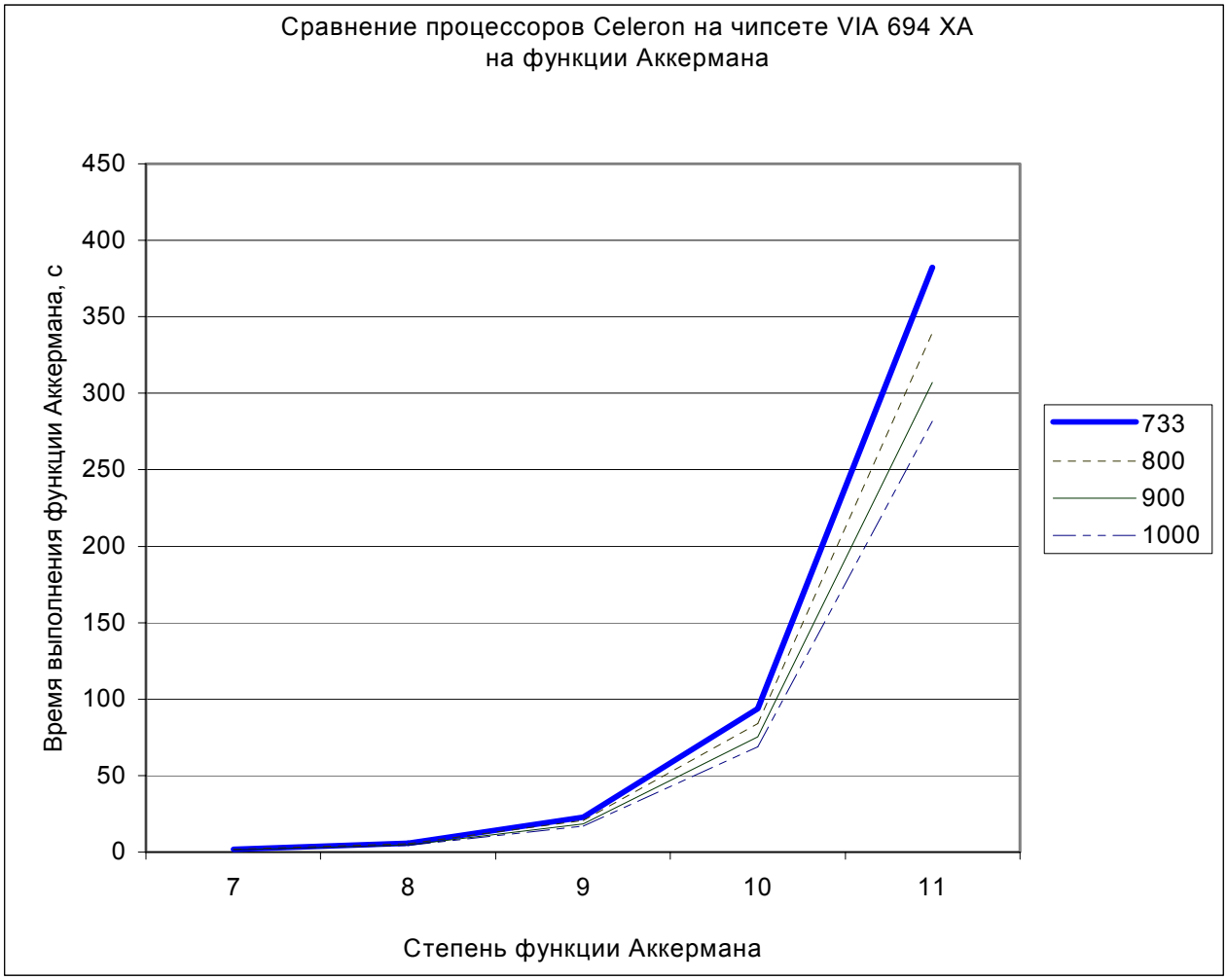

Рис. 3.6.

Общие выводы которые можно сделать анализируя полученные результаты: с ростом частоты работы процессора прямо пропорционально увеличивается и общая производительность системы. Коэффициент пропорциональности здесь также примерно равен единице, это факт показывают как стандартные тесты измерения производительности, так и разработанный тест, на основе вычисления функции Аккермана.

3.2.3. Результаты тестирования компьютерных конфигураций на платформе EPOX 3PTA, чипсет – Intel 815EP

Алгоритм проведения экспериментов на данной материнской плате не меняется.

В приведённой ниже таблице представлены результаты тестирования различных процессоров на стандартных тестах производительности Whetstone и Dhrystone.

Таблица 3.9.

Результаты тестирования процессоров
на тестах Whetstone и Dhrystone

Процессоры класса Celeron		Dhrystone		Whetstone	
Частота,Mhz	Прирост, %	Рейтинг, VAX MIPS rat.	Прирост, %	Рейтинг, MWIPS	Прирост, %
733	0,00	338,53	0,00	265,13	0,00
800	9,14	372,07	9,91	291,53	9,96
1000	25,00	465,21	25,03	364,18	24,92

По данным, представленным в таблице 3.9., были построены графики, приведенные на рисунке 3.6.

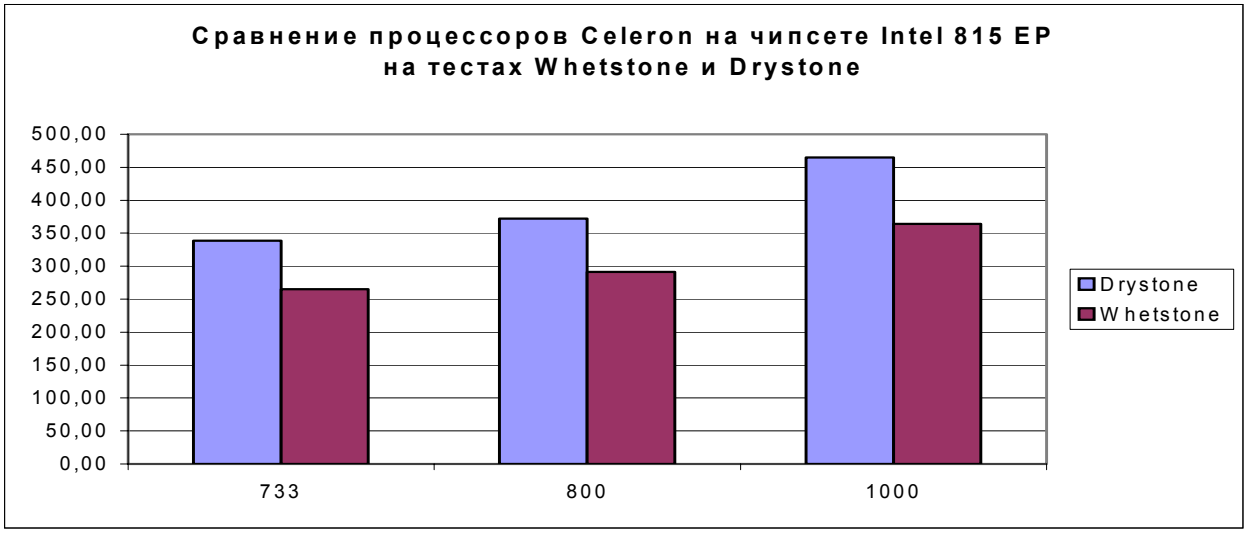

Рис. 3.6.

В таблице 3.10. представлены результаты тестирования процессоров на тесте, основанном на базе вычисления функции Аккермана.

Таблица 3.10.

Результаты тестирования процессоров
на базе вычисления функции Аккермана

Процессоры класса Celeron, Mhz	733	800

Параметр функции Аккермана	7	8	9	10	11	7	8	9	10	11
Время выполнения теста, сек	1,81	5,77	22,95	94,58	387,83	1,7	5,33	20,81	85,03	347,4
Процессоры класса Celeron, Mhz	**1000**									
Параметр функции Аккермана	7	8	9	10	11					
Время выполнения теста, сек	1,48	4,34	16,86	68,98	282,32					

По данным, представленным в таблице 3.10., были построены графики, приведенные на рисунке 3.7.

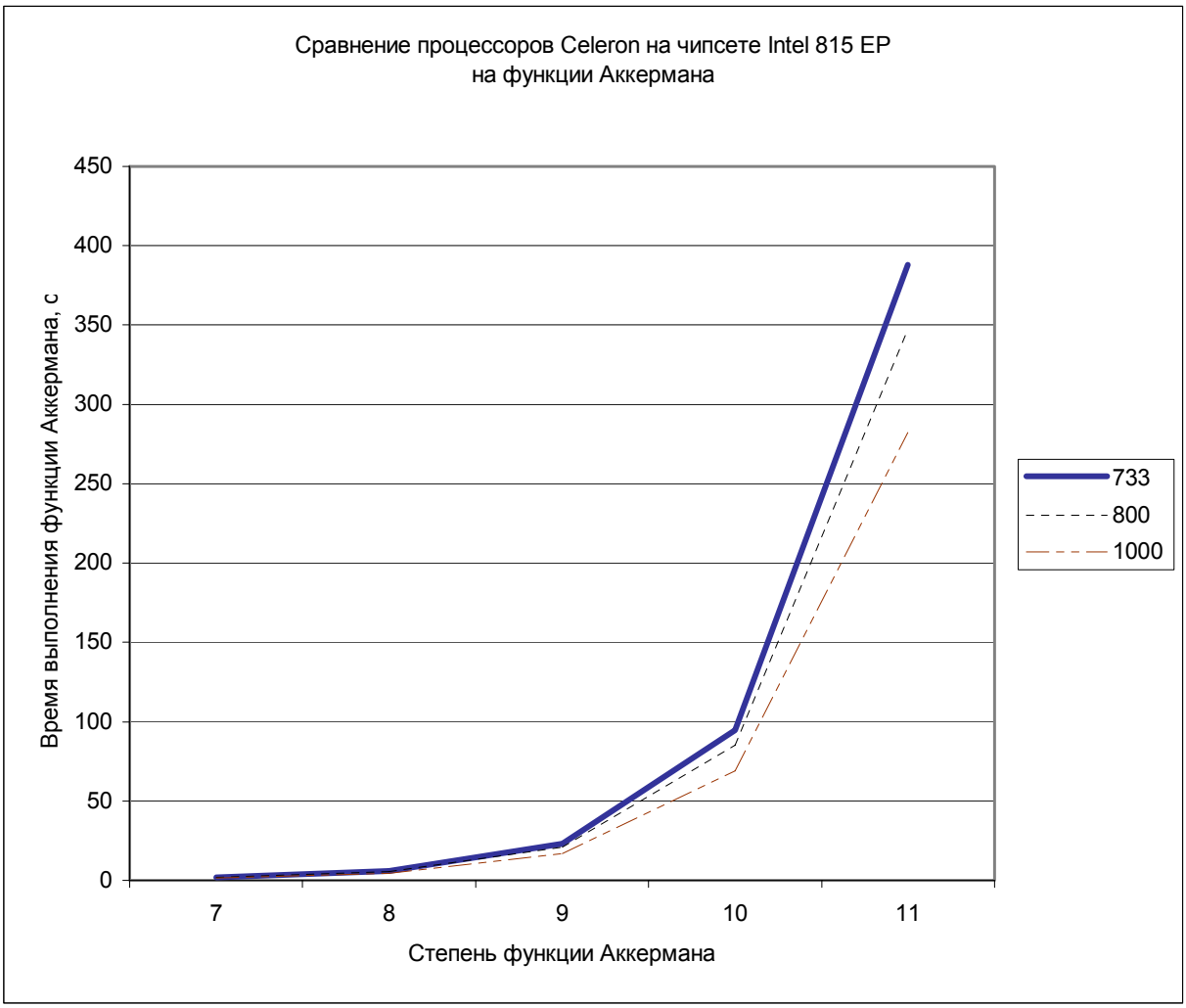

Рис.3.7.

Общие выводы по результатам тестирования совпадают с выводами, полученными ранее в предыдущих параграфах, поэтому в этом параграфе кратко приведены только результаты тестирования.

3.2.4. Сравнение производительности процессоров на различных платформах

В этом параграфе приводятся результаты сравнения производительности процессоров полученных на различных платформах, тем самым проводится сравнение самих материнских плат.

В таблице 3.11. представлены результаты сравнения материнских плат тестом Whetstone.

Таблица 3.11.

Результаты сравнения материнских плат тестом Whetstone

Серия чипсетов	Процессоры класса Celeron, MHz		
	733	800	1000
VIA 693A	266,03	290,45	363,43
VIA 694	266,27	290,65	363,88
Intel 815 EP	265,13	291,53	364,18

Графически, результаты сравнения приведены на рисунке 3.8.

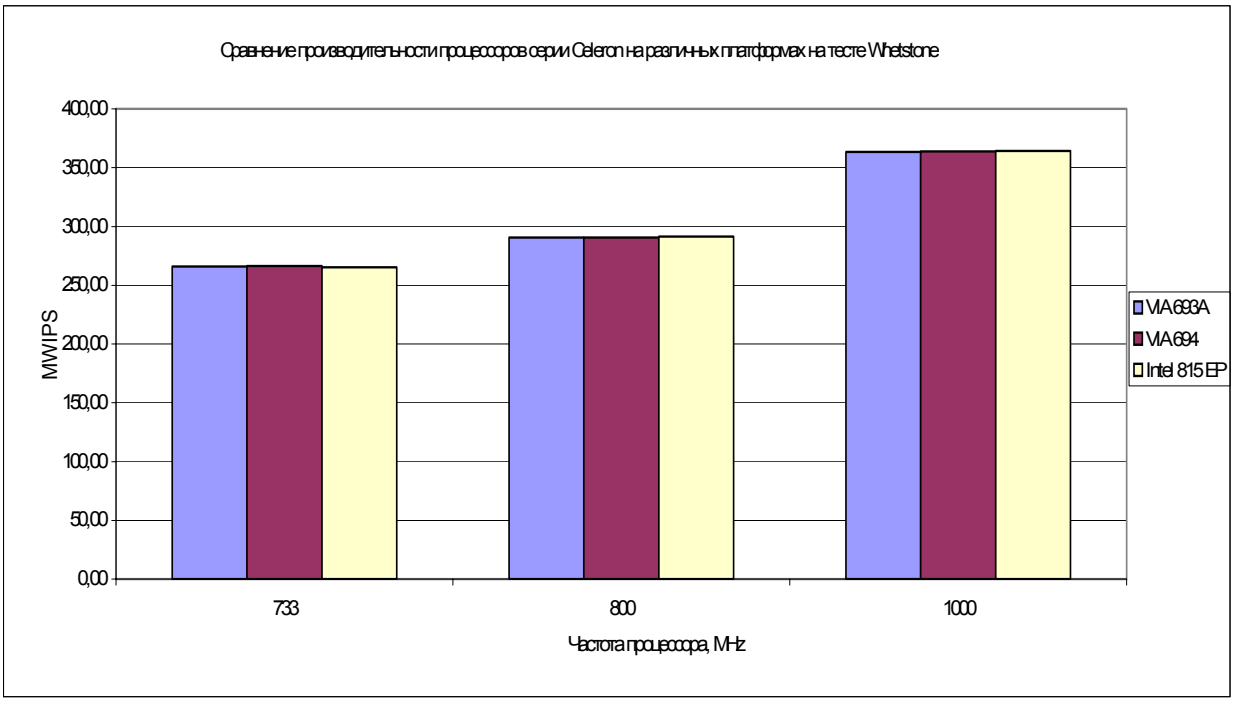

Рис 3.8.

На построенном графике видно, что процессоры, работающие на одной тактовой частоте, показали на различных материнских платах одинаковые результаты на тесте Whetstone.

В таблице 3.12. представлены результаты сравнения материнских плат тестом Dhrystone.

Таблица 3.12.

Результаты сравнения материнских плат

тестом Dhrystone

Серия чипсетов	Процессоры класса Celeron, MHz		
	733	800	1000
VIA 693A	338,65	370,93	463,43
VIA 694	339,79	370,34	464,02
Intel 815 EP	338,53	372,07	465,21

Графически, результаты сравнения приведены на рисунке 3.9.

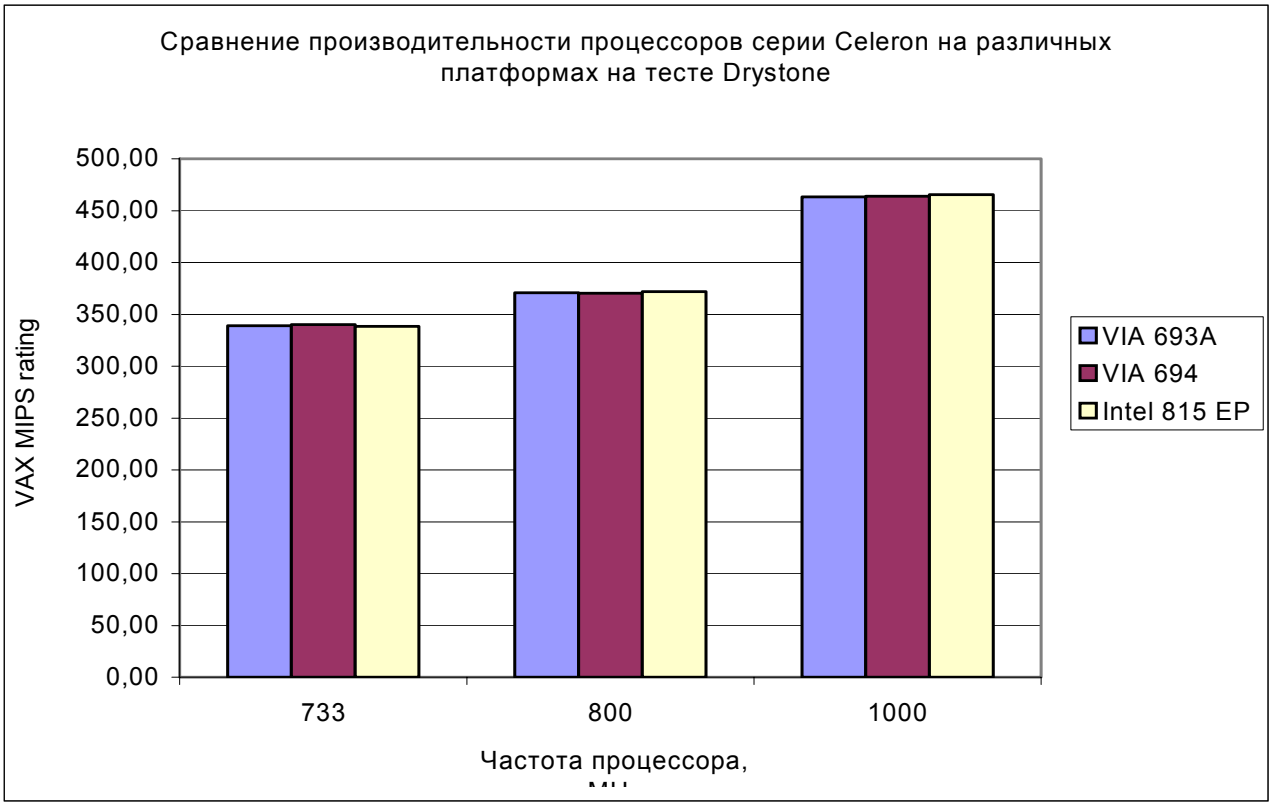

Рис. 3.9.

Очевидно, что с помощью стандартного теста Dhrystone, сравнить различные вычислительные платформы, также не удастся, т.к. данный тест «не замечает» смены платформы.

В таблице 3.13. представлены результаты сравнения материнских плат тестом Whetstone.

Таблица 3.13.

Результаты сравнения материнских плат тестом, основанным на базе вычисления функции Аккермана

Серия чипсетов	Процессоры класса Celeron, MHz		
	733	800	1000
VIA 693A	441,10	370,81	310,44
VIA 694	382,45	339,55	281,66
Intel 815 EP	387,83	347,40	289,32

Графически, результаты сравнения приведены на рисунке 3.10.

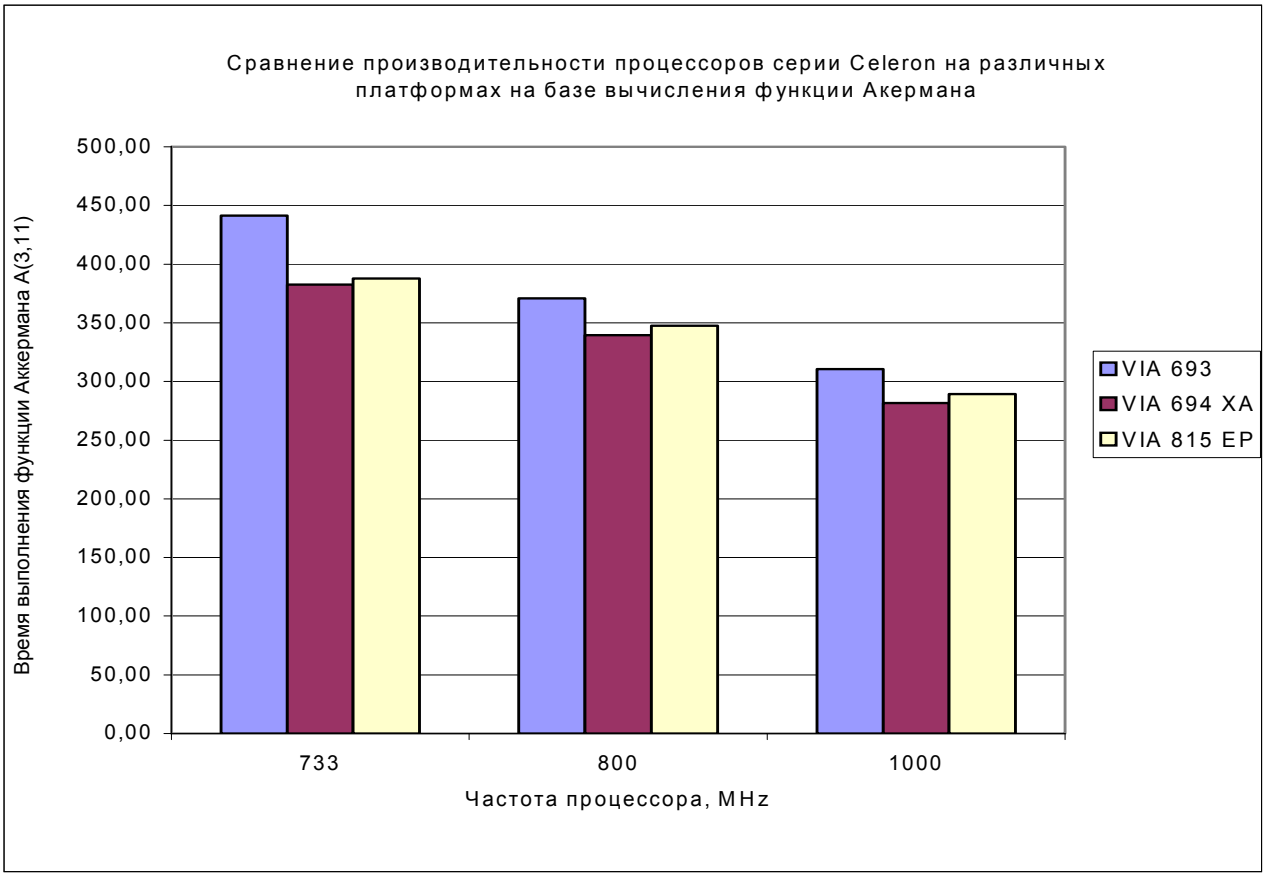

Рис. 3.10.

Анализируя построенный график, делаем вывод, что с помощью разработанного теста на базе вычисления функции Аккермана можно сравнивать между собой производительность различных материнских плат. В частности, по результатам тестирования можно утверждать, что из рассматриваемых платформ, наиболее производительной, является материнская плата фирмы ACORP на чипсете VIA 694, а самой “медленной” - плата этой же фирмы на чипсете VIA 693A. Вычислительная платформа фирмы EPOX на чипсете Intel 815EP занимает промежуточное положение.

3.3. Анализ результатов тестирования

Основной причиной проведения экспериментов является желание выяснить, какие комплектующие вычислительной системы, в какой степени оказывают влияние на производительность вычислительной системы в целом. По полученным результатам можно сделать следующие выводы:

- Увеличение объема оперативной памяти не отразилось на результатах тестирования всеми тремя тестами. Данный факт объясняется следующими причинами: во всех рассматриваемых тестах нет громоздких промежуточных результатов и поэтому, для выполнения тестов достаточно и минимального использованного объема оперативной памяти – 64Мб, и дальнейшее наращивание объема уже не приводит к приросту производительности.
- С помощью стандартных тестов Whetstone и Dhrystone можно объективно сравнивать производительности вычислительных систем собранных на одной платформе. При этом можно отметить, что прирост производительности всей системы и прирост тактовой частоты процессора связаны прямо пропорциональным законом, причем коэффициент пропорциональности примерно равен единице. При смене же платформы, вычислительные системы, обладающие идентичными процессорами, показывают на этих тестах одинаковые результаты, что не позволяет сравнить между собой производительность различных материнских плат с помощью тестов Whetstone и Dhrystone.
- Результаты, полученные при тестировании компьютерных конфигураций разработанным тестом, основанным на базе вычисления функции Аккермана, в рамках одной платформы близки к результатам, полученным другими тестами. Но этот тест также позволяет сравнивать производительность материнских плат, что и было продемонстрировано в этой главе во втором параграфе, также этот тест позволяет оценить вклад быстродействия платформы в общую производительность системы.

Таким образом, используя описанный тестовый пакет, пользователь сможет оценить производительность различных комплектующих вычислительной системы и выбрать наиболее подходящие для решения поставленных задач.