

ПРОЦЕССОРЫ ЭЛБРУС НА АРХИТЕКТУРЕ ЭЛЬБРУС 2000 Е2К: ОСОБЕННОСТИ И ПРОИЗВОДИТЕЛЬНОСТЬ

Докладчик: Садовой Григорий Владимирович Р3207

Краткая история

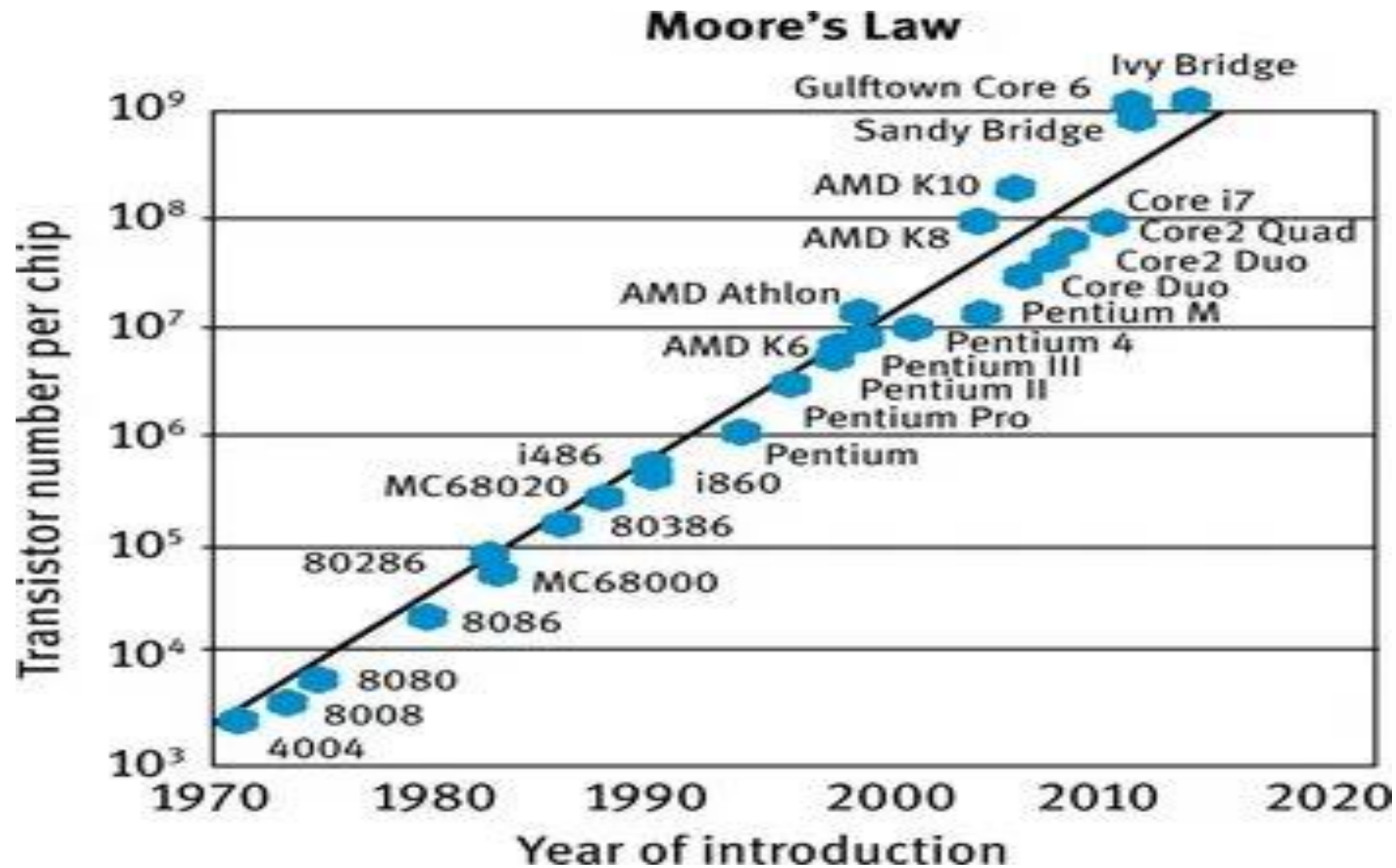
- 1978 год: Эльбрус 1 – 15 млн флопс, 64 М RAM – опирался на опыт создания БЭСМ-6.
- 1980 год: Эльбрус-2 – 8 процессоров, 125 млн флопс, 144М RAM.
- 1985 год: Начата разработка Эльбрус-3. Серийное производство провалилось из-за распада СССР.
- 2014 год: Первый процессор серии Э2К.

Семейство Эльбрус Е2К

- 16 ядер
- 2ГГц
- 1500 Гфлопс
- До 50 команд на такт
- Шина 200 Гбайт/с



Закон Мура



VLIM

- VLIM (Very Long Instruction Word) – архитектура процессоров характеризующаяся возможностью объединения нескольких простых команд в так называемую связку. Входящие в нее команды должны быть независимы друг от друга и выполняться параллельно. Таким образом, из нескольких машинных команд компилятор формирует одно очень длинное командное слово.
- Идея VLIM базируется на том, что задача эффективного планирования параллельного выполнения команд возлагается на сложный компилятор.

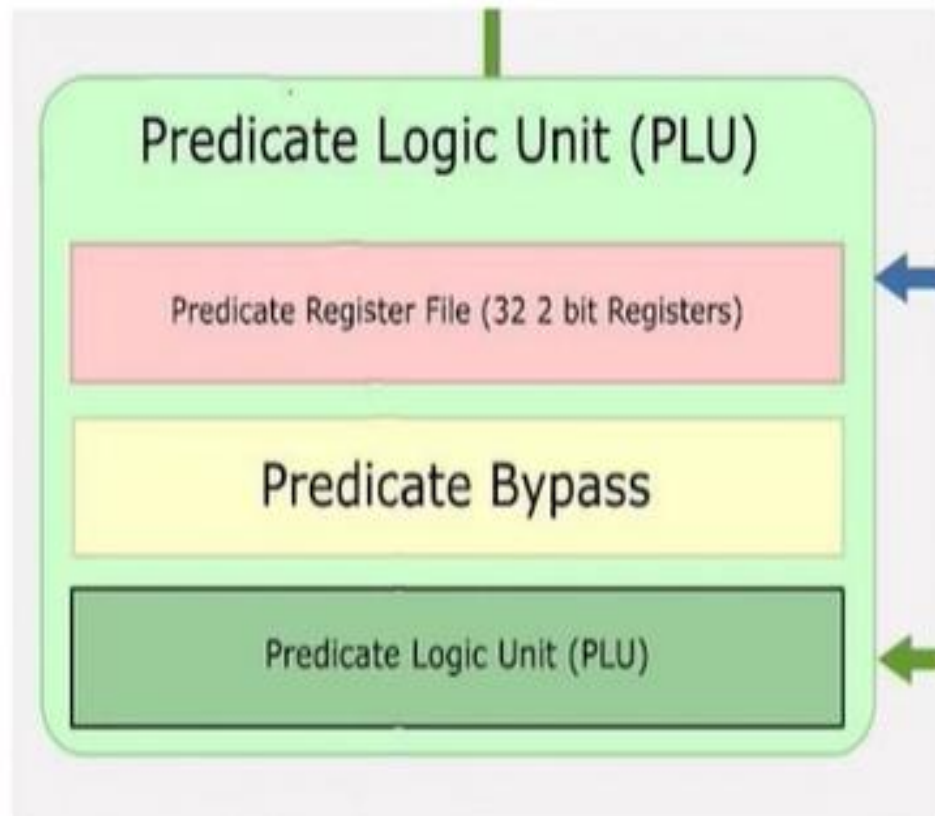
VLIM в Эльбрусе

Под "широкой командой" в архитектуре Эльбрус понимается набор элементарных операций, которые могут быть запущены на исполнение в одном такте.

В ЦК «Эльбрус» доступны:

- **6 арифметико-логических устройств (АЛУ)**, исполняющих операции:
 - целочисленные (Int)
 - вещественные (FP)
 - сравнения (Cmp)
 - чтения из памяти (LD)
 - записи в память (ST)
 - операции с упакованными векторами (Vect)
 - деления и квадратного корня (Div/Sqrt)
- **1 устройство для операций передачи управления (CT)**
- **3 устройства для операций над предикатами (PL)**
- **6 квалифицирующих предикатов (QP)**
- **4 устройства для команд асинхронного чтения данных по регулярным адресам в цикле (APB)**
- **4 литерала размером 32 бита для хранения константных значений (LIT)**

Предикаты



Логический предикат (ЛП) — результат операции сравнения двух чисел (целых или вещественных), выполняемой в арифметико-логических каналах. Он представляется двухразрядным составным значением, содержащим: Тег (0 — обычный предикат, 1 — диагностический предикат), Булево значение (0 — false, 1 — true)

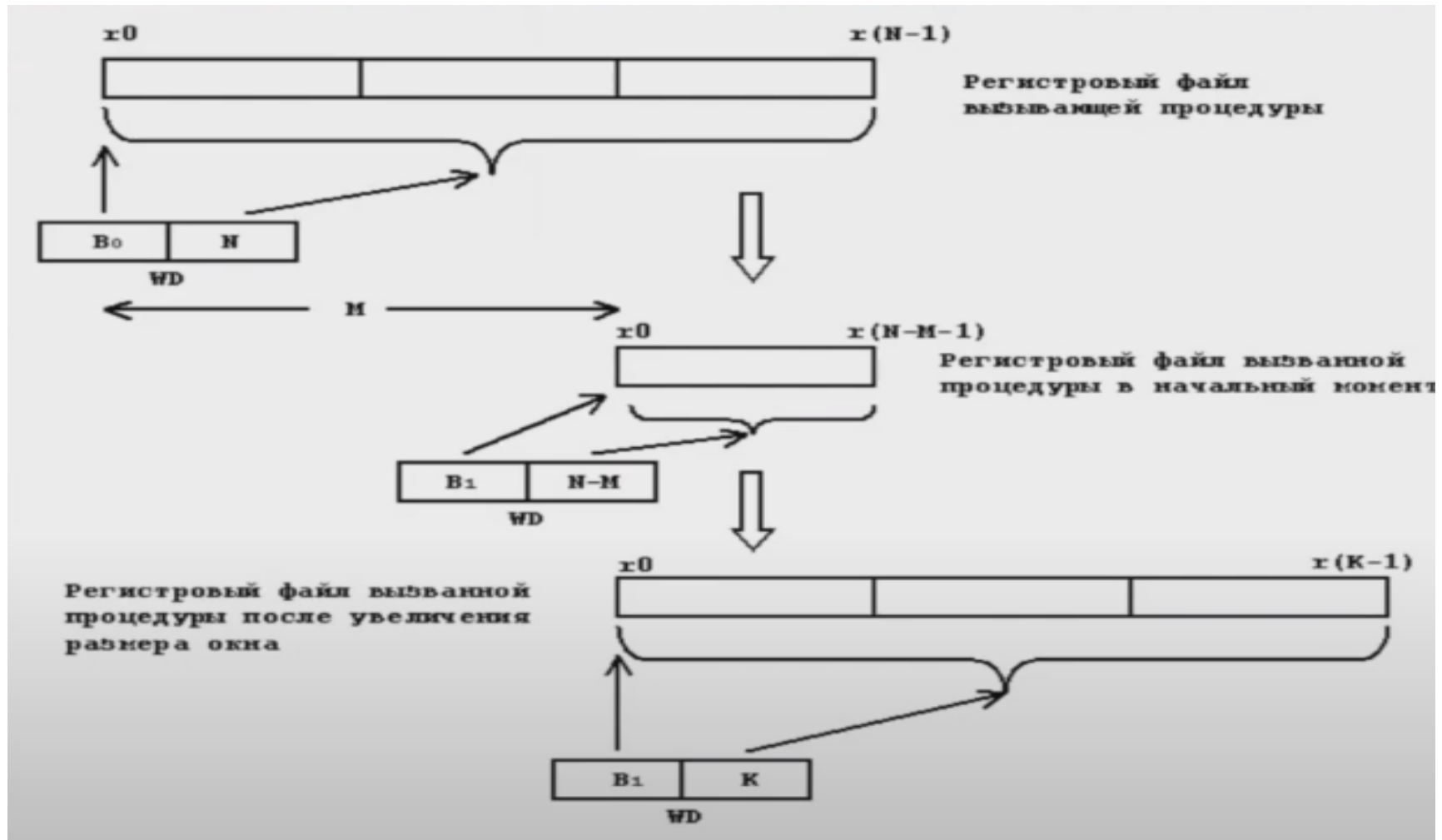
Устройство логических предикатов (УЛП или PLU - Predicate Logic Unit) разработано с целью:

1. Разгрузки арифметических устройств
2. Выполнения операций над малоформатными значениями предикатов
3. Управления действиями арифметико-логического канала

Функции УЛП:

- Синтез предикатов из предикатного файла
- Вычисление вторичных предикатов по двум первичным
- Задание предикатного (условного) выполнения операций
- Запись предикатов в предикатный файл (32 2-битных регистра)

Регистровые окна



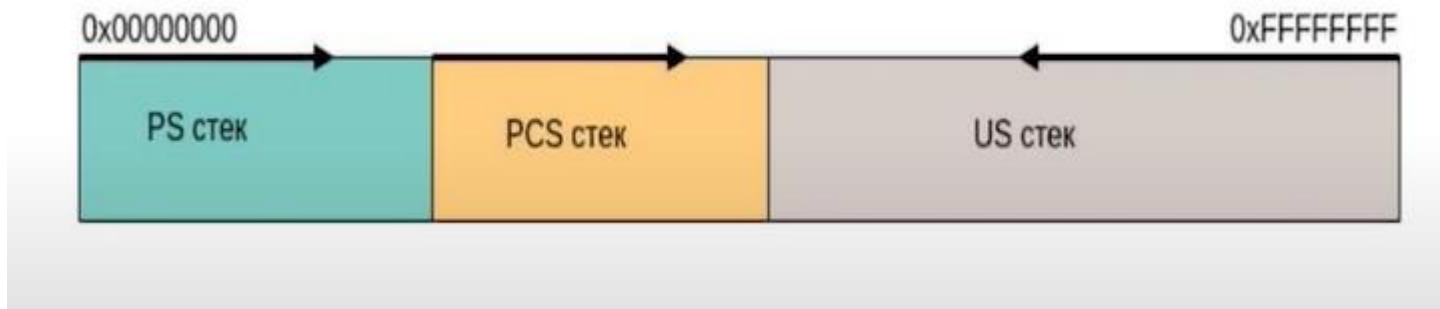
Подкачка данных для цикла

- Процессоры Э2К способны выполнять циклы с наложением итерации – одна или несколько итерации цикла могут начать исполнение до того, как предыдущая итерация завершилась.
- Для этого в регистровом и предикатном файле может быть организован конвейер регистров. Несмотря на то что код всех итераций будет обращаться к одному и тому же регистру, процессор будет выделять каждой новой запущенной итерации новый регистр из конвейера.

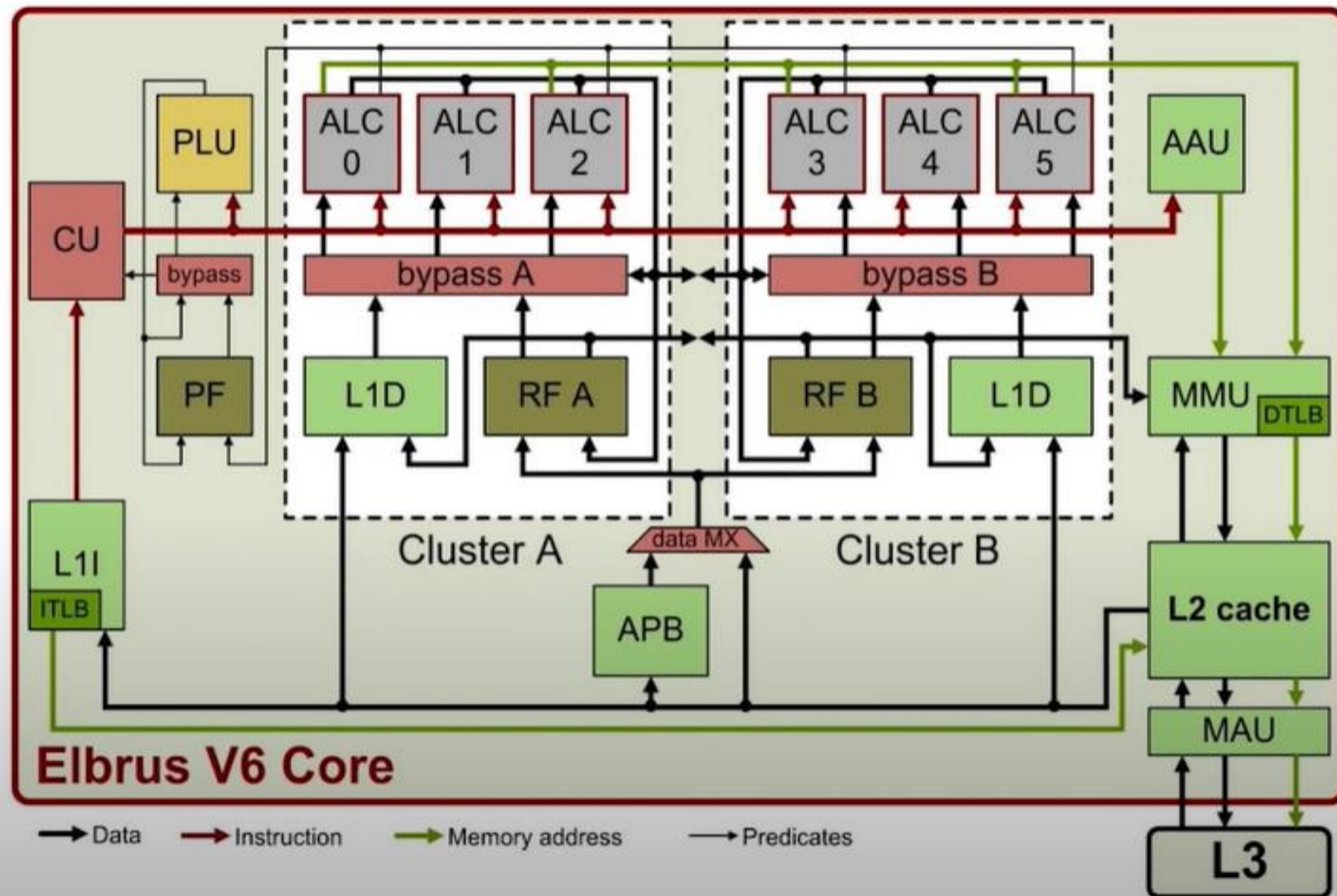
Стеки

В архитектуре Эльбруса различают следующие разновидности стеков:

- Стек процедур (Procedure Stack)
- Стек пользователя (User Stack)
- Стек связующие информации (Procedure Chain Stack)



Ядро



MAU – Memory Access Unit
MMU – Memory Management Unit
AAU – Array Access Unit
CU – Control unit
IB – Instruction Buffer
ALC – Arithmetic Logic Channel
TLB – translation lookaside buffer
PLU – Predicate Logic Unit
RF – Register file
APB – Array Prefetch Buffer

Пример кода

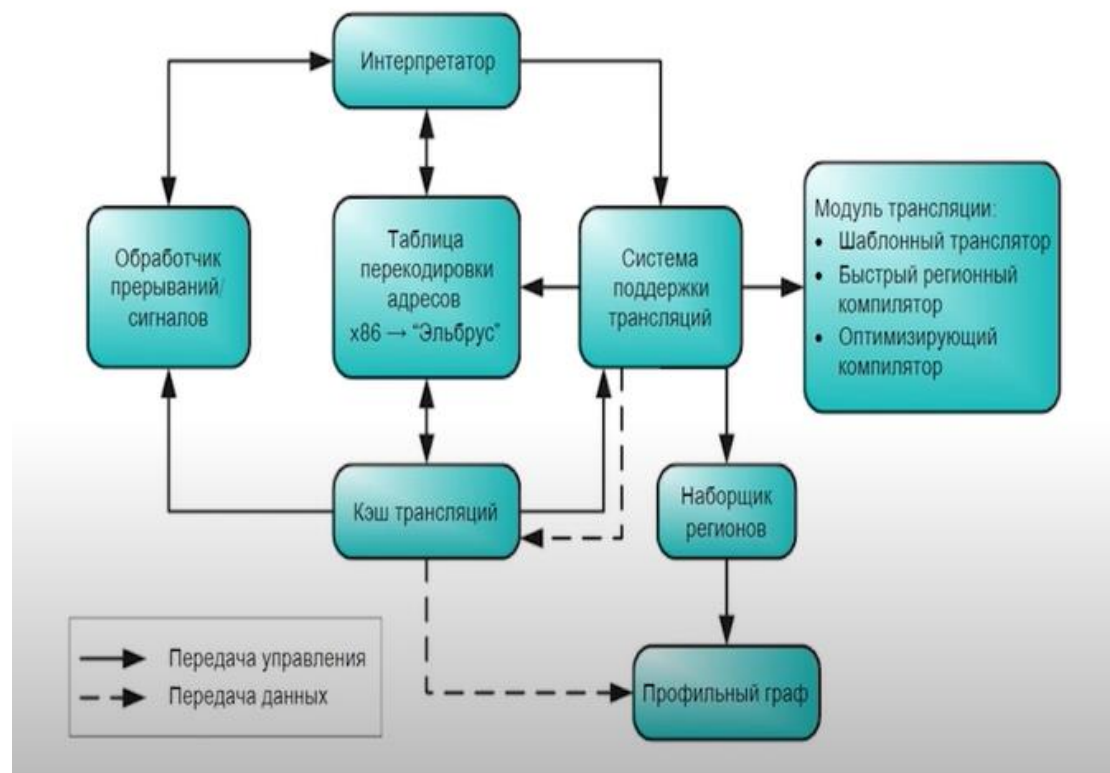
```
void a(int n, int a, int b, int c, int *x) {  
    for(int i = 0; i < n; i++) {  
        x[i] = (x[i] + a) * b + c;  
    }  
}
```

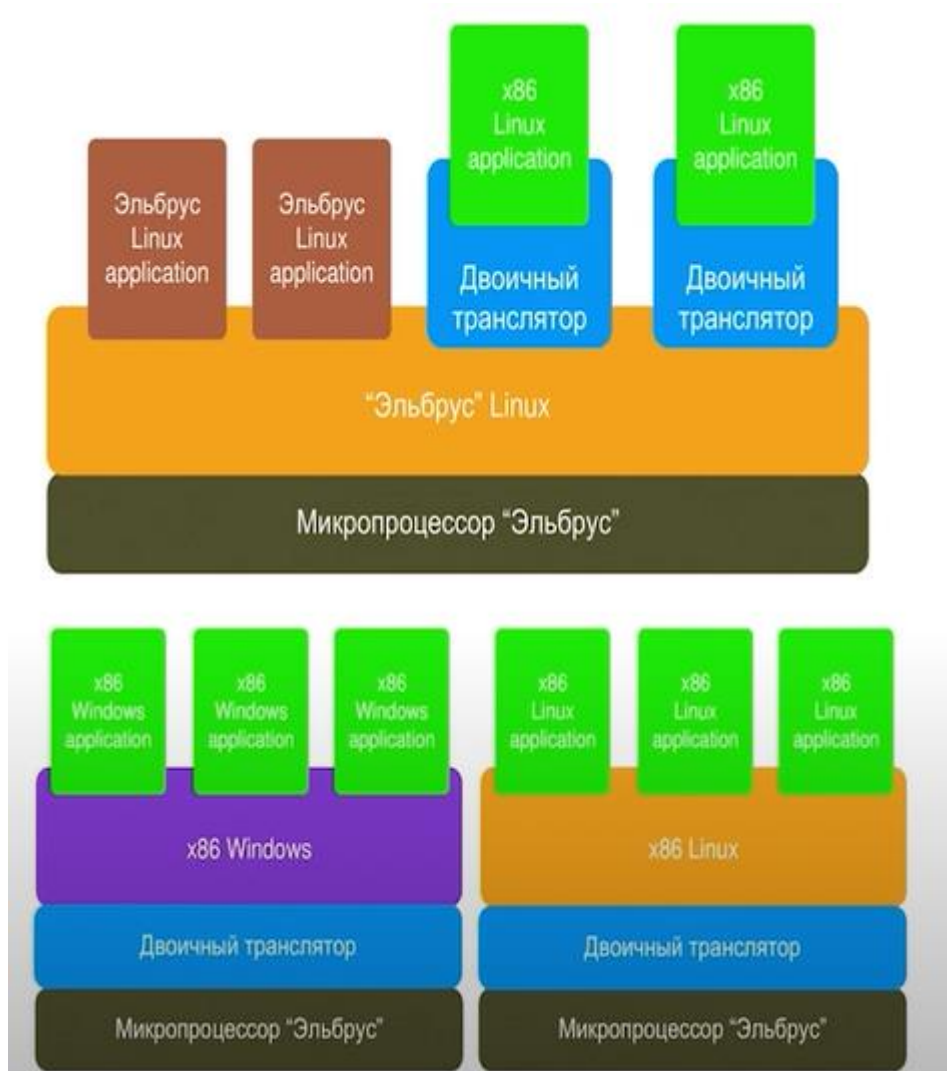
```

a(int, int, int, int, int*):
{
    setwd wsz = 0x6, nfx = 0x1, dbl = 0x1
    return %ctpr3
}
{
    cmplsb,0 0x0, %r0, %pred0
}
{
    merges,0,sm 0x1, %r0, %r9, %pred0
    addd,1 0x0, 0x0, %dr10 ? %pred0
}
{
    shrs,0,sm %r9, 0x1, %r8
}
{
    ct %ctpr3 ? ~%pred0
    cmpesb,0,sm %r8, 0x0, %pred1
    cmplsb,1,sm 0x0, %r8, %pred2
}
{
    nop 2
    return %ctpr3
    muls,0,sm %r1, %r2, %r7
    merges,1,sm 0x1, %r8, %r0, %pred2
    pass %pred0, @p0
    pass %pred1, @p1
    landp @p0, @p1, @p4
    pass @p4, %pred0
}
{
    ibranch .L416 ? %pred0
}
{
    setwd wsz = 0x11, nfx = 0x1, dbl = 0x1
    setbn rsz = 0xa, rbs = 0x6, rcur = 0x0
    disp %ctpr1, .L20
    addd,0 %dr4, 0x4, %dr6
    addd,1 0x0, _f64, _lts1 0x40ff2000000000, %dg16
    addd,2 0x2, 0x0, %dg17
    addd,3 0x0, 0x0, %r11
}
{
    return %ctpr3
    addd,0,sm 0x0, 0x0, %db[11]
    insfd,1 %dg16, _f32s, _lts0 0x8800, %dr0, %dg16 .L20:
    aaurwd,2 %dr10, %aasti1
    aaurwd,5 %dr4, %aad0
}
{
    addd,0,sm %db[11], _f16s, _lts0lo 0x10, %dg18
    addd,1,sm 0x8, %db[11], %dg17
    aaurwd,2 %dg17, %aaincr1
}
{
    ldw,0,sm %dr4, %db[11], %g19
    addd,1,sm %db[11], _f16s, _lts0lo 0x18, %dg21
    ldw,2,sm %dr6, %db[11], %g20
    addd,3,sm %db[11], _f16s, _lts0hi 0x20, %dg22
    addd,4,sm %db[11], _f16s, _lts1lo 0x28, %dg23
    addd,5,sm %db[11], _f16s, _lts1hi 0x30, %db[21]
}
{
    ldw,0,sm %dr6, %dg18, %g18
    andd,1 %dg16, _f64, _lts0 0xffffffff, %dg26
    ldw,2,sm %dr4, %dg17, %g24
    ldw,3,sm %dr6, %dg17, %g17
    addd,4,sm 0x8, %db[21], %db[19]
    ldw,5,sm %dr4, %dg18, %g25
}
{
    rwd,0 %dg16, %lsr
    shld,1 %dg26, 0x1, %dr10
    ldw,2,sm %dr4, %dg21, %b[7]
    ldw,3,sm %dr6, %dg21, %b[8]
    ldw,5,sm %dr4, %dg22, %b[5]
}
{
    rwd,0 %dr0, %lsr1
    adds,1 %r3, %r7, %r0
    ldw,2,sm %dr6, %dg22, %b[6]
    ldw,3,sm %dr4, %dg23, %b[3]
    ldw,5,sm %dr6, %dg23, %b[4]
}
{
    ldw,0,sm %dr6, %db[21], %b[2]
}
{
    muls,0,sm %g19, %r2, %b[15]
    muls,1,sm %g20, %r2, %b[16]
}
{
    nop 4
    muls,0,sm %g24, %r2, %b[13]
    muls,1,sm %g17, %r2, %b[14]
    muls,3,sm %g25, %r2, %b[11]
    muls,4,sm %g18, %r2, %b[12]
}
{
    loop_mode
    muls,0,sm %b[7], %r2, %b[9]
    muls,1,sm %b[8], %r2, %b[10]
    ldw,2,sm %dr4, %db[21], %b[1]
    ldw,3,sm %dr6, %db[19], %b[0]
    adds,4,sm %r0, %b[15], %b[20]
}
    adds,5,sm %r0, %b[16], %b[18]
}
{
    loop_mode
    alc alcf=1, alct=1
    abn abnf=1, abnt=1
    ct %ctpr1 ? %NOT_LOOP_END
    staaw,2 %b[20], %aad0[ %aasti1 ]
    addd,4,sm 0x8, %db[19], %db[17]
    staaw,5 %b[18], %aad0[ %aasti1 + _f32s, _lts0 0
    incr,5 %aaincr1
}
{
    setwd wsz = 0x6, nfx = 0x1, dbl = 0x1
    shld,0,sm %dr10, 0x2, %dr0
    shls,1 %r8, 0x1, %r6
}
{
    aaurw,2 %r11, %aabf0
}
.L82:
{
    nop 2
    ldw,0,sm %dr4, %dr0, %g16
    cmpesb,1 %r6, %r9, %pred0
}
{
    ct %ctpr3 ? %pred0
}
nop
{
    adds,0,sm %g16, %r1, %g16
}
{
    nop 5
    muls,0,sm %g16, %r2, %g16
}
{
    adds,0,sm %g16, %r3, %g16
}
{
    ct %ctpr3 ? ~%pred0
    stw,2 %dr4, %dr0, %g16 ? ~%pred0
}
.L416:
{
    ibranch .L82
    shld,0,sm %dr10, 0x2, %dr0
    shls,1 %r8, 0x1, %r6
}
elbrus_optimizing_compiler_v1.28.09_Nov_28_2023 = 0x0

```

Бинарный транслятор, RTC, intel





Сравнения и тесты

Intel I7 2600 вышел в 2011


Эльбрус 8С вышел в 2016

| Slower than Core i7 2600 [n times] running with same frequency | | | | | | |
|----------------------------------------------------------------|------|-----|------------|-----|--------|------|
| Cpu | Java | C# | JavaScript | Php | Python | Lua |
| Single-thread | | | | | | |
| Elbrus 2C+ 500 | | | | 2.3 | 4.5 | 2.4 |
| Elbrus 1C+ 985 | 3.5 | 4.5 | 5 | 2.3 | 3.6 | 1.75 |
| Elbrus 4C 750 | 2.5 | 4.2 | 2.75 | 1.5 | 3.5 | 2 |
| Elbrus 8C 1300 | 2 | 3 | 2.5 | 1.5 | 3.5 | 2.3 |
| Elbrus 8C 1300 RTC | | 1.3 | | 2.5 | 2.6 | |
| Elbrus 8CB 1550 | 2 | 3 | 2.25 | 1.5 | 3.5 | 2.2 |
| Elbrus 8CB 1550 RTC | | 1.2 | | 2.7 | 2.5 | |
| Intel Core i7-2600 | 1 | 1 | 1 | 1 | 1 | 1 |
| Multi-thread | | | | | | |
| Elbrus 2C+ 500 | | | | | 8.5 | |
| Elbrus 1C+ 985 | 5 | 7 | | | 7.4 | |
| Elbrus 4C 750 | 2.75 | 3 | | | 3 | |
| Elbrus 8C 1300 | 1.15 | 1.5 | | | 1.5 | |
| Elbrus 8C 1300 RTC | | 0.8 | | | 2.8 | |
| Elbrus 8CB 1550 | 1.15 | 1.2 | | | 1.35 | |
| Elbrus 8CB 1550 RTC | | 0.8 | | | 2.3 | |
| | 1 | 1 | 1 | 1 | 1 | 1 |

| Slower than Core i7 2600 (3.4 GHz) [n times] | | | | | | |
|----------------------------------------------|------|------|------------|-----|--------|-----|
| Cpu | Java | C# | JavaScript | Php | Python | Lua |
| Single-thread | | | | | | |
| Elbrus 2C+ 500 | | | | 16 | 30 | 16 |
| Elbrus 1C+ 985 | 11 | 15.5 | 16 | 8 | 12.5 | 6 |
| Elbrus 4C 750 | 10 | 19 | 12.5 | 5 | 15.5 | 10 |
| Elbrus 8C 1300 | 5.5 | 10.5 | 6.5 | 3 | 9 | 6 |
| Elbrus 8C 1300 RTC | | 3.5 | | 7 | 7.5 | |
| Elbrus 8CB 1550 | 4.5 | 8 | 5 | 2.5 | 7.8 | 5 |
| Elbrus 8CB 1550 RTC | | 3 | | 6 | 5.6 | |
| Intel Core i7-2600 | 1 | 1 | 1 | 1 | 1 | 1 |
| Multi-thread | | | | | | |
| Elbrus 2C+ 500 | | | | | 58 | |
| Elbrus 1C+ 985 | 18 | 24 | | | 25 | |
| Elbrus 4C 750 | 12.5 | 12.5 | | | 13.5 | |
| Elbrus 8C 1300 | 3 | 4.5 | | | 4.2 | |
| Elbrus 8C 1300 RTC | | 2 | | | | |
| Elbrus 8CB 1550 | 2.5 | 4 | | | 3.8 | |
| Elbrus 8CB 1550 RTC | | 1.5 | | | 5.1 | |
| Intel Core i7-2600 | 1 | 1 | 1 | 1 | 1 | 1 |

Современное использование

- Сооснователь МЦСТ Борис Бабаян и его сын Евгений работают над созданием процессора на новой архитектуре «Эльбрус-Б». Чип планируют выпустить к 2027 году. По словам разработчиков, новая архитектура будет быстрее иностранных аналогов в 30–200 раз.



МЦСТ
ЭЛЬБРУС
Эльбрус-8CB
1891BM12Я
T6G703.00.01
1736


Цена: по запросу

Количество:

ХАРАКТЕРИСТИКИ

Микросхема центрального процессора 1891BM12Я — вычислитель серверного класса с усовершенствованным набором векторных команд. Содержит 8 ядер архитектуры «Эльбрус» 5-го поколения с тактовой частотой до 1500 МГц. Позволяет строить многопроцессорные серверы и рабочие станции, а также бортовые вычислители, требовательные к скорости обработки и передачи информации.

Технические характеристики товара могут отличаться от указанных на сайте, уточняйте технические характеристики товара на момент покупки и оплаты. Вся информация на сайте о товарах носит справочный характер и не является публичной офертой.





СТОЕЧНЫЕ СЕРВЕРЫ 19' АРХИТЕКТУРЫ

СЕРВЕРЫ НА ПРОЦЕССОРАХ ЭЛЬБРУС



ЯХОНТ-УВМ Б110

- 10 x SFF дисков
- 2 процессора
- 8 модулей DDR4
- 1+1 БП

[ПОДРОБНЕЕ](#)



ЯХОНТ-УВМ Б114

- 4 x LFF, 5xSSD
- 2 процессора
- 8 модулей DDR4
- 1+1 БП

[ПОДРОБНЕЕ](#)