МЦСТ
ЭЛЬБРУС

Настоящее и будущее микропроцессоров "Эльбрус" в российских компьютерах

Владимир Волконский

ЗАО «МЦСТ»

Лекция для слушателей Летней Суперкомпьютерной Академии МГУ
23.07.2017

МЦСТ
ЭЛЬБРУС

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
  - сохраняя возможность вносить изменения в архитектуру

Инновационные технологии Эльбрус

Параллельная энергоэффективная архитектура

- 25-48 оп. за такт, явный параллелизм операций
- Высокая однопоточная производительность
- Процессор общего назначения
- Собственный оптимизирующий компилятор
- Сбалансированное соотношение процессора и памяти

Эффективная двоичная совместимость с Intel x86, x86-64

- Исполнение Windows XP, Windows 7 и выше, Linux, QNX
- Слой совместимости для приложений в кодах x86/x86-64 среде Linux
- Производительность до 80% от нативной, по логической скорости лучше Intel Core
- Базируется на аппаратно поддержанной технологии динамической двоичной компиляции
- Лицензионная независимость от Intel

Технология защищённых вычислений

- Защита (аппаратная) логической структуры памяти
- Гарантированное обнаружение уязвимостей
- 10х повышение скорости отладки программ
- Надежность программ, защита от компьютерных вирусов

Программируемость, надежность, энергоэффективность

Характеристики	Intel Sandy Bridge	Эльбрус-8С
Пиковые возможности МП – операций за такт	8	25
Число операций за такт на SPECcpu2006	1,54	4,88

МЦСТ
ЭЛЬБРУС

Достижение высокой логической скорости МП линии «Эльбрус»

МЦСТ ЭЛЬБРУС

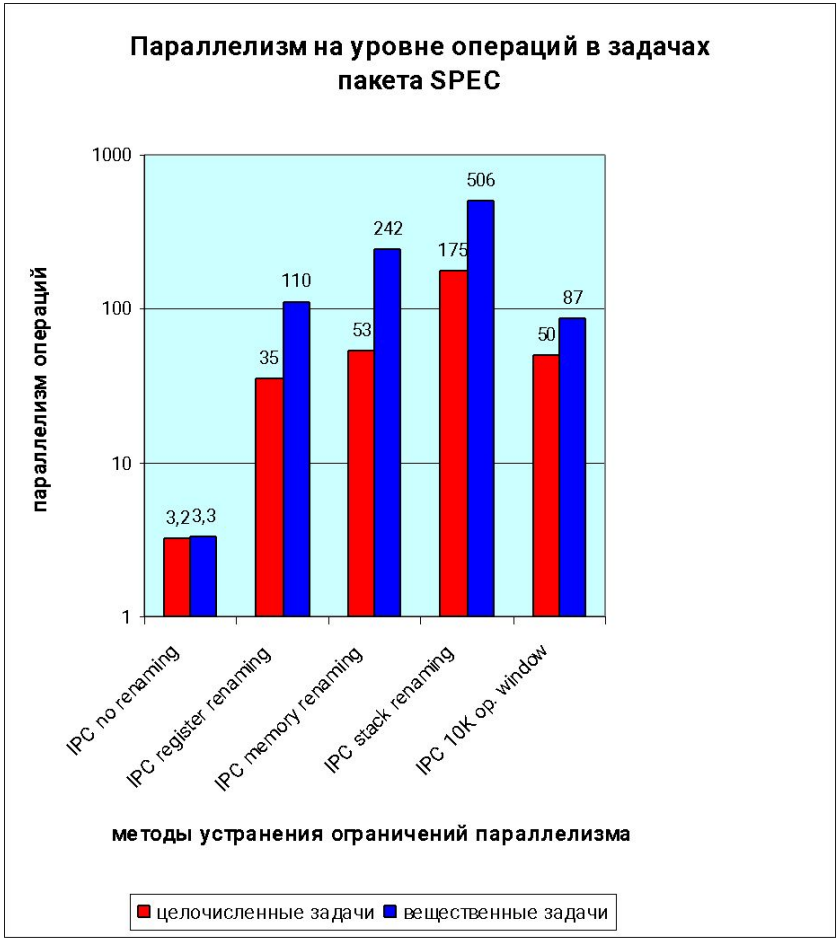
Почему важна производительность ядра

- 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 уметь использовать

МЦСТ ЭЛЬБРУС

Параллелизм архитектуры «Эльбрус»

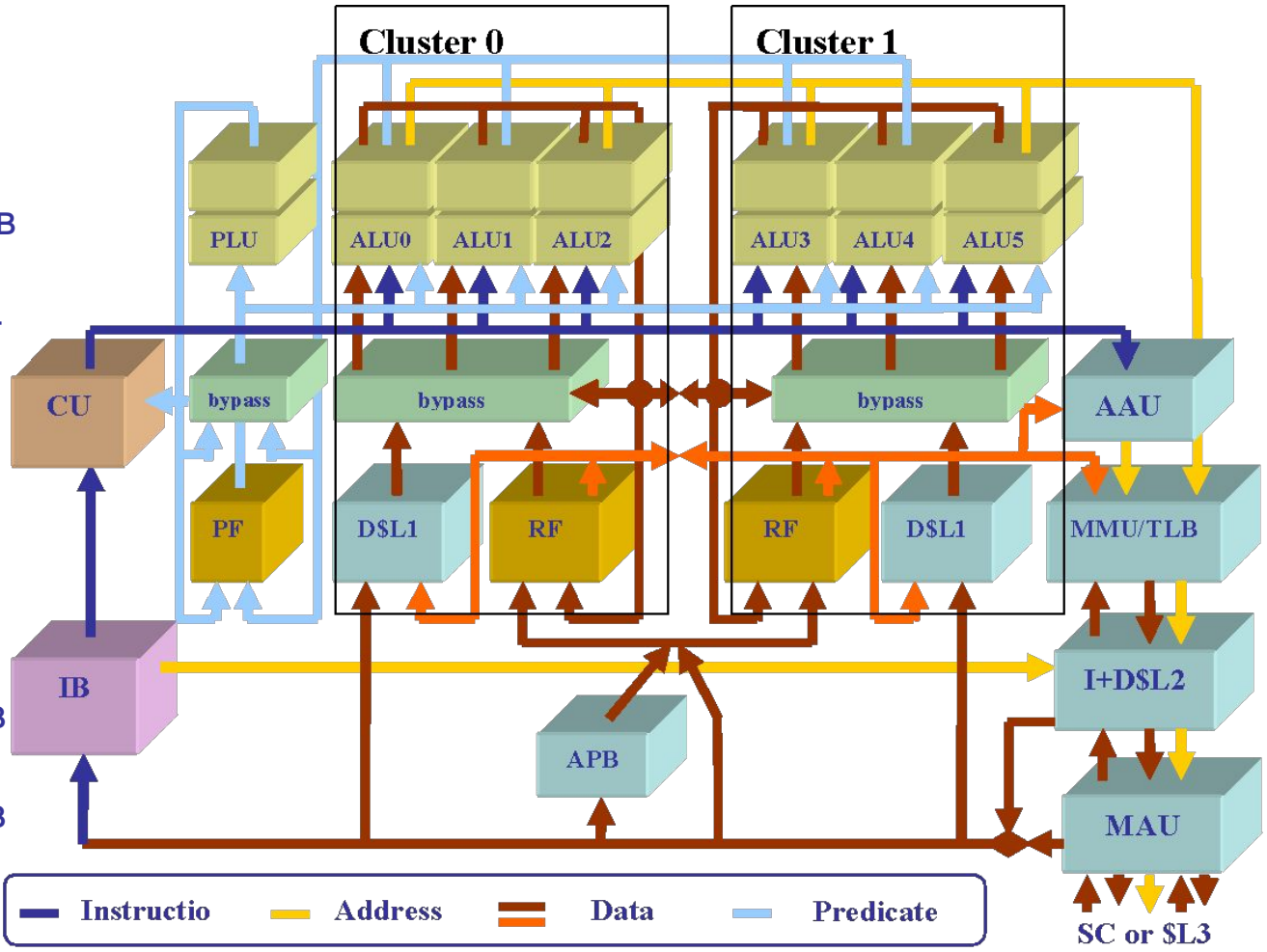
- Параллелизм скалярных операций
  - До 30 операций за такт в разрабатываемых МП
  - До 40-50 операций за такт в перспективных МП
- Параллелизм векторных операций
  - Упакованные данные в векторе
    - байтовые – 8, двухбайтовые – 4, 32-разрядные – 2
  - Как минимум, удвоение числа операций в перспективных 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

Структура ядра МП линии Эльбрус

2 кластера

IB –буфер команд (I$L1)
CU – устр. управления
PF – файл предикатов
PLU – вычисл. предикатов
D$L1 – кэш 1-го уровня
ALU1-6 – каналы арифм.-логических устр.
RF – регистровый файл
APB – буфер подкачки массивов
AAU – устр. асинхронной подкачки массивов
MMU – устр. преобраз. виртуальных адресов
TLB – кэш преобраз. виртуальных адресов
I+D$L2 – кэш 2-го уровня
MAU – устр. доступа к памяти
SC – системный контроллер
$L3 – кэш 3-го уровня
bypass – быстрые связи



Пиковая производительность ядра

	лин.уч.	циклы		
Int (8) / FP (9/12) / St (2) / Ld (4)	-	10/12	+	+
Обработка предикатов	-	3	+	+
Передача управления	-	1	+	+
Загрузка литерала 32/64	-	4/2	+	
Асинхронная загрузка в РФ	-	4		+
Адресная арифметика	-	4		+
Обработка счетчика цикла	-	1		+
Всего:		14/16	23/25	

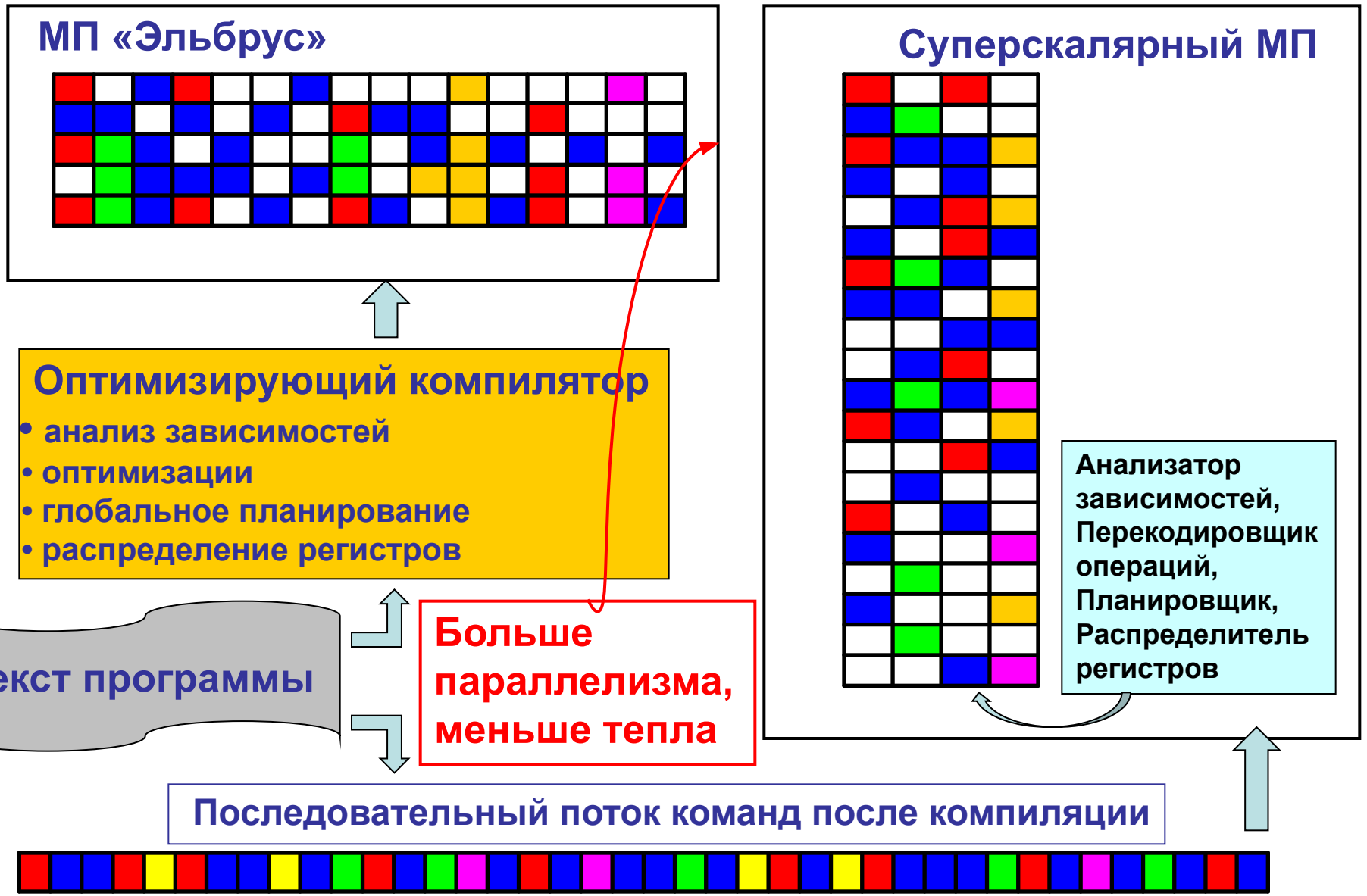
В МП Эльбрус-8С увеличивается число вещ. операций за такт до 12 dp
За счет расширения регистров в 2 раза Эльбрус-8СВ и Эльбрус-16С число операций за такт удваивается (до 24 dp)

МЦСТ ЭЛЬБРУС

Поддержка параллельных вычислений

- Одновременный запуск большого числа операций
- Большие регистровый и предикатный файлы
  - Окна произвольного размера – экономия регистров
  - Стековая организация – снижение накладных расходов на переключение
- Оптимизация операций передачи управления (подготовки переходов)
- Режим условного (предикатного) выполнения операций
- Режим спекулятивного выполнения операций
  - Обеспечение корректности параллельных вычислений
    - Механизм отложенного прерывания
  - Оптимизация одновременного исполнения программных ветвей с разными вероятностями реализации
- Поддержка циклических вычислений; программная конвейеризация циклов
  - Автоматическая генерация и использование цикловых событий и состояний для управления вычислениями и отдельными операциями
  - Механизм циклического переименования регистров RF и PF
  - Специализированное устройство регулярной адресной арифметики для обращения к элементам массивов
  - Специализированное устройство предварительной (асинхронной) подкачки элементов массивов

Параллелизм МП «Эльбрус» и суперскалярного МП



Параллелизм операций (1)

Пример: исходный текст

```
u = (a – c) – (b + c) – (c + d);
x = (e – f);
y = (a + b) + e;
z = (a + b) + (a – c) + (e – (b – d));
```

Всего 36 операций

a, b, c, d, e, f – операции считывания данных из памяти
u, x, y, z – операции записи данных в память

Параллелизм операций (2)

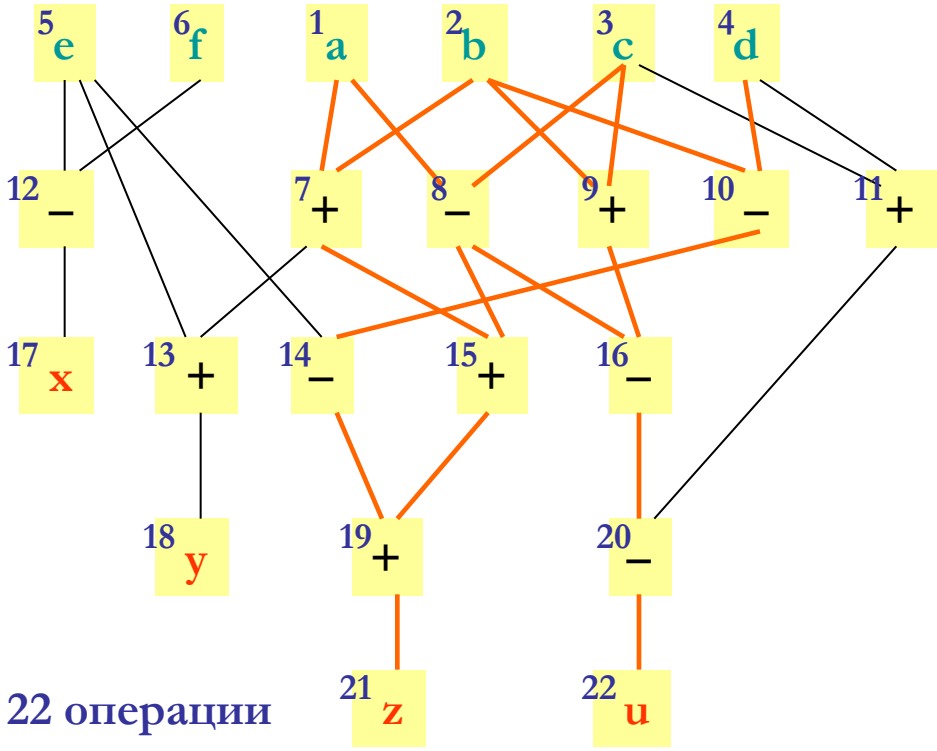
Пример: исходный текст

```
u = (a – c) – (b + c) – (c + d);
x = (e – f);
y = (a + b) + e;
z = (a + b) + (a – c) + (e – (b – d));
```

36 операций

Число операций в графе уменьшилось за счет оптимизирующего компилятора

Представление в виде графа зависимостей



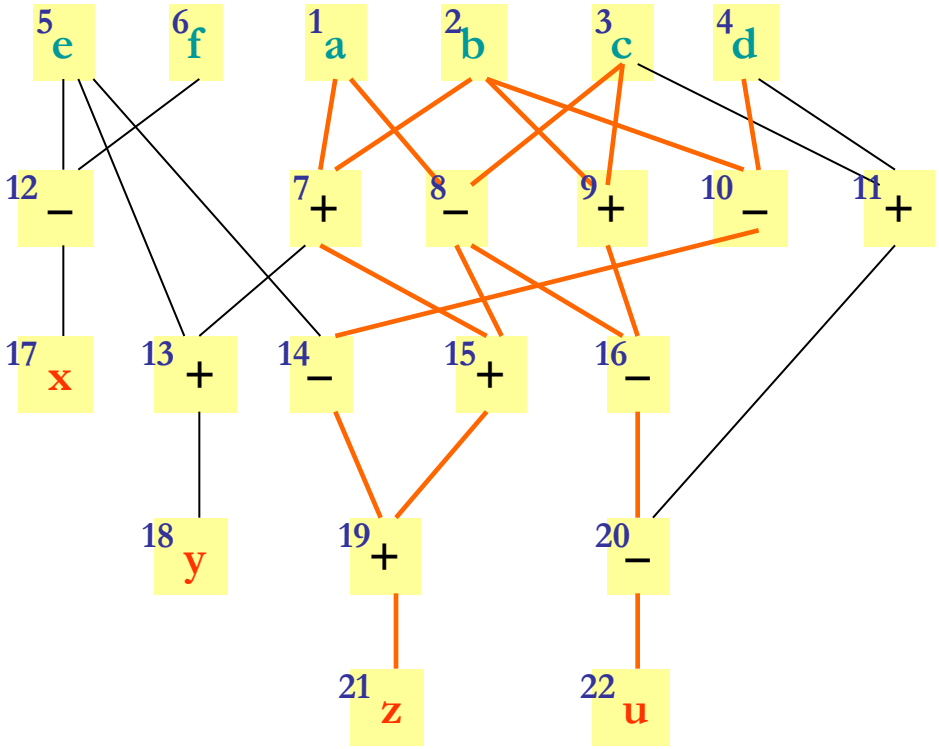
22 операции

Зависимости между операциями

Критический путь

Параллелизм операций (3)

Представление в виде графа зависимостей



Критический путь – **5 тактов**

Параллельный код Эльбруса

```
1  2   3   4
5  6   7   8   9  10
11 12  13  14  15 16
17 18  19  20
21 22
```

5 тактов

Каждая строка соответствует одной широкой команде, запускающей все операции в ней параллельно

Совпадает с критическим путем!

Параллелизм операций (4)

Последовательный код

1
3
8
2
9
16
4
11
20
22
5
6
12
17
7
13
18
15
10
14
19
21

22 такта

Аппаратный Планировщик Intel x86

Параллельный код Intel

1			
3			
2	8		
4	7	9	
5	16	11	10
6	20	15	14
22	12	13	19
17			
18			
21			

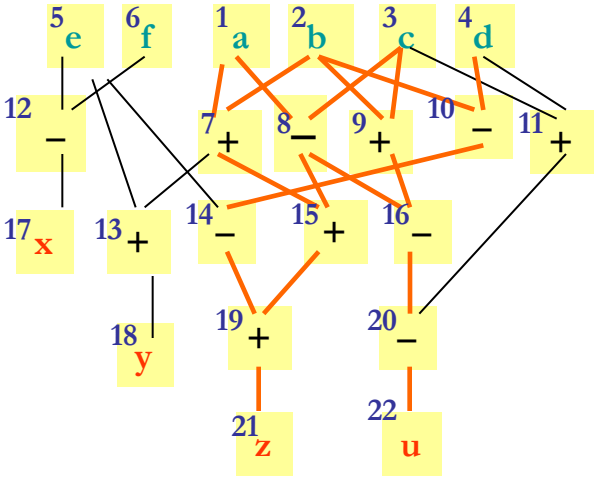
10 тактов

Параллельный код Эльбруса

1	2	3	4		
5	6	7	8	9	10
11	12	13	14	15	16
17	18	19	20		
21	22				

5 тактов

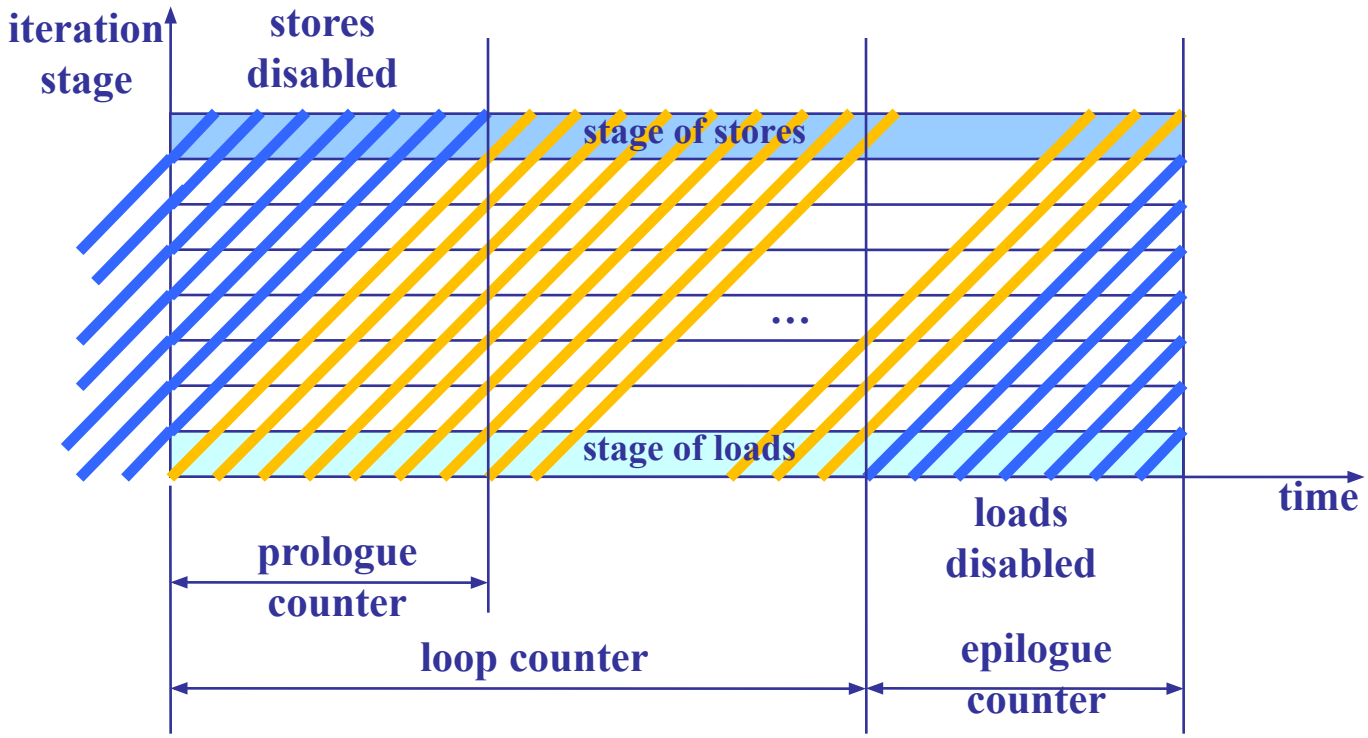
Код планируется компилятором по графу



В 2 раза быстрее Intel

МЦСТ ЭЛЬБРУС

Конвейеризованные циклы

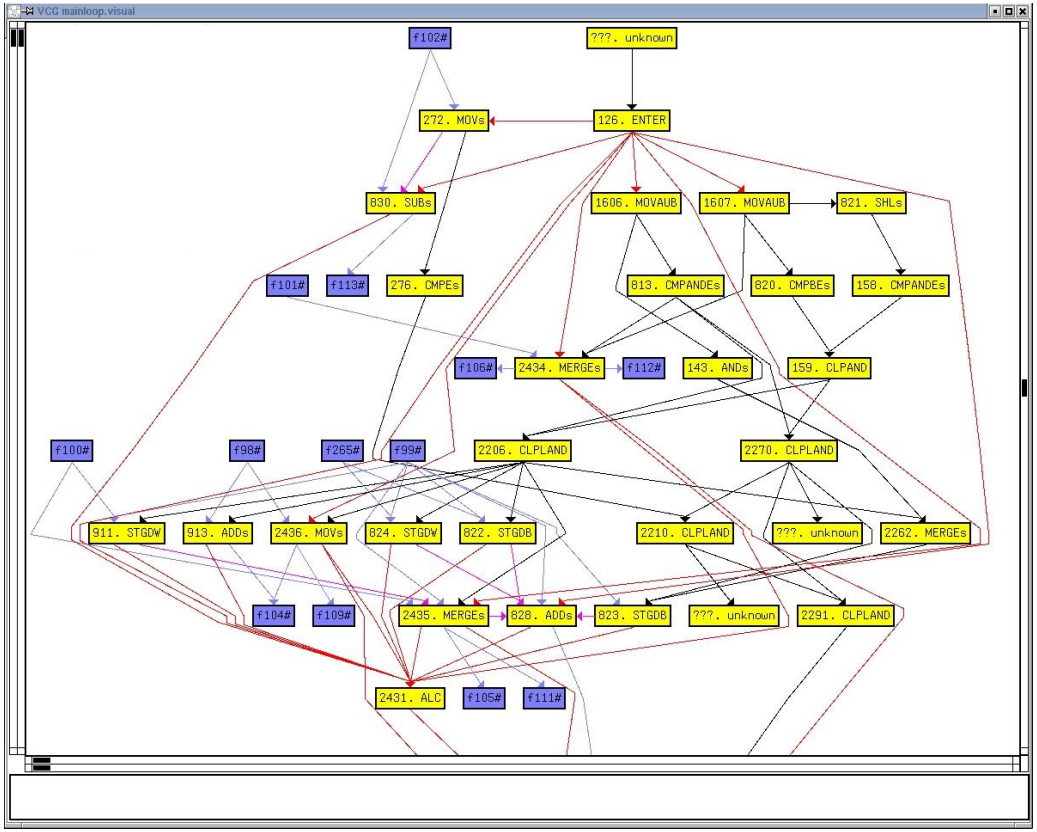


- Переименование регистров в цикле
- Поддержка пролога и эпилога
- Спекулятивные вычисления и предикаты

МЦСТ ЭЛЬБРУС

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

Трасса выполнения с блокировками при обращении в память (без оптимизаций)

Синхронная подкачка данных в цикле
- потери из-за остановок подкачки

Асинхронная фоновая подкачка данных в специальный буфер
-потери минимизированы до одного самого долгого доступа

- простой из-за блокировки

- полезные вычисления

МЦСТ ЭЛЬБРУС

Исходный код SpMV (CSR)

```
double A[size]; // Разреженная матрица в формате CSR
double x[N]; // Вектор
double r[N]; // Результат
int iA[N+1]; // Индексы начал строк в массиве jA
int jA[size] // Индексы элементов матрицы в строке;
…
for( int i = 0; i < N; i++){
  double s=0.0;
  for ( int k = iA[i]; k < iA[i+1]; k++) {
     s +=  A[k]*x[jA[k]];
  }
  r[i] = s;
}
// преобразованный цикл для более оптимального исполнения
for (j=0; j<sz; j++) {
  s+=A[j]*X[jA[j]];
  if (j==iA[i+1]-1) // последняя итерация бывшего внутреннего цикла
  {                 // действия охватывающего цикла
    r[i]=s;
    s=0;
    i++;
  }
}
```

Код Эльбруса для SpMV (CSR)

```
{
! Подготовка внешнего цикла, начало конвейера
}
{! Внешний цикл 6 тактов
loop_mode:      ! Внутренний цикл 1 такт
 ct    %ctpr1 ? ~%pred3 && %NOT_LOOP_END          ! Переход на начало
 fmuld,0,sm     %db[54], %db[109], %db[64]         ! A[k]*x[…]
 faddd,1,sm     %db[69], %db[72], %db[61]?%pcnt0   ! s += …
 ldgdd,2,sm     %r4, %b[76], %db[69]               ! Загрузка x[jA[k]]
 cmpesb,3,sm    %r0, %r15, %pred0                  ! Условие выхода
 shls,4,sm      %b[64], 0x3, %b[72]                ! jA[k]*sizeof(double)
 adds,5,sm      0x1, %r0, %r0 ? %pcnt3             ! Инкремент k++
 movad,1        area=0, ind=0, am=1, be=0, %db[0]! Загрузка A[k]
 movaw,3        area=0, ind=0, am=1, be=0, %b[54]! Загрузка jA[k]
}

{
loop_mode: ! Корректирующий код, конвейеризирован программно 6 татов
…
loop_mode:
 ct %ctpr1 ? %NOT_LOOPEND ! Завершение внешнего цикла
}
! Доработка конвейера внешнего цикла, конец внешнего цикла
```

(внутренний цикл «раскручен» на 4 итерации)

Архитектура и производительность МП: HPL & HPCG

Top 10 HPL (Linpack), ноябрь 2016						Top 10 HPCG *)		
No		Cores млн.	Pflops	Peak	%peak	No HPCG	Tflops	%peak
1	SW TaihuLight, China,	10,5	93,0	125	74	4	371	0,3
2	Tianhe-2, China, Xeon Phi	3,1	33,8	54,9	61	2	580	1,1
3	Titan, USA, Opteron+Nvidia	0,56	17,6	27,1	65	7	322	1,2
4	Sequoia, USA, **BlueGene/Q**	1,57	17,2	20,1	**85**	6	330	**1,6**
5	Cori, USA, Xeon Phi	0,62	14,0	27,9	50	5	355	1,3
6	Oakforest-PACT, Japan, Xeon Phi	0,56	13,5	24,9	54	3	385	1,5
7	K computer, Japan, **SPARC64 viii-fx**	0,70	10,5	11,3	**93**	1	603	**5,3**
8	PizDaint, Swiss, Xeon+Nvidia	0,21	9,8	16,0	61	13	125	1,6
9	Mira, USA, **BlueGene/Q**	0,79	8,6	10,0	**86**	10	167	**1,7**
10	Trinity, USA, Xeon E5	0,30	8,1	11,0	74	8	183	1,6

*) Задача HPCG предложена автором Linpack Донгаррой как более близкая к реальным HPC задачам

Универсальные МП лучше сбалансированы (в соотношении «пиковая производительность – пропускная способность памяти»)

Новые архитектуры должны компенсировать замедление «закона» Мура

- множество специализированных ядер (GPGPU, Intel Phi)
- более мощные универсальные ядра (Nvidia Denver, Intel Soft Machine, Эльбрус)

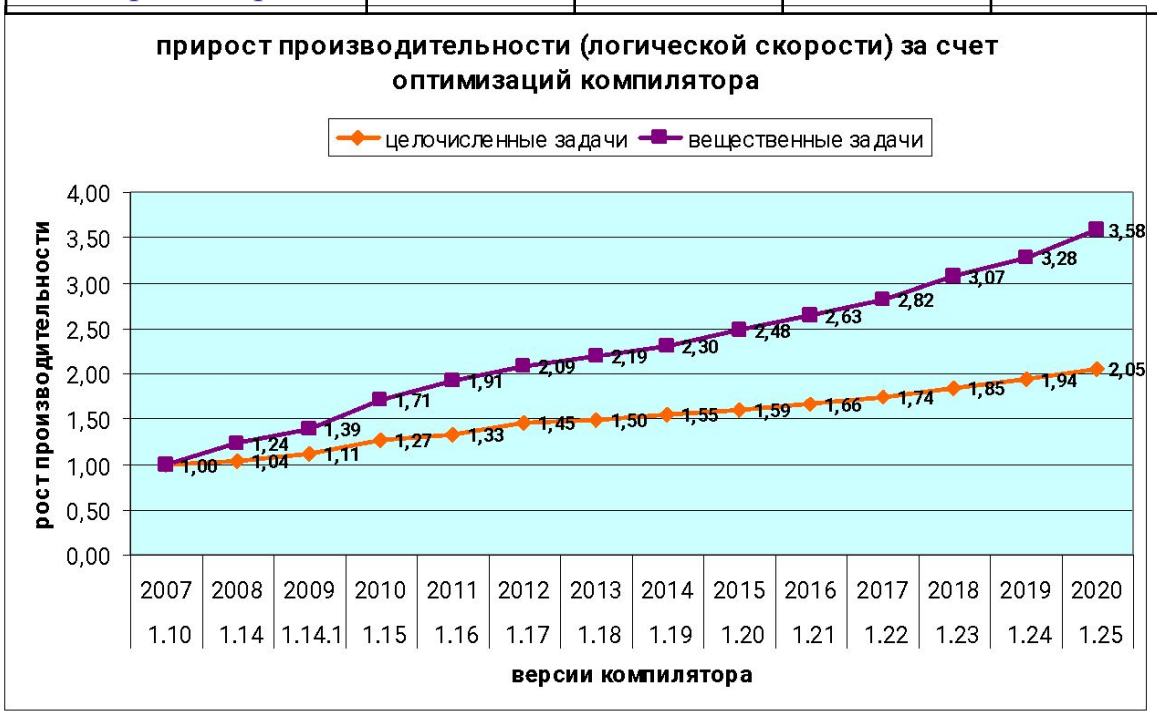
На базе МП Эльбрус-8С – 85% на HPL, 6+% на HPCG (и это не предел)

Рост производительности за счет компилятора

Производительность ядра Эльбрус существенно зависит от оптимизирующего компилятора

- На текущей версии компилятора на пакете SPECcpu2006 логическая скорость процессора Эльбрус-8С (1,3 ГГц) превосходит Intel Sandy Bridge (3,76 ГГц, компилятор gcc) на 17% на целочисленных задачах и на 30% на вещественных задачах
- За 2016 год производительность на пакете SPECcpu2006 выросла на 10-12%
- Реальный прирост производительности за счет оптимизаций компилятора, начиная с 2007 г. (появление архитектуры) составил 1,6 раз для целочисленных и 2,6 раз для вещественных задач
- К моменту выхода Эльбрус-16С только за счет компилятора производительность целочисленных задач вырастет на 20%, а вещественных – на 35%

	Производительность на ядро		Пересчет на 1 ГГц	
Пакет \ компьютер	Эльбрус-8 С	Intel Sandy Bridge	Эльбрус-8 С	Intel Sandy Bridge
Тактовая частота, ГГц	1,3	3,76	1,0	1,0
SPECcpu2006int	13,03	32,18	10,02	8,56
SPECcpu2006fp	17,02	37,68	13,09	10,02



МЦСТ
ЭЛЬБРУС

Обеспечение безопасности и надежности на базе МП линии «Эльбрус»

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

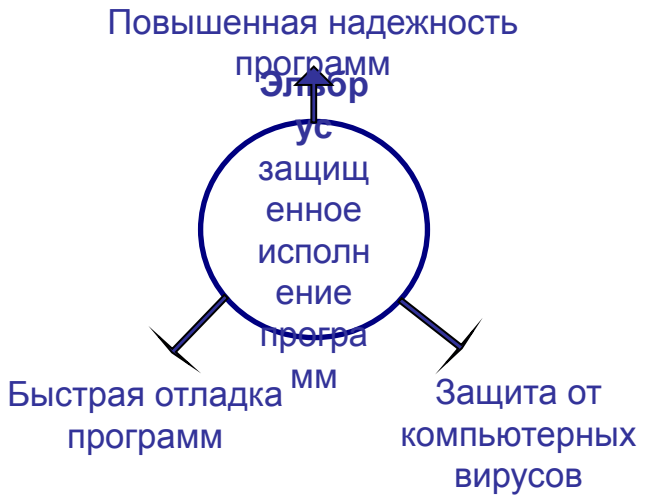
Нет аппаратной защиты от вирусов

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

Защищенное исполнение программ

Защищенное исполнение программ

- Защита памяти с помощью тегов
  - Структурированная память
  - Доступ к объектам через дескрипторы
  - Контекстная защита по областям видимости
- Обнаруживает критические уязвимости
  - Переполнение буфера
  - Неинициализированные данные
  - Обращение по зависшим ссылкам
  - Нарушение стандартов языков
- Эффективность исполнения (80% от нативной) за счет аппаратной поддержки



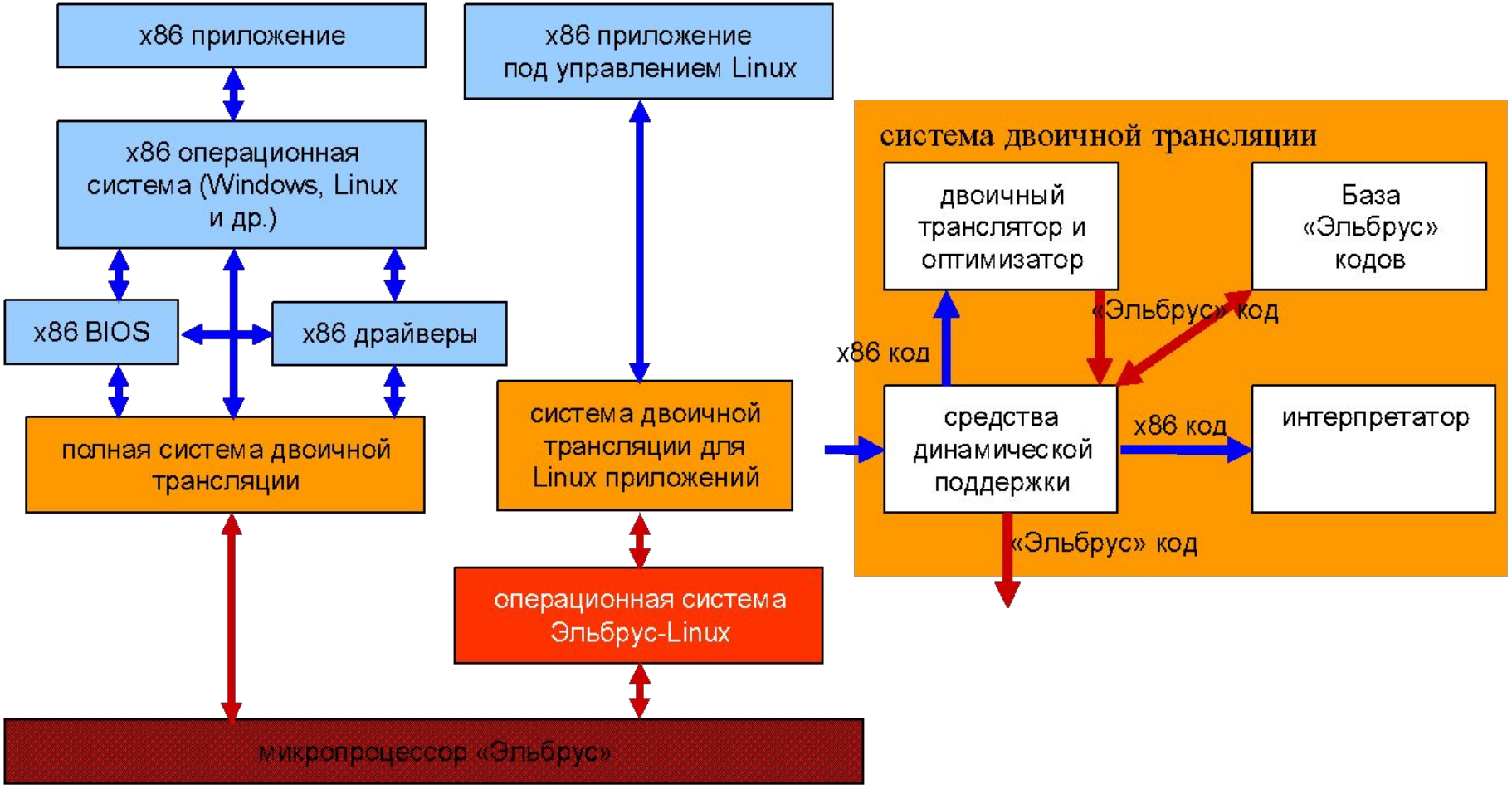
Задачи	Всего задач	Задач с найденными ошибками
Пакет SPEC95	18	11
Пакет SPECcpu2000	26	14
Пакет SPECcpu2006	29	6
Пакет негативных тестов *samate* на защищенность	888	874 (14 не нарушают защиту)

Технология не имеет аналогов в мире, обеспечивает конкурентные технологические преимущества перед импортными МП

МЦСТ
ЭЛЬБРУС

Обеспечение совместимости на базе МП линии «Эльбрус»

Двоичная трансляция для совместимости



Обеспечивает эффективную совместимость за счет использования параллелизма

Параллелизм операций (5)

Последовательный код

1
3
8
2
9
16
4
11
20
22
5
6
12
17
7
13
18
15
10
14
19
21

22 такта

Аппаратный Планировщик Intel x86

Параллельный код Intel

1			
3			
2	8		
4	7	9	
5	16	11	10
6	20	15	14
22	12	13	19
17			
18			
21			

10 тактов

Параллельный код Эльбруса

1	2	3	4		
5	6	7	8	9	10
11	12	13	14	15	16
17	18	19	20		
21	22				

5 тактов

В 2 раза быстрее Intel

Скрытый двоичный транслятор

Выполняется быстрее Intel в 1,66

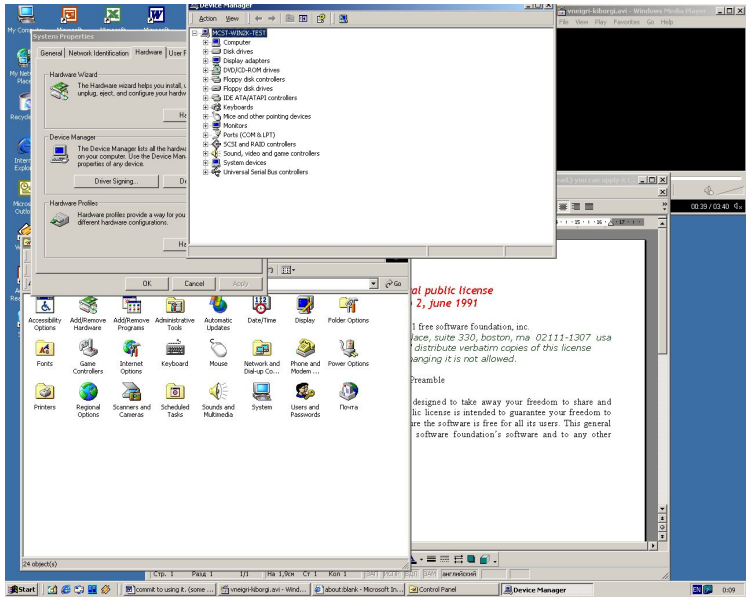
Параллельный код Эльбруса

1	3	2	4		
5	6	7	8	9	10
11	12	13	14	15	16
19	20				
22	17				
18	21				

6 тактов

МЦСТ ЭЛЬБРУС Эффективная двоичная совместимость с Intel x86, x64

- Функциональность
  - Полная совместимость с архитектурой Intel x86 (x86-64 с МП Эльбрус-4С)
  - Прямое исполнение 20+ операционных систем, в том числе: Windows XP, Windows 7, 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»
  - Логическая скорость (производительность при равных тактовых частотах) выше Intel Core 2 на пакете SPECcpu2006
- Лицензионная независимость от Intel



МЦСТ
ЭЛЬБРУС

Изделия на базе микропроцессоров с архитектурой «Эльбрус»

Изделия на базе МП Эльбрус-4С

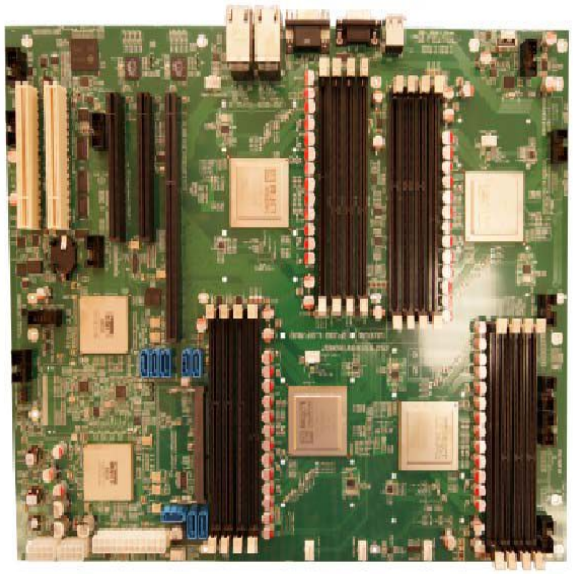
Сервер приложений

Настольный компьютер

Система хранения данных

Сервер баз данных

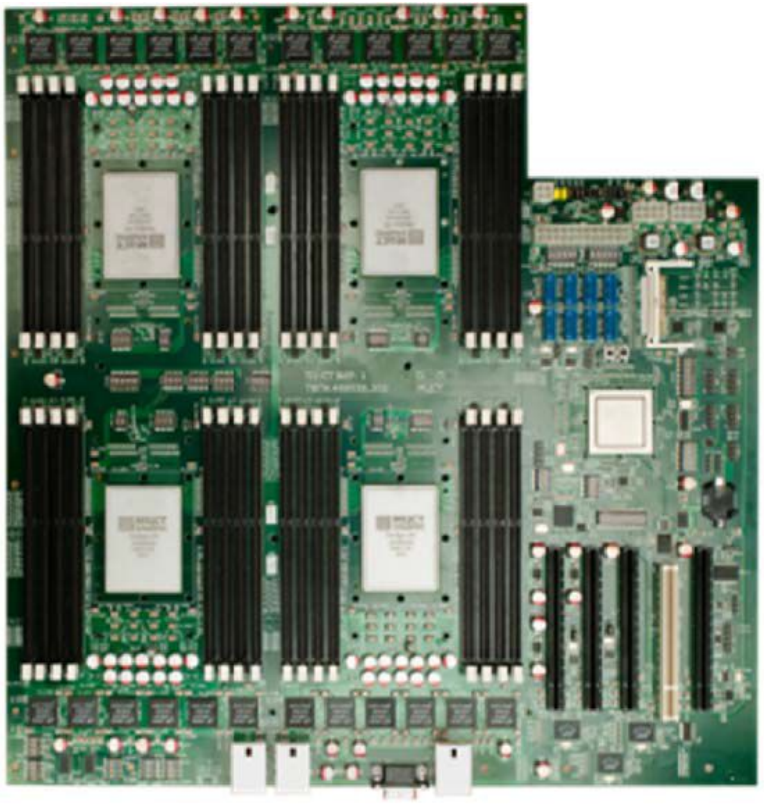
Сервер и шкаф на МП Эльбрус-4С с воздушным охлаждением



Характеристики сервера (узла) и шкафа

- Производительность, Тфлопс – 0,1 / 6,4
- Объем памяти DDR3, Гбайт – 96 / 6144
- Межузловые связи – 2D тор
- Мощность шкафа – 18 КВт

Сервер на МП Эльбрус-8С с воздушным охлаждением



Характеристики сервера (узла) и шкафа

- Производительность, Тфлопс – 0,5 / 16
- Объем памяти DDR3, Гбайт – 256 / 8192
- Межузловые связи – 10 (40) Gb Ethernet | Infiniband | СМПО
- Мощность шкафа – 15 КВт

ОПО Эльбрус

- Собственная программа начального старта (BIOS)
- Ядро базируется на ОС Linux со встроенными средствами защиты
  - Обеспечивает работу в режиме реального времени
  - Поддерживает
    - систему совместимости для приложений в кодах Intel x86
    - эффективное защищенное исполнение программ
- Современные средства разработки программ
  - Оптимизирующие компиляторы с языков C, C++, Fortran, Java, C#, JavaScript, средства сборки, отладки, профилирования, библиотеки
    - Новые средства глобального анализа и динамической оптимизации программ
    - Средства и библиотеки для распараллеливания
      - автоматическое распараллеливание под архитектуру (на уровне скалярных и упакованных операций, потоков управления)
      - Высокопроизводительные библиотеки оптимизированы под Эльбрус
      - Библиотека MPI, расширения OpenMP,
  - Возможность использования свободного ПО
    - совместимость с Гну-компиляторами
- Дистрибутив операционной системы Linux Debian
  - Утилиты, сервисы, библиотеки общего назначения
  - Графическая подсистема, работа с сетью, работа с СУБД, СХД, офисные пакеты, работа с периферийными устройствами
  - Управление ресурсами кластера

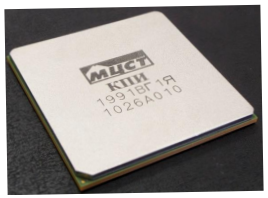
Все инфраструктурное ПО создается российскими разработчиками

МЦСТ
ЭЛЬБРУС

Развитие МП линии «Эльбрус» и компьютеров на них

Развитие серверных МП линии «Эльбрус»

Эльбрус-4С		Эльбрус-8С		Эльбрус-8СВ		Эльбрус-16С
0.8 ГГц, 4 Я	2 года	1.3 ГГц, 8 Я	3 года	1.5 ГГц, 8 Я	3 года	2.0 ГГц, 16 Я
3*DDR3-1600		4*DDR3-1600		4*DDR4-2400		4*DDR4-3200
50 Gflops sp	**4-5x**	250 Gflops sp	**2x+**	580 Gflops sp	**2x+**	1500 Gflops sp
45 Вт		~60…90 Вт		~60…90 Вт		~90…110 Вт
65 nm		28 nm		28 nm		16 nm
2013		2015		2018		2021



Контроллеры периферийных интерфейсов КПИ-1 и КПИ-2

Контроллеры периферийных интерфейсов встроены в МП

На базе новых МП проектируются компьютеры и программное обеспечение

МЦСТ ЭЛЬБРУС

МП Эльбрус-8СВ

- ❑ Госконтракт с Минпромторгом РФ.
- ❑ Сроки завершения: 2018 г.
- ❑ стадия разработки – готовность к первому tapeout в 2017
- ❑ Характеристики МП:
- ✔ производительность - до 580 / 290 Gflops (sp / dp);
- ✔ количество ядер – 8;
- ✔ тактовая частота – 1,5 ГГц;
- ✔ ОЗУ – DDR4-2400, четыре канала (до 76,8 ГБ/с)
- ✔ канал ввода-вывода: 16 Гбайт/с (дуплекс), использует КПИ-2 для связи с внешними устройствами
- ✔ до 4 микропроцессоров с общей памятью
- ✔ потребляемая мощность ~75 Вт;
- ✔ технология – 28 нм;
- ✔ количество транзисторов > 3 млрд;
- ✔ Площадь 332 кв. мм

МП Эльбрус-8СВ может размещаться на таких же модулях, что и Эльбрус-8С, повышая их производительность в 2+ раза

Решения на базе технологий «РСК-торнадо»

Масштабируемая серверная система с водяным охлаждением на Э8С / Э16С

- Производительность, Тфлопс – 200+ / 1200+
- Объем памяти, Тбайт – 100+ DDR3 / 400+ DDR4
- Межузловые связи – Infiniband | СМПО / СМПО
- Мощность шкафа, кВт – 200 / 300

Особенности МП Эльбрус-16С

Основные технологические нововведения

- вся система на одном кристалле, включая контроллеры периферийных устройств
- поддержка виртуализации, в том числе в кодах Intel x86-64
- масштабируемая векторизация
- аппаратная поддержка динамической оптимизации (рост производительности ядра)

Характеристики МП:

- ✔ производительность - до 1500 / 750 Gflops (sp/dp);
- ✔ количество ядер – 16;
- ✔ тактовая частота – 2 ГГц;
- ✔ Кэш-память (L2 + L3) – 40 Гбайт
- ✔ ОЗУ – DDR4, четыре канала (до 102 ГБ/с)
- ✔ Система на кристалле включает: PCIe 3.0, 1/10 Gb Ethernet, SATA 3.0, USB 3.0
- ✔ до 4 микропроцессоров с общей памятью
- ✔ до 48 ГБ/с межпроцессорный обмен
- ✔ потребляемая мощность ~90 Вт;
- ✔ технология – 16 нм;
- ✔ количество транзисторов ~ 6 млрд;
- ✔ Площадь ~400 кв. мм

Сроки завершения ОКР: 2021 г., продукция – с 2022 г.

Может использоваться для создания суперкомпьютера свыше 100 петафлопс

МЦСТ
ЭЛЬБРУС

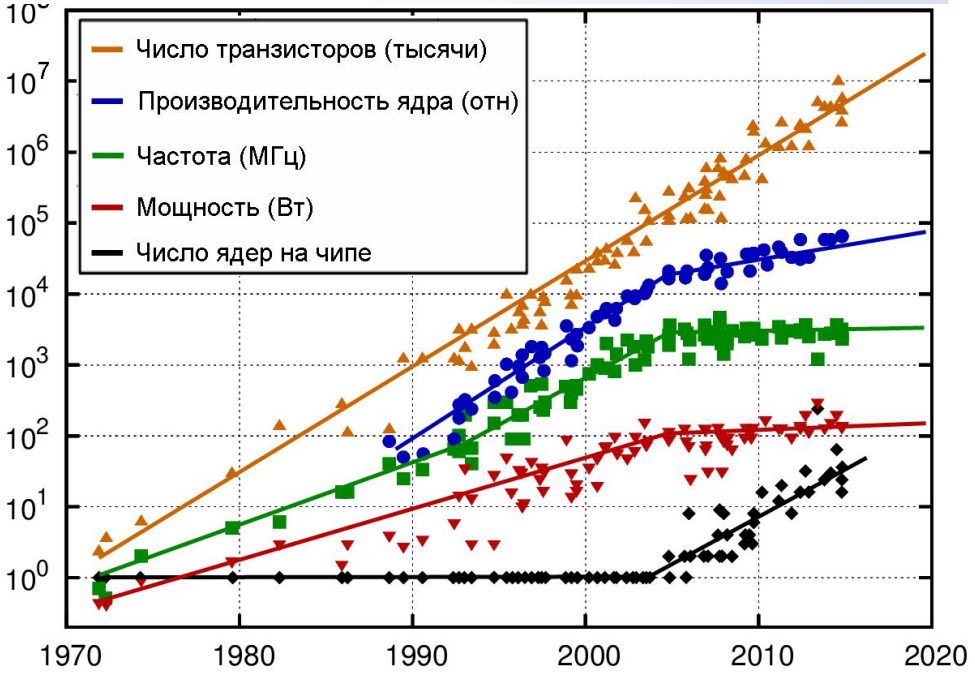
СПАСИБО за внимание!

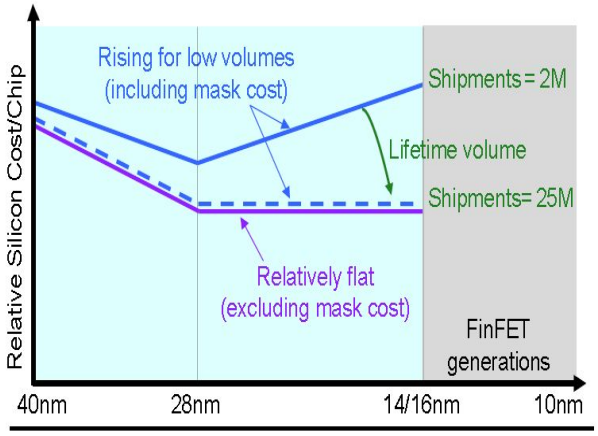
Проблемы кремниевой технологии

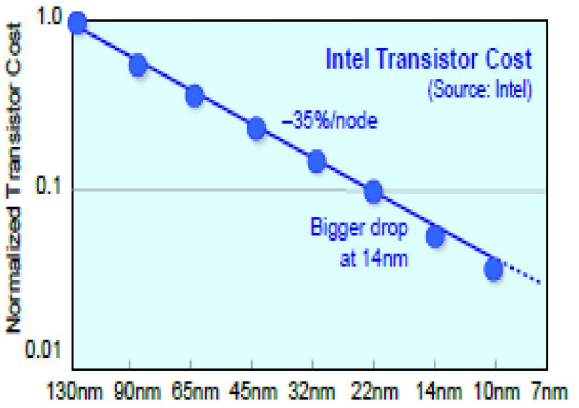
«Закон» Мура дает сбои

- с середины нулевых остановился рост тактовой частоты из-за мощности и замедлился рост производительности процессора
- с середины нулевых прирост производительности, в основном, за счет увеличения числа ядер (транзисторов)
- продолжает удваиваться число транзисторов за 2 года
- стоимость транзистора для всех фабрик ниже 28 нм, кроме Intel, перестает падать, что ведет к удорожанию чипов

Тем не менее до 2030 г. замены кремнию пока не видно







Нужно искать новые архитектурные решения в логике микропроцессора, чтобы сохранить поступательный рост производительности

Сравнение на пакете SPECcpu2006 на ядро