МЦСТ ЭЛЬБРУС ИНЭУМ

Микропроцессоры с высокой скоростью исполнения однопоточных приложений

Александр Ким, Владимир Волконский, Федор Груздов,
Мурад Нейман-заде, Сергей Семенихин, Михаил Слесарев
ОАО «ИНЭУМ им. И.С. Брука», ЗАО «МЦСТ»

IV Московский Суперкомпьютерный Форум
23.10.2013

МЦСТ
ЭЛЬБРУС
ИНЭУМ

Современные мировые тенденции развития ВТ и ИТ

- Экспоненциальный рост числа транзисторов в микропроцессорах
  - Массовое использование параллелизма вычислений
- Повышение энергетической эффективности микропроцессоров
- Экспоненциальный рост объема и сложности программного обеспечения
  - Требуется существенное повышение надежности и безопасности
- Накопленный объем ПО требует решения проблемы совместимости

МЦСТ ЭЛЬБРУС ИНЭУМ

Технологии, разработанные в процессе создания МП «Эльбрус»

Высокопроизводительная энергоэффективная параллельная архитектура ядра микропроцессора на базе «широкого командного слова», выполняет до 30 операций за такт

- исполнение планируется оптимизирующим компилятором, число операций за такт (ЧОТ) на пакете SPECcpu2000 больше 3, планируется довести до 6 (Intel Core 2 – 1,1)

Защищенные вычисления, основанные на контекстной защите памяти на базе тегированной архитектуры

- обеспечивают стойкость к компьютерным вирусам и быструю отладку программ

Аппаратно поддержанная двоичная трансляция

- обеспечивает совместимость с другими платформами на уровне исполняемых кодов

МЦСТ
ЭЛЬБРУС
ИНЭУМ

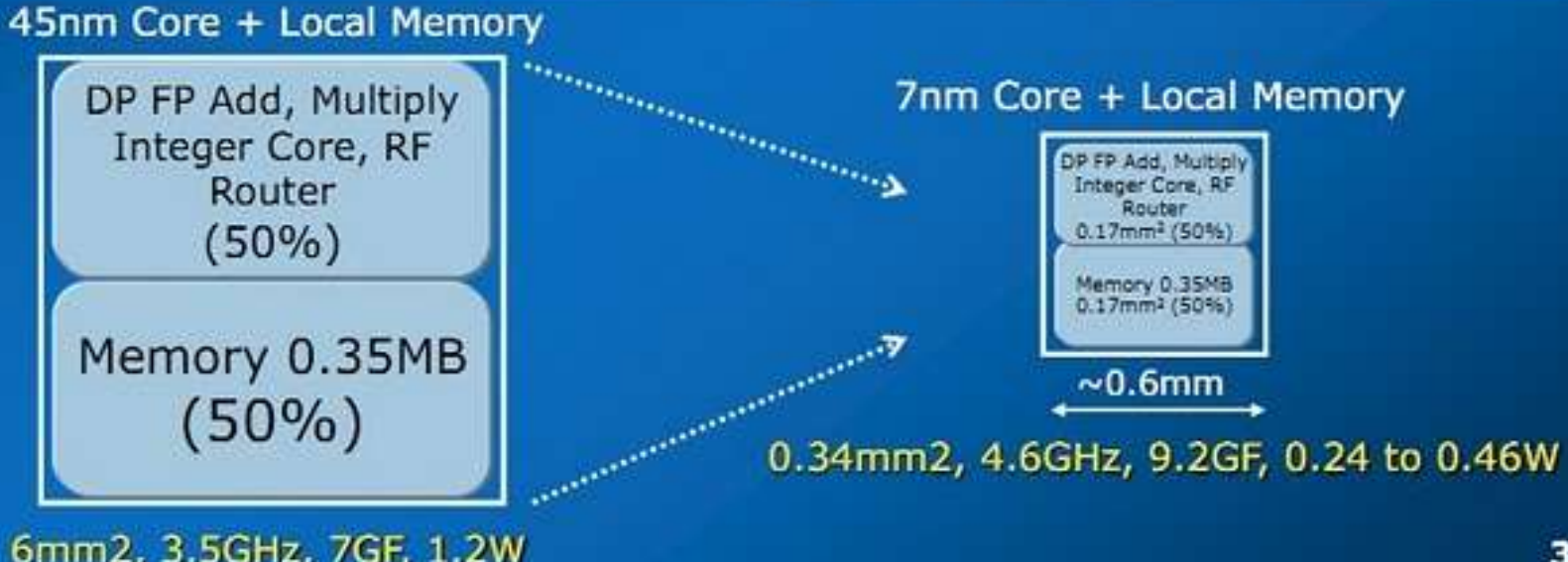
Достижение высокой логической скорости и экономного потребления энергии на базе МП линии «Эльбрус»

МЦСТ ЭЛЬБРУС ИНЭУМ

Развитие технологии

Technology Outlook

Technology (High Volume)	45nm (2008)	32nm (2010)	22nm (2012)	14nm (2014)	10nm (2016)	7nm (2018)	5nm (2020)
Transistor density	1.75	1.75	1.75	1.75	1.75	1.75	1.75
Frequency scaling	15%	10%	8%	5%	4%	3%	2%
Vdd scaling	-10%	-7.5%	-5%	-2.5%	-1.5%	-1%	-0.5%
Dimension & Capacitance	0.75	0.75	0.75	0.75	0.75	0.75	0.75
SD Leakage scaling/micron	1X Optimistic to 1.43X Pessimistic						



МЦСТ ЭЛЬБРУС ИНЭУМ

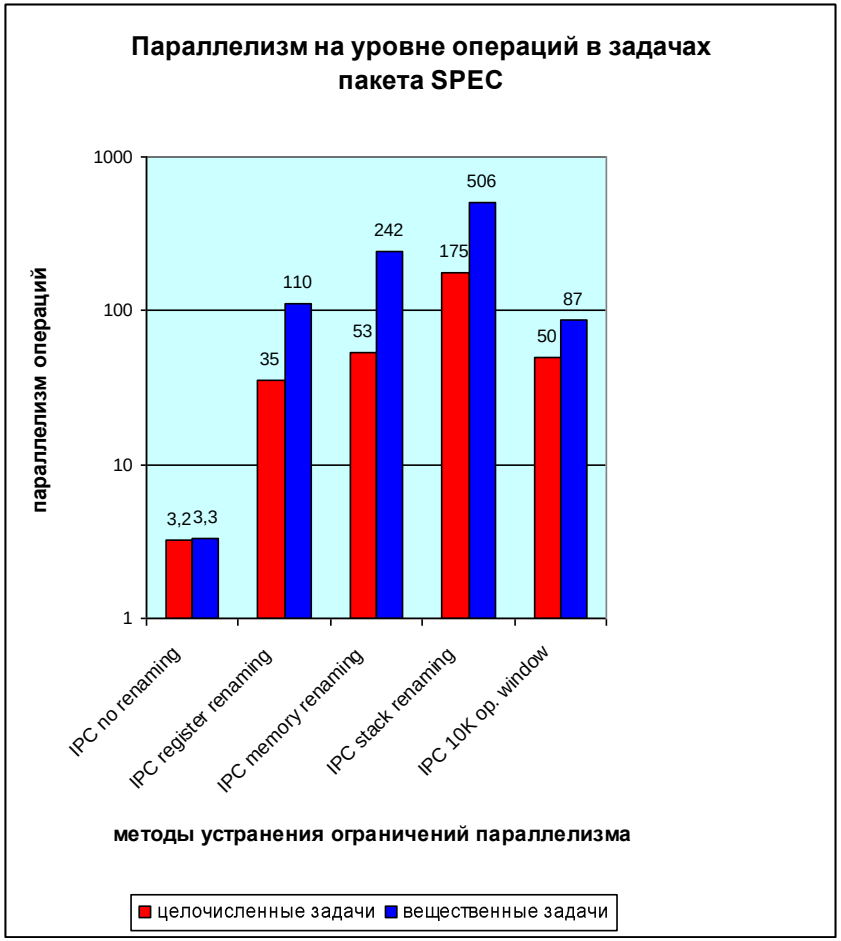
Почему важна производительность ядра

- Экспоненциально растущие транзисторы вкладываются в аппаратный параллелизм
  - Параллелизм ядра ограничен многими факторами
  - Самое простое решение – много процессорных ядер
- Параллелизм аппаратуры необходимо использовать в алгоритмах
  - Языки программирования последовательные
  - Языки параллельного программирования почти не приживаются
- Разные уровни параллелизма автоматизируются с разным успехом
  - Параллелизм на уровне операций поддается аппаратной и аппаратно-программной оптимизации, он универсален
  - Векторный параллелизм поддается аппаратно-программной оптимизации, но имеет ограниченное применение
  - Параллелизм потоков управления трудно автоматизируется
    - Требуются усилия программистов по распараллеливанию программ
  - Параллелизм потоков распределенных вычислений почти не автоматизируется
    - Требуется серьезная переработка программ для распараллеливания
- Далеко не все программы удается распараллелить на потоки
- При распараллеливании на потоки остаются последовательные участки
  - Закон Амдала требует наличия мощного ядра для последовательного исполнения

Производительное ядро повышает эффективность параллельных систем

Параллелизм на уровне операций

- Анализ трасс исполнения показывает значительный потенциал параллелизма
  - Целочисленные задачи: 81 - 240 оп./такт
  - Вещественные задачи: 36 - 4003 оп./такт
- Параллелизм ограничен зависимостями по памяти
  - Переименование регистров повышает его с 3,2 / 3,3 до 35 / 110 оп./такт
  - Отказ от переиспользования памяти и стека увеличивает параллелизм в 5 раз до 175 / 506 оп./такт
- Вещественные задачи обладают большим потенциалом параллелизма
  - Доминируют циклы
  - Выше векторный параллелизм
- При снятии ограничений по памяти параллелизм не локален
  - Это свидетельствует о наличии многопоточного параллелизма



Резервы параллелизма операций огромны, их нужно использовать

МЦСТ ЭЛЬБРУС ИНЭУМ

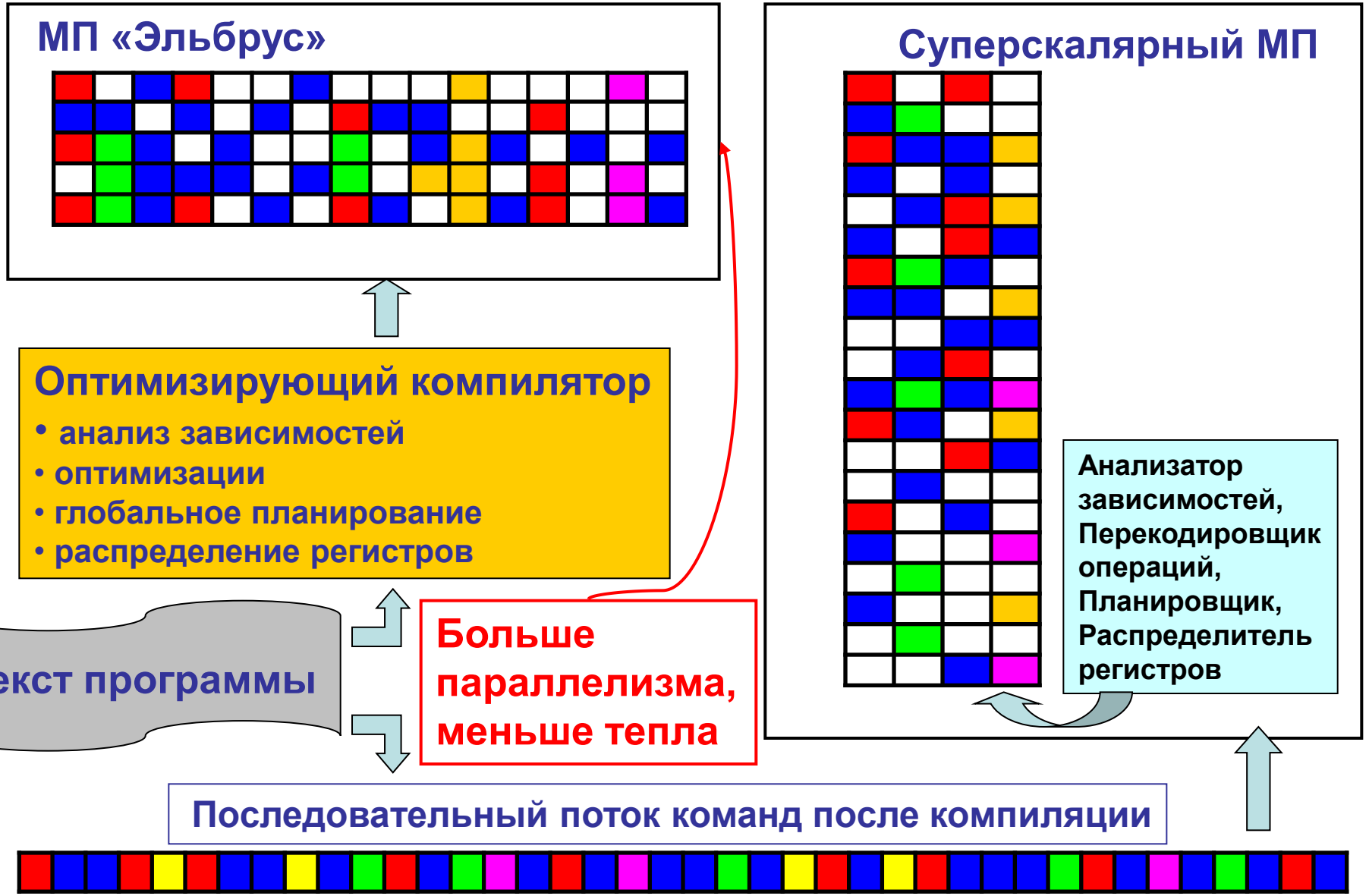
Параллелизм архитектуры «Эльбрус»

- Параллелизм скалярных операций
  - До 30 операций за такт в разрабатываемых МП
  - До 40-50 операций за такт в перспективных МП
- Параллелизм векторных операций
  - Упакованные данные в векторе
    - байтовые – 8, двухбайтовые – 4, 32-разрядные – 2
  - Как минимум, удвоение числа операций в перспективных МП
- Параллелизм потоков управления на общей памяти
  - Поддержка многоядерности
  - Поддержка многопроцессорности
  - Поддержка легкой многопоточности в перспективных МП
- Параллелизм на распределенной памяти
  - Высокоскоростные каналы обмена

Используется с помощью оптимизирующего и распараллеливающего компилятора

Параллелизм МП «Эльбрус» и суперскалярного МП

МЦСТ ЭЛЬБРУС ИНЭУМ



МЦСТ ЭЛЬБРУС ИНЭУМ

Специальные методы оптимизации и распараллеливания

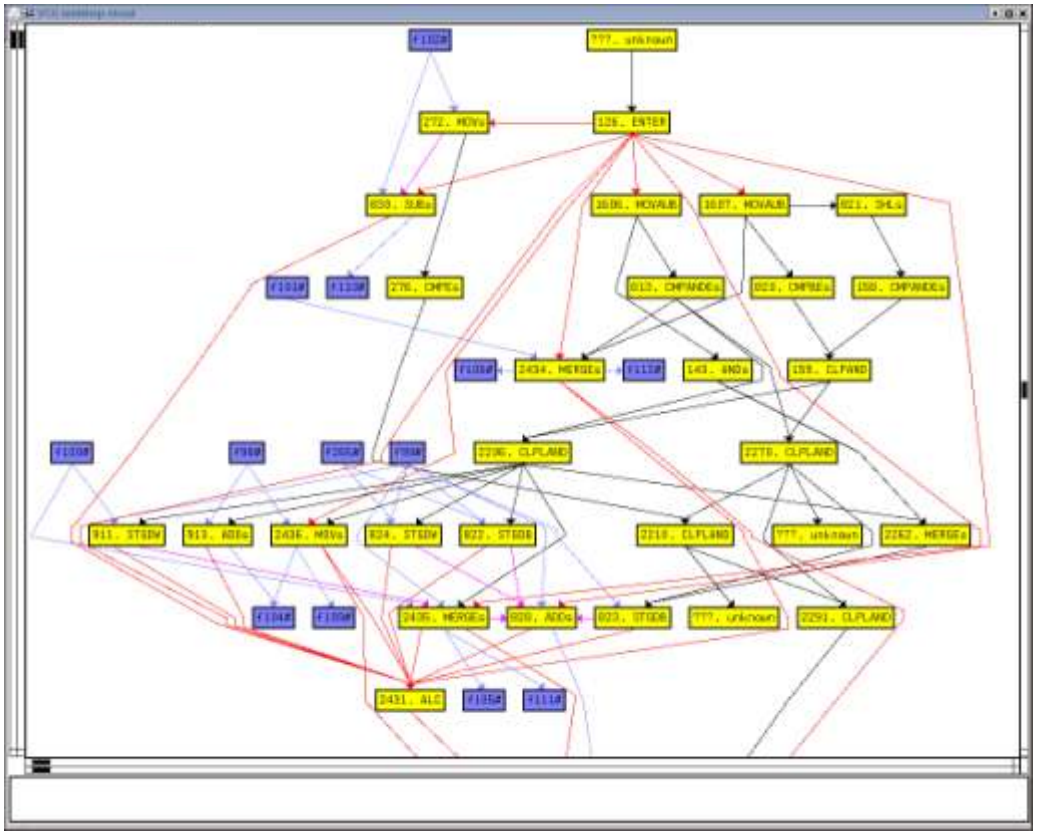
Реализованы в оптимизирующем компиляторе

- Профилирование (статическое и динамическое)
- Классификация регионов оптимизации и выбор соответствующих эвристик
- Учет обратных связей от поведения программы в эвристиках оптимизаций
- Динамический и статический анализ зависимостей
- Межпроцедурный анализ программы с выбором методов распараллеливания
  - Распараллеливание на уровне операций
  - Векторизация и распараллеливание на уровне упакованных операций
  - Распараллеливание на потоки управления
  - Распараллеливание на распределенную память
- Автоматическое распараллеливание на всех уровнях

МЦСТ ЭЛЬБРУС ИНЭУМ

Распараллеливание программ

```
LOCAL void sweep(void)
{
    struct segment *seg;
    NODE *p;
    int n;
    /* empty the free list */
    fnodes = NIL;
    nfree = 0;
    /* add all unmarked nodes */
    for (seg = segs; seg != NULL; seg = seg->sg_next) {
        p = &seg->sg_nodes[0];
        for (n = seg->sg_size; n--; p++)
            if (!(p->n_flags & MARK)) {
                switch (ntype(p)) {
                case STR:
                        if (p->n_strtype == DYNAMIC && p->n_str != NULL) {
                            total -= (long) (strlen(p->n_str)+1);
                            free(p->n_str);
                        }
                        break;
                case FPTR:
                        if (p->n_fp)
                            fclose(p->n_fp);
                        break;
                case VECT:
                        if (p->n_vsize) {
                            total -= (long) (p->n_vsize * sizeof(NODE **));
                            free(p->n_vdata);
                        }
                        break;
                }
                p->n_type = FREE;
                p->n_flags = 0;
                rplaca(p,NIL);
                rplacd(p,fnodes);
                fnodes = p;
                nfree++;
            }
            else
                p->n_flags &= ~(MARK | LEFT);
    }
```



Исходный код

Фрагмент промежуточного представления

```
fc050446 98c11d24 a01dd460 90c00826 9208c106 a229c141 802bd028 001ffffc 61630263 44410000
fe471467 80003000 a026c046 900bcc09 260bc822 8e220b20 8e28c022 240bc080 19231001 10020000 61634068 08400000 cc644440 50641064
fe4f1477 806f05e7 9017c115 90c01715 240bc122 a224d042 8e162114 260bc480 00000000 00000540 00004ae3 61636326 68666064 c8624840 0d640c40 12695064
```

1 2 3 4 5 6 7 8 9 10 11 12 13 14 15 16

Результирующий параллельный код – до 16 операций за такт

Компиляторы обеспечивают эффективное распараллеливание на уровне операций

Асинхронная подкачка данных

- Основные свойства
  - Освобождает основной конвейер от операций обращения в память и вычисления адресов
  - Устраняет простои от неготовности данных
  - Не опустошает кэш
  - Увеличивает параллелизм обращений в память
  - Эффективно скрывает время доступа в память
- Применима как для вычислительных задач, так и для задач обработки данных

Значительно сокращает потери от доступа в память

МЦСТ ЭЛЬБРУС ИНЭУМ

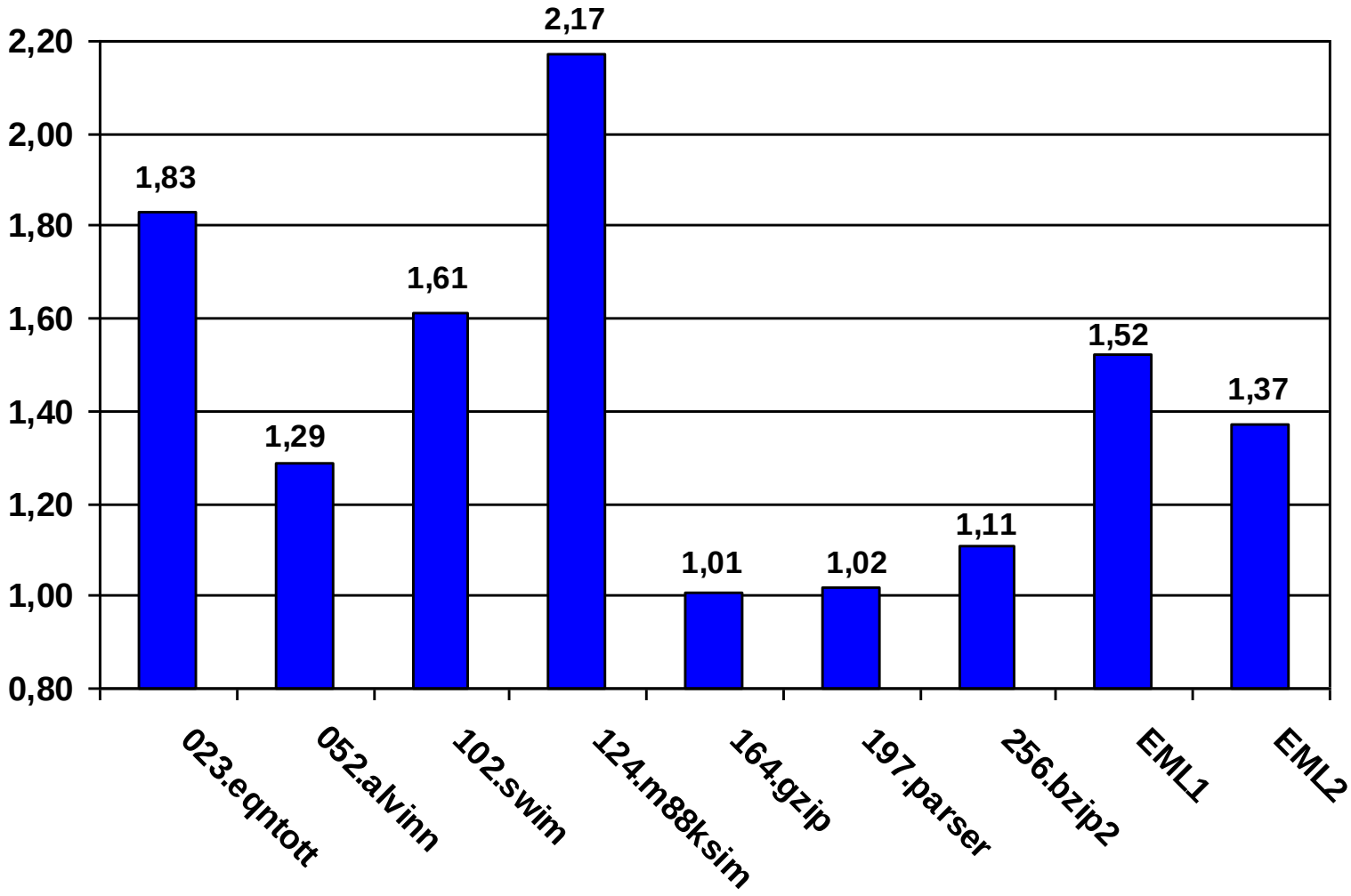
Число операций, выполняемых в ядре за такт

	Niagara RISC	Sparc64 VIIIfx	Core i7	Power 7	Эльбрус
Кол-во операций в исполнении	76	48	168	120	Вся программа
Макс. кол-во операций в такт	1	4	6	8	23
Среднее кол-во операций в такт при выполнении задач SPEC	0,5	1,1	1,26	1,18	3,1

Эльбрус имеет в 3 раза более высокую логическую скорость

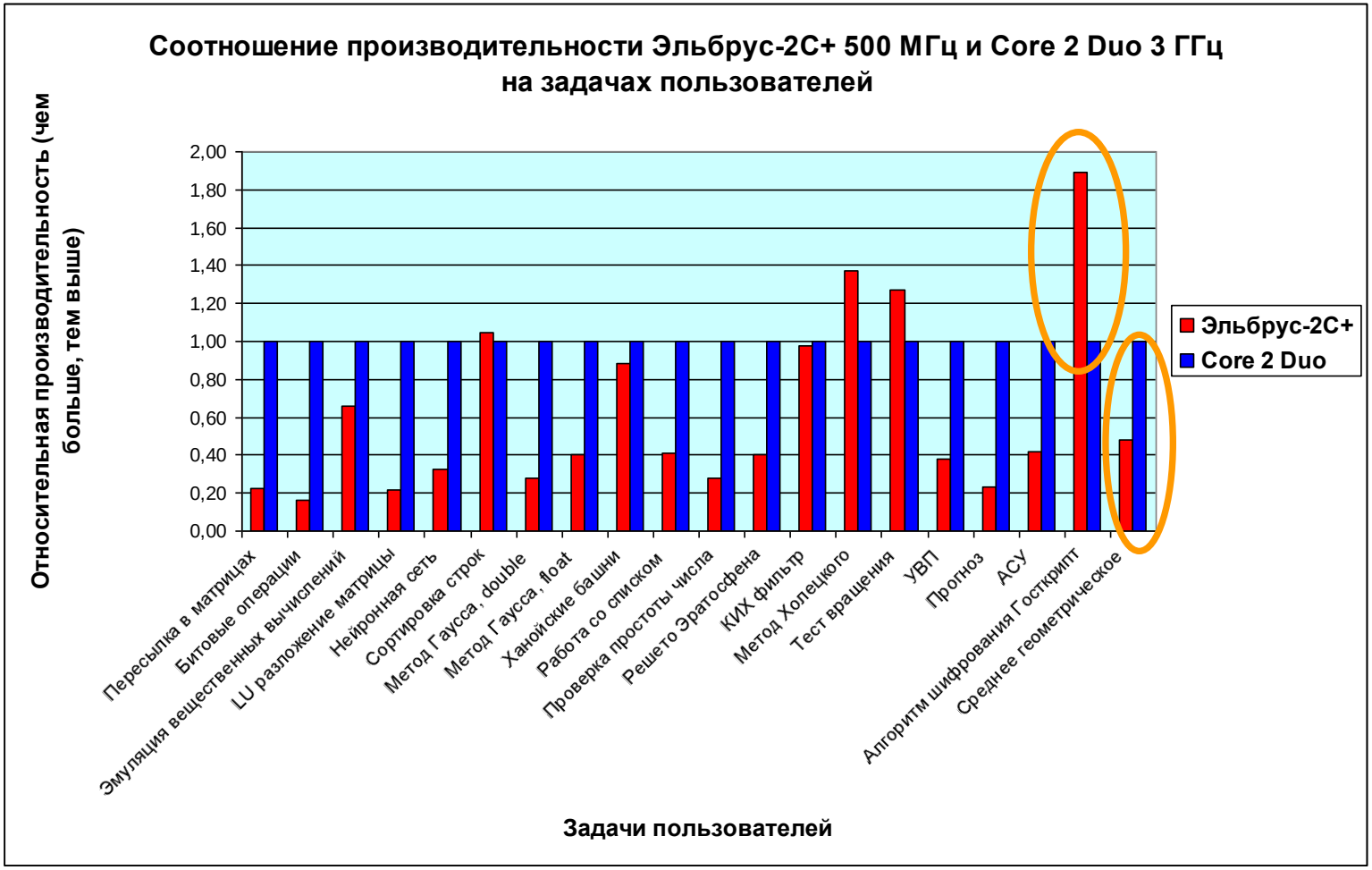
МЦСТ ЭЛЬБРУС ИНЭУМ

Прирост производительности от применения векторизации



Заметный прирост производительности за счет автоматической векторизации

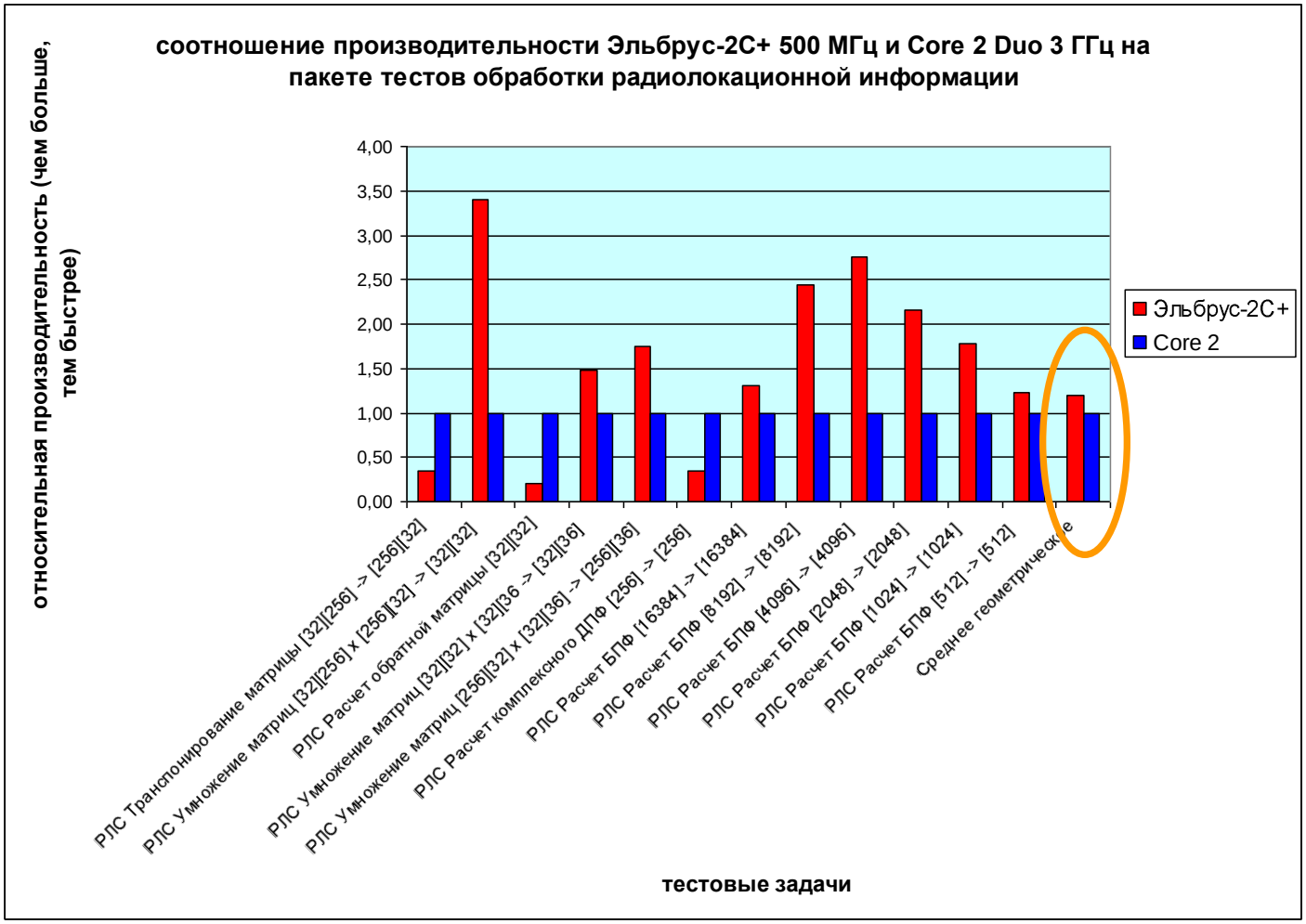
Сравнение на задачах пользователей (испытания 2012 г.)



Логическая скорость Эльбруса в 3 раза выше, т.к. при в 6 раз меньшей тактовой частоте абсолютная производительность ниже только в 2 раза

МЦСТ ЭЛЬБРУС ИНЭУМ

Сравнение на обработке радиолокационной информации



Абсолютная скорость Эльбруса выше на 20% при тактовой частоте в 6 раз меньше

МЦСТ ЭЛЬБРУС ИНЭУМ

Результаты распараллеливания на потоки на общей памяти

- Вещественные задачи с интенсивными обменами с памятью
- Прирост производительности при одновременном исполнении двух одинаковых задач для 2 процессоров – **1,68**
  - конфликты по доступу в память
- Абсолютный прирост за счет автоматического распараллеливания – **1,42**
  - Наличие не распараллеливаемых участков
  - Дополнительные конфликты по памяти при параллельном исполнении циклов
  - Эффективность по отношению к 2-м задачам – **77%**

Автоматическое распараллеливание на потоки дает прирост производительности

МЦСТ ЭЛЬБРУС ИНЭУМ

Результаты сравнительного анализа комиссии Минпромторга, декабрь 2012 г.

Сравнение ВК на базе «Эльбрус-2С+» (0,5 ГГц, 4 универсальных ядра) и ВК на базе Intel Core-i3 (3 ГГц, 2 ядра)

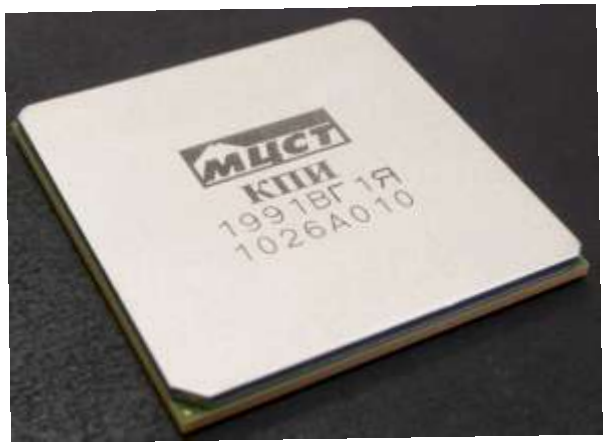
- Пакет задач пользователей (исполнение на одном ядре)
  - логическая скорость выше в 3 раза (исполняется на Э-2С+ в 2 раза дольше)
- Задачи радиолокации (исполнение на одном ядре)
  - логическая скорость выше в 7 раз (исполняется на Э2С+ на 14% быстрее)
- Параллельное исполнение задачи ТОМ на всех ядрах
  - логическая скорость выше в 2,5 раза (исполняется на 4-х ядрах Э-2С+ на 20% дольше, чем на 2-х ядрах Core-i3)
- Высокопроизводительные библиотеки
  - Линейная алгебра – логическая скорость выше в 6,5 раз (исполняется на Э-2С+ на 9% быстрее)
  - Обработка сигналов – логическая скорость выше в 14,3 раз (исполняется на Э-2С+ в 2,39 раза быстрее)
  - Алгоритмы над векторами – логическая скорость выше в 14,5 раз (исполняется на Э-2С+ в 2,43 раза быстрее)
  - видео – логическая скорость выше в 8,9 раз (исполняется на Э-2С+ в 1,49 раза быстрее)

Подтверждена высокая логическая скорость, сопоставим с процессорами Intel

МП «Эльбрус-2С+» и КПИ (чипсет)

МП Эльбрус-2С+: Год выпуска – 2011

- Технология – 90 нм, 10 слоев металла
- Тактовая частота – 500 МГц
- Количество транзисторов – 368 млн
- Рассеиваемая мощность – 25 Вт
- Состав микросхемы
  2 ядра с архитектурой «Эльбрус»,
  4 ядра ДСП с архитектурой Мультикор,
  средства межядерных связей.
- Суммарная производительность 28 Гфлопс:
  2-х ядер «Эльбрус» - 16 Гфлопс,
  4-х ядер ДСП - 12 Гфлопс
- Размер кристалла - 17,2x16,8 мм

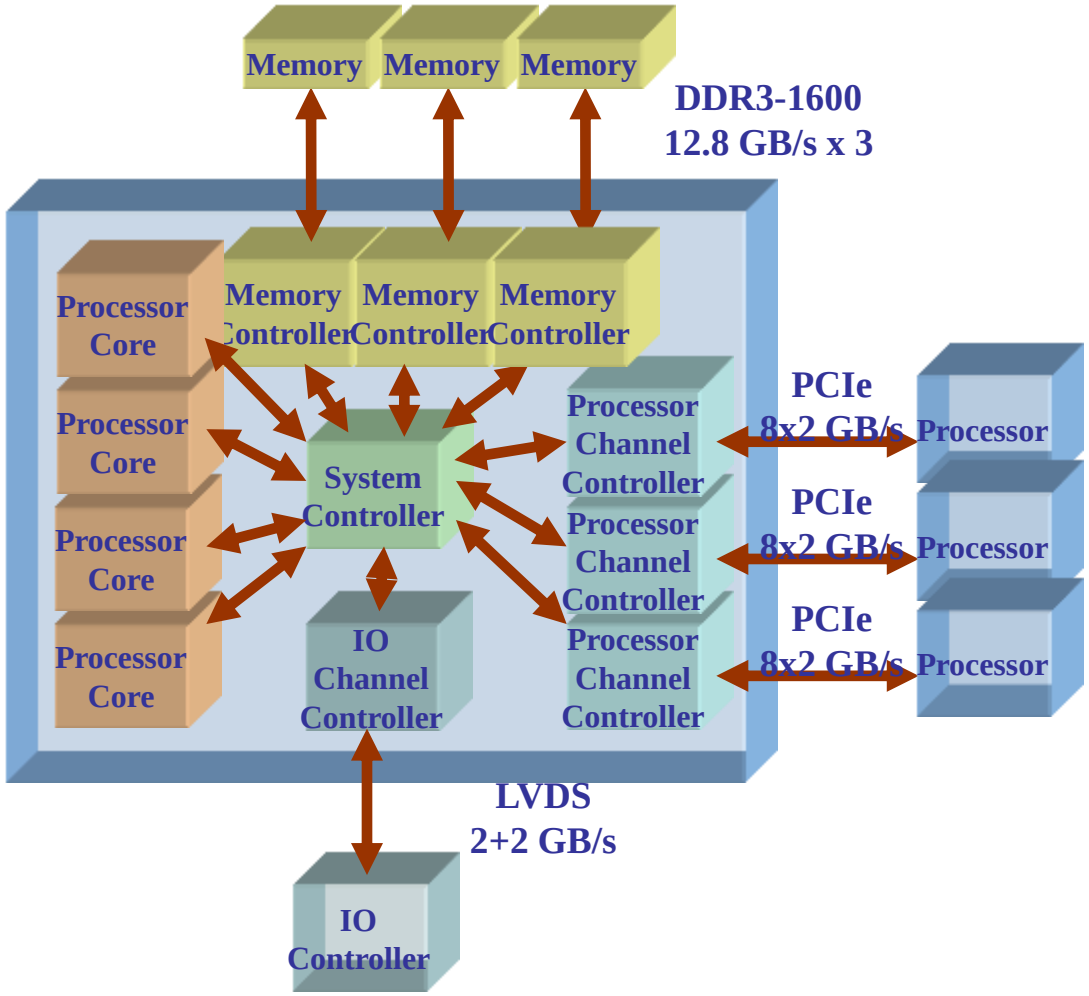


КПИ: Год выпуска – 2010

- Технология – 0.13 мкм, 9 слоев металла
- Тактовая частота – 250 Мгц
- Количество транзисторов – 30 млн
- Рассеиваемая мощность – 5 Вт
- Реализует 14 широко распространенных интерфейсов, в том числе:
  системный, PCI Express, PCI,
  Ethernet (10/100/1000), SATA(связь с дисками),
  USB 2.0,
  RS 232/485 (последовательный канал и др.)
- Размер кристалла – 10,6x10,6 мм

МП Эльбрус-4С (Эльбрус-2S)

- 4 универсальных ядра МП Эльбрус
- 8 Мбайт L2 кэш
- Технологические нормы 65 нм
- 986 млн. транзисторов
- Площадь 380 кв. мм
- Тактовая частота 1 ГГц
- Производительность 64 Gflops
- Пропускная способность памяти 38,4 Гбайт/сек
- 4-процессорная NUMA (каналы 16 Гбайт/сек)
- Год выпуска 2013



Макет МП Эльбрус-2S

Процессор изготовлен, идет подготовка к испытаниям

МЦСТ ЭЛЬБРУС ИНЭУМ

Пути повышения энергетической эффективности

- Технологические методы
  - Уменьшение размеров элементов
  - Снижение напряжения питания
  - Сокращение потерь от токов утечки
- Логические методы
  - Отключение неработающего оборудования
  - Динамическое управление частотой и напряжением питания
- Архитектурные методы
  - Замена динамического распараллеливания последовательного кода на программное при компиляции
    - В 3 раза более высокая скорость при той же рассеиваемой мощности
  - Планирование кода при компиляции для снижения потребления энергии
    - Результат – снижение рассеиваемой мощности на 20%

Технологических и логических методы дополняются архитектурными

МЦСТ ЭЛЬБРУС ИНЭУМ

Решение других актуальных задач развития ВТ и ИТ в МП «Эльбрус»

Экспоненциальный рост объема и сложности ПО

- Требуется существенное повышение надежности и безопасности

- Накопленный объем ПО требует решения проблемы совместимости

МЦСТ ЭЛЬБРУС ИНЭУМ

Рост объема и сложности ПО

- Экспоненциальный рост объема и сложности ПО
  - На примере дистрибутива Linux Debian (число пакетов увеличилось в 78 раз за 17 лет)

1996	1998	2000	2002	2005	2007	2009	2011	2013
474	1500	3900	8500	15400	18000	23000	29000	37000

  - Размер кода: 2001 г. – 55 млн. строк, 2007 – 288 млн. строк, 2013 – 457 млн. строк
- Показатели надежности
  - Число ошибок на 1000 строк кода (KLOC)
    - 2-5 для ПО с открытым кодом
      - потенциально в дистрибутиве Linux Debian 2013 г. 1 млн. ошибок
    - 0,5 для Windows
    - 0,1 для промышленного ПО (трудоемкость отладки в 10 раз превышает коммерческое ПО)
- Низкий уровень аппаратуры
  - Низкий ассемблерный уровень программирования
  - Отсутствие контроля критических ошибок

Существует огромный риск потери контроля над ПО и киберпространством

Пример – вирус stuxnet, поразивший и разрушивший ядерные объекты Ирана

Существо защищенного исполнения

Защита в МП «Эльбрус»

- Все указатели на объекты защищены тегами
  - Подделать указатель невозможно
- Указатели обеспечивают контроль границ объектов
- Поддерживается языковая модульность
  - Модулю доступны только свои функции и данные и явно переданные указатели на данные других модулей

Гарантируется защита от проникновения компьютерных вирусов

Повышается надежность и безопасность программ

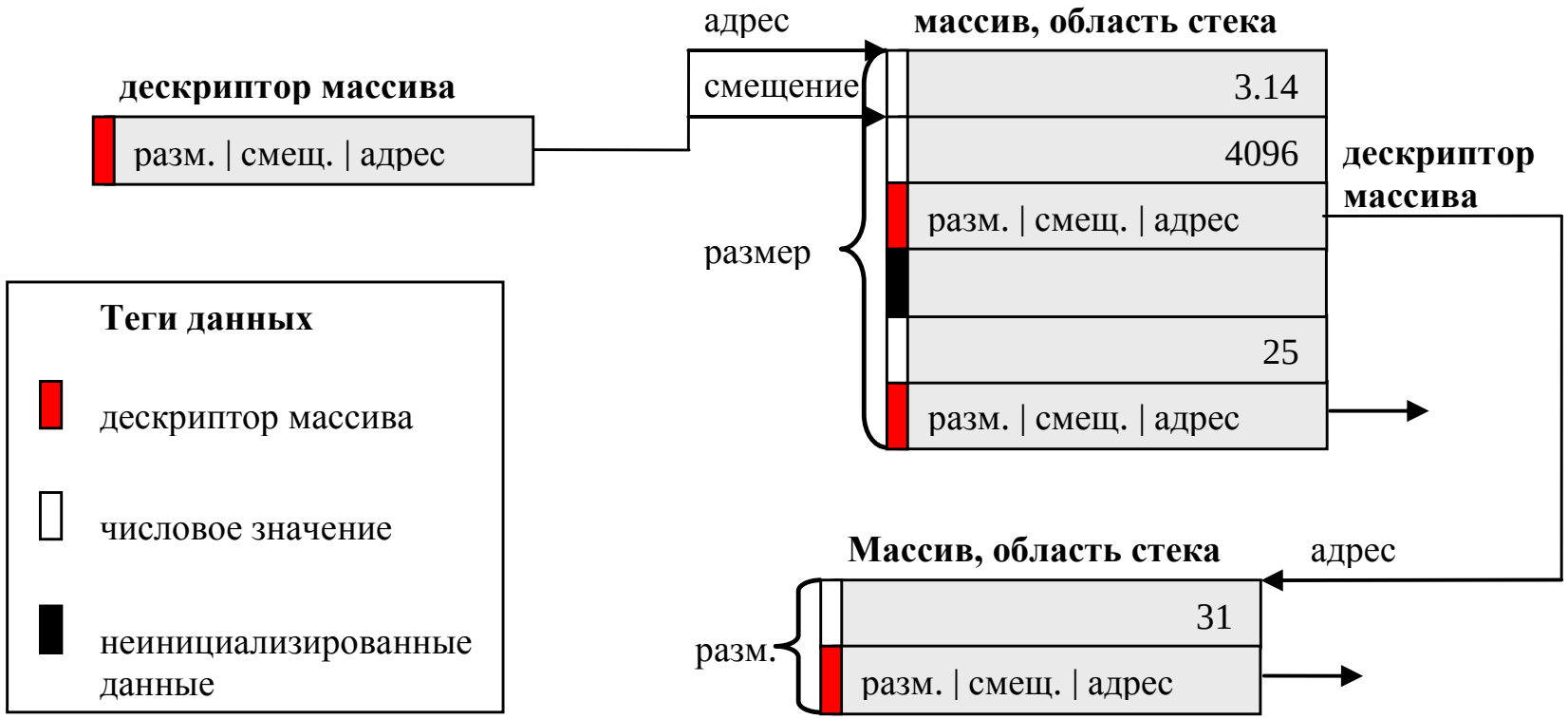
Традиционные архитектуры

- Числовые данные и ссылки на объекты неразличимы
  - Для обращения к данным используется пойнтер – просто число
- Объекты размещаются в линейной памяти и их границы не контролируются
- Разбиение программы на модули не понимается аппаратурой
  - Можно легко испортить работу надежного модуля

Нет аппаратной защиты от вирусов

Нет аппаратной защиты от ошибок в программах – снижается надежность и безопасность программ

Защита данных тегами



Память становится структурированной, отражает существо алгоритмов

Данные в памяти различимы

Выполняются важнейшие свойства объектов и ссылок

МЦСТ ЭЛЬБРУС ИНЭУМ

Программная реализация контроля границ с аппаратной поддержкой

- Контроль границ объектов остается самой серьезной критической уязвимостью во всем программном обеспечении

- Intel Pointer Checker – программные средства отладки с контролем границ
  - Требует специального управления компиляцией программ
  - Замедление исполнения в 3-5 раз

- В 2013 г. Intel ввел в систему команд будущих МП аппаратную поддержку технологии Pointer Checker
  - Дополнительные объекты для границ в памяти, новые регистры, специальные операции
    - Явные дополнительные проверки в коде
  - Замедление исполнения в 2-3 раза, пригодно только для отладки
  - Свидетельствует о важности аппаратной поддержки повышения безопасности и надежности программ
  - Не является универсальным средством контроля безопасности исполнения

Решения Intel можно использовать только для отладки, а этого не достаточно

МЦСТ ЭЛЬБРУС ИНЭУМ

Защищенное исполнение: результаты

Задачи	Всего задач	Задач с найденными ошибками
Задачи пользователей	7	4
Пакет SPECint95	8	7
Пакет негативных тестов *samate* на защищенность	888	874 (14 не выявленных не нарушают защиту)

Типы обнаруживаемых ошибок:

- нарушение границ объекта (переполнение буфера) – 53%
- использование неинициализированных данных – 37%
- использование опасных конструкций языка или опасных отклонений от стандарта языка – 10%

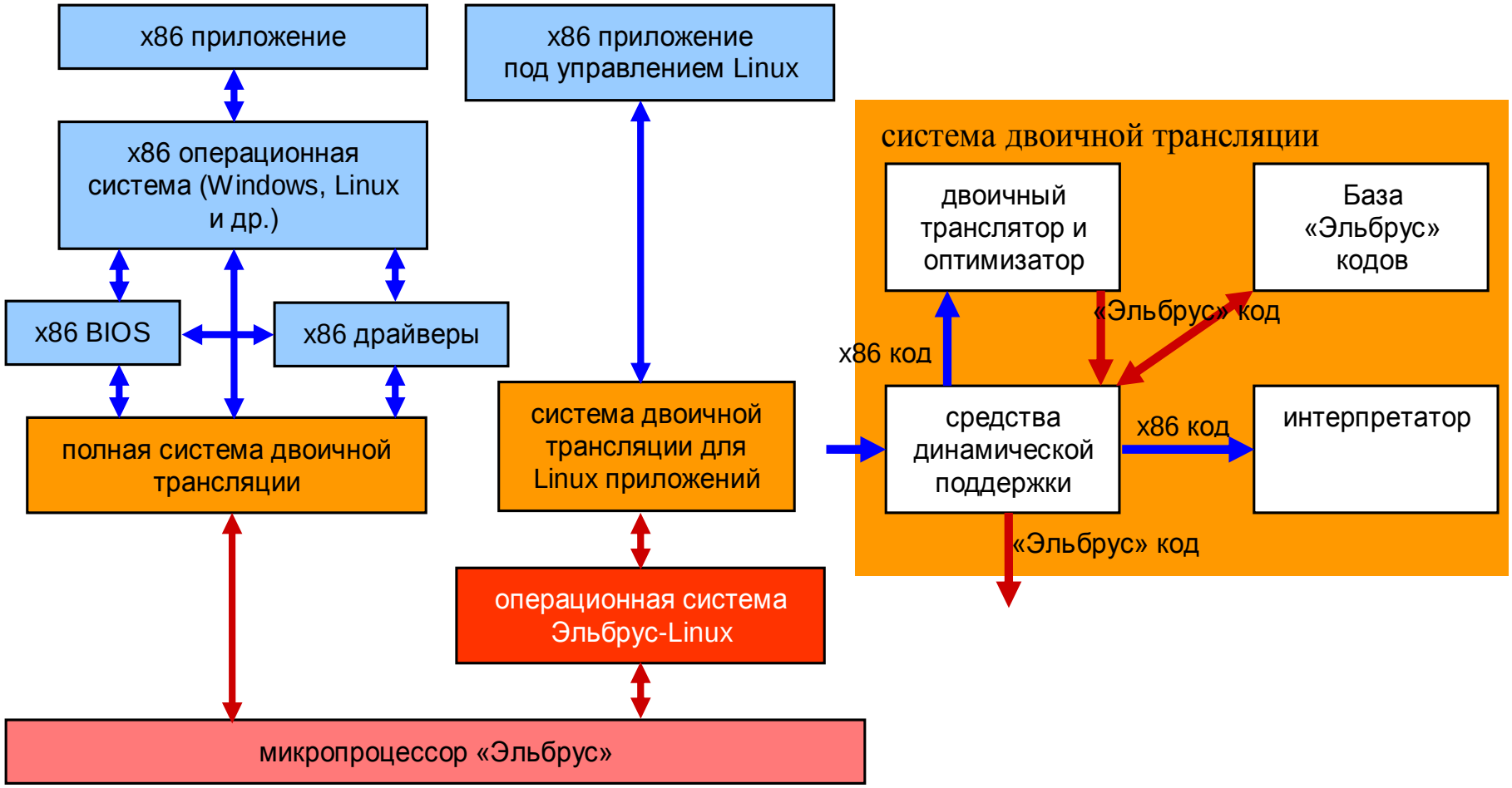
Эффективный инструмент отладки и обеспечения надежности программ

МЦСТ ЭЛЬБРУС ИНЭУМ

Что дает защищенное исполнение программ в архитектуре «Эльбрус»

- Фундаментальные средства борьбы с любыми уязвимостями из-за ошибок в программах
- Локализацию сложных программных ошибок без потери эффективности
- Эффективное исполнение благодаря параллельному аппаратному контролю
  - Операции не замедляются
- Возможность создания надежных и безопасных программ большими коллективами в сжатые сроки
- Необходимо дальнейшее развитие технологии и большой объем доработок существующих программ до требований повышения их безопасности и надежности

Двоичная трансляция для совместимости



Обеспечивает эффективную совместимость за счет использования параллелизма

МЦСТ ЭЛЬБРУС ИНЭУМ

Эффективная двоичная совместимость с Intel x86, x86-64

- Функциональность
  - Полная совместимость с архитектурой Intel x86 (x86-64 с МП Эльбрус-4С)
  - Прямое исполнение 20+ операционных систем, в том числе: MSDOS, Windows XP, Linux, QNX
  - Прямое исполнение 1000+ самых популярных приложений
  - Исполнение приложений под ОС «Эльбрус» (Linux)
- Производительность – 80% от нативной
  - Достигается за счет скрытой системы оптимизирующей двоичной трансляции
  - Мощная аппаратная поддержка в МП «Эльбрус»
- Лицензионная независимость от Intel

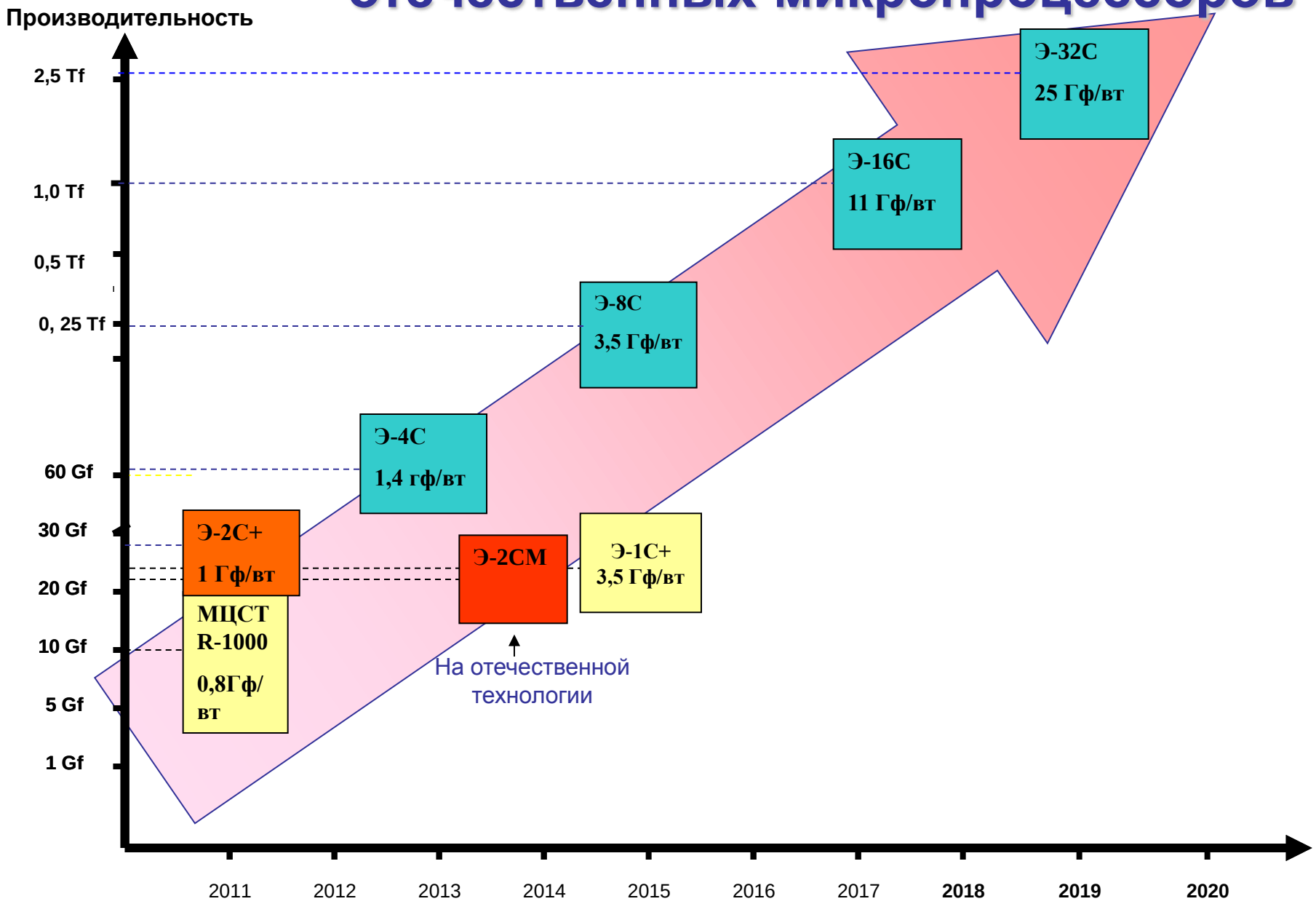
Совместимость по кодам с архитектурой, доминирующей в суперкомпьютерах, серверах, лэптопах, ноутбуках

МЦСТ
ЭЛЬБРУС
ИНЭУМ

Развитие МП линии «Эльбрус»

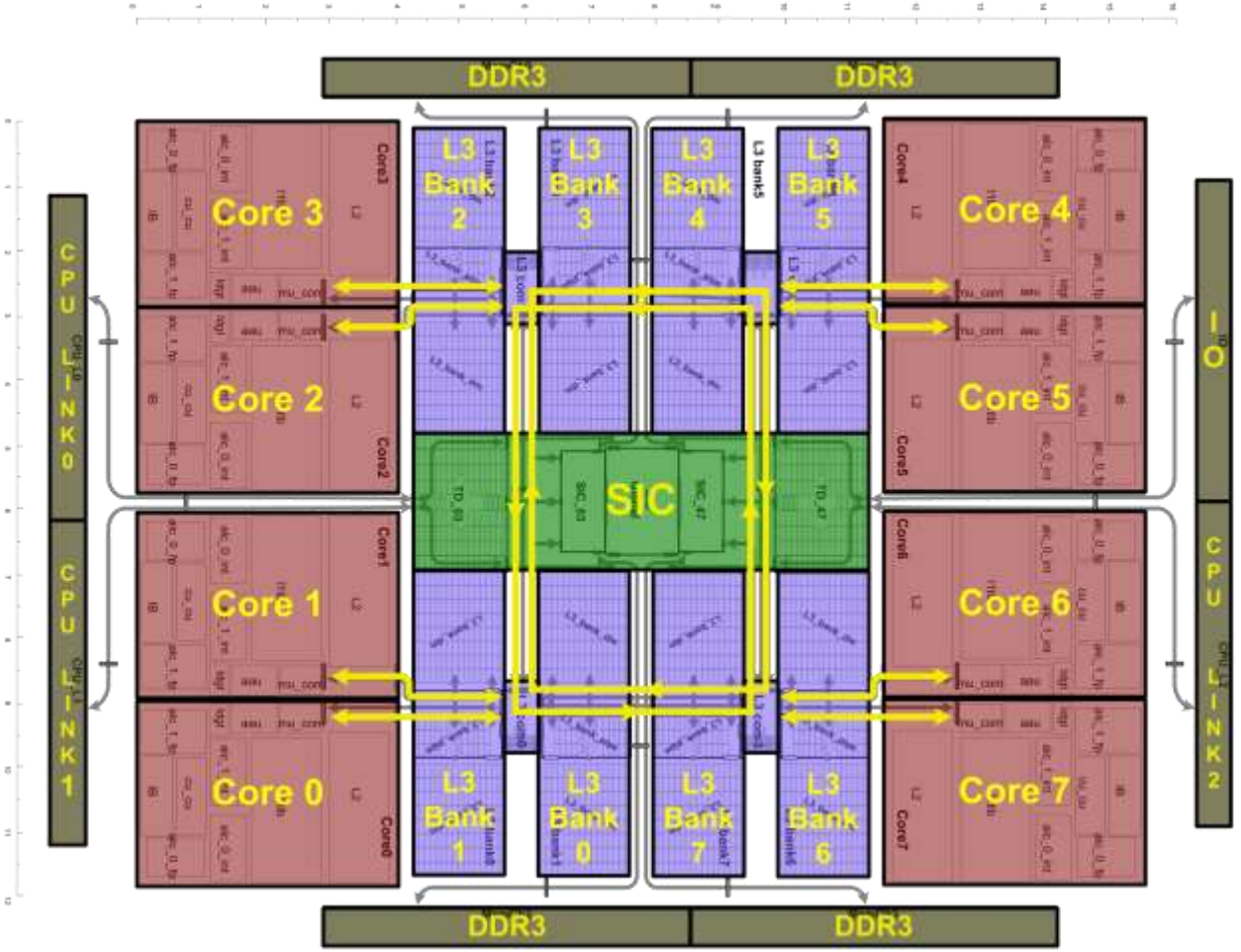
МЦСТ ЭЛЬБРУС ИНЭУМ

Перспектива развития отечественных микропроцессоров



МП Эльбрус-8С

Характеристики
- 8 ядер «Эльбрус»
  - число операций за такт – 30
- Частота 1,3 - 1,5 ГГц
- Производительность 250 Gflops
- Технологический процесс 28 нм
- Площадь кристалла 350 кв. мм
- L2 Cache – 512 KB на ядро
- L3 Cache – 16 MB, shared
- Год выпуска 2015



Более производительное ядро

МЦСТ ЭЛЬБРУС ИНЭУМ

Направления развития МП

- Разработка МП на технологии 28нм, 22нм, 14нм, 10 нм с переходом на полнозаказную схемотехнику для достижения тактовой частоты более 3 Ггц

- Развитие энергоэффективной архитектуры «широкого командного слова» для обеспечения производительности более 32 Гфлпс/ГГц на ядро с достижением *однопоточной* производительности более 100 Гфлпс на ядро и соотношение >20 Гфлпс/Вт , необходимого для создания экзафлопсных систем

- Реализация в одном МП до 32 ядер

- Обеспечение высокой достоверности и надежности вычислений за счет развития технологии защищенных вычислений

- Встраивание в МП высокоскоростных каналов межпроцессорной и межмашинной связи для создания высокопроизводительных СВТ

МЦСТ ЭЛЬБРУС ИНЭУМ

Развитие технологии защищенного исполнения программ

- Внедрение технологии
  - Адаптация базового ПО к режиму защищенного исполнения
    - Требуется перекомпиляция с доработкой в связи с использованием низкоуровневого программирования
      - Данные и ссылки рассматриваются как набор неструктурированных битов
      - Сознательно используются непереносимые или неопределенные свойства языков программирования
    - Устранение ошибок, не обнаруживаемых на обычных архитектурах
- Добавление аппаратной поддержки для облегчения переноса программ
  - Более полная поддержка смешанного режима исполнения
    - Новые приложения – в защищенном режиме, надежные библиотеки – в обычном
- Более широкая поддержка в операционной системе и в компиляторах
  - Защищенный режим в самой ОС
  - Специальные оптимизации в компиляторах

МЦСТ
ЭЛЬБРУС
ИНЭУМ

СПАСИБО за внимание!