Микропроцессоры. Базовая архитектура МП i80x86

МП – интегральная схема (*ИС, чип*) или группа ИС, реализующие функции обработки цифровой информации, управления процессом обработки и обмена с другими устройствами.

Все МП работают в соответствии с принципом программного управления фон-Неймана.

Различают:

- Однокристальные МП (с фиксированной разрядностью и системой команд) – большинство МП общего назначения (*general purpose*).

- Многокристальные МП (с изменяемой разрядностью и системой команд) – специализированные системы (*specialized systems*).

Характеристики МП:

алгоритмические, схемотехнические,

эксплуатационные:

- тип технологии
- кол-во кристаллов
- количество ядер
- система команд
- разрядность
- кэш, допустимая емкость ЗУ
- быстродействие
- потребляемая мощность
- тип сокета

Базовая архитектура МП Intel 80x86

Рассматриваемая далее архитектура характерна для всех МП фирмы Intel, начиная с 80**386**, в значительной степени присутствует и в первом МП этой линейки – 8086, а также МП-аналогах.

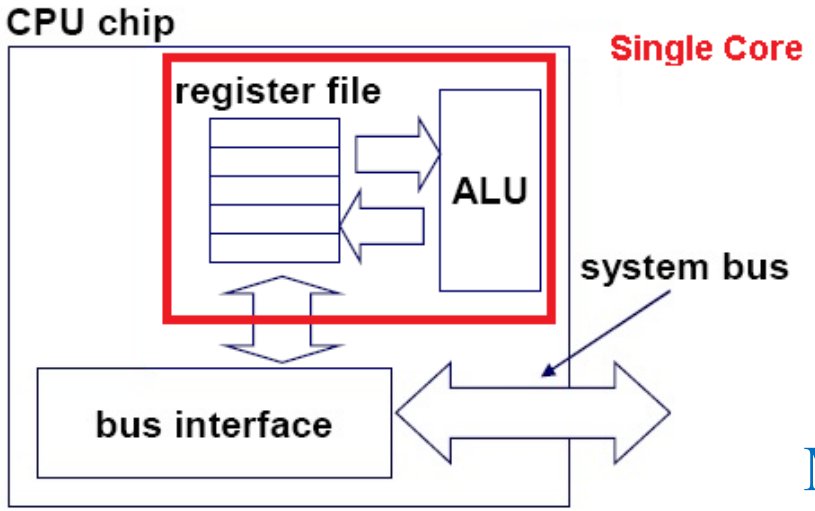
В архитектуре можно выделить 2 группы аппаратных средств:

- для прикладного программирования;
- для управления вычислительным процессом.

Условные обозначения

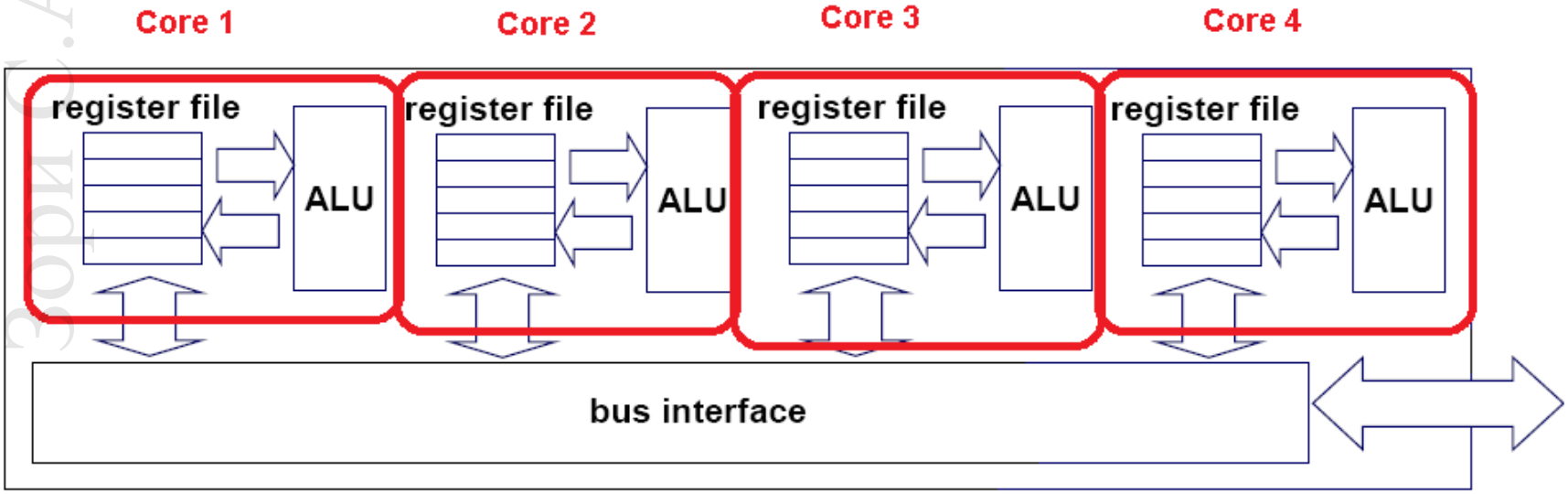
- **ЦП, CPU** – центральный процессор, central processor unit
- **АЛУ, ALU** – арифметико-логическое устройство, arithmetic logic unit
- **УУ, CU** – устройство управления, control unit
- **AGU** – устройство вычисления адреса ОЗУ
- **FPU** – floating point unit, арифметический сопроцессор
- **SIMD/AVX PU**– поточное /векторное вычислительное устройство

Эволюция МП 80x86



Многоядерность ↓

Chip Multi-Processor : runs multiple threads on each core aka Simulateous multithreading



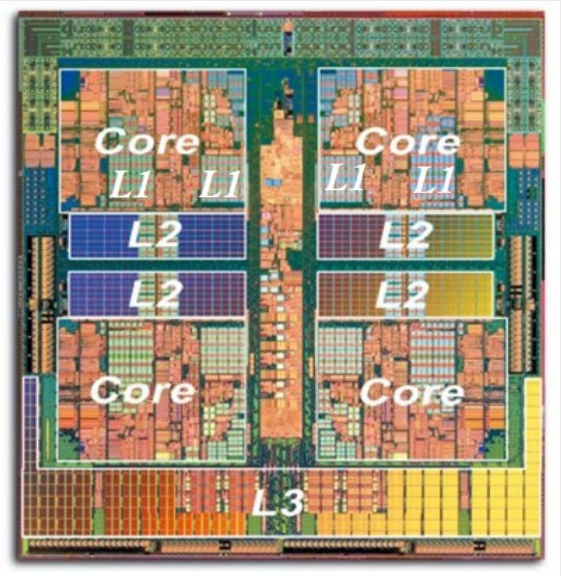
Классификация поколений МП x86

- 1 поколение — IA-16, 1 ядро.
- 2 поколение — IA-32, поддержка виртуальной памяти, L1.
- 3 поколение — конвейерность, FPU.
- 4 поколение — суперскалярность, целочисленные SIMD, конвейерный FPU.
- 5 поколение — L2, вещественные SIMD.
- 6 поколение — многоядерность, IA-64.
- 6+ поколение — L3, контроллер памяти, GPU.

Кэш МП 80x86

Уровни кэширования:

L1 (первый уровень),
L2 (второй уровень),
L3 (третий уровень).



- **Кэш первого уровня (L1)** – наиболее быстрый уровень кэш-памяти, который встроен в ядро процессора, данный уровень обладает наименьшим временем доступа и работает на частоте процессора. Дорогой. Является буфером между процессором и кэш-памятью второго уровня.

Кэш МП 80x86

- **Кэш второго уровня (L2)** – второй уровень на ядре, более масштабный, нежели первый, дешевле, но в результате, обладает меньшими «скоростными характеристиками». Соответственно, служит буфером между уровнем L1 и L3.
- **Кэш третьего уровня (L3)** – третий уровень, еще более медленный, нежели два предыдущих. Но всё равно он гораздо быстрее, нежели оперативная память. Если два предыдущих уровня разделяются на каждое ядро, то данный уровень является общим для всего процессора.

Архитектура процессора

В общем случае, МП можно рассматривать архитектурно *с точки зрения программиста* так:

архитектура ЦП

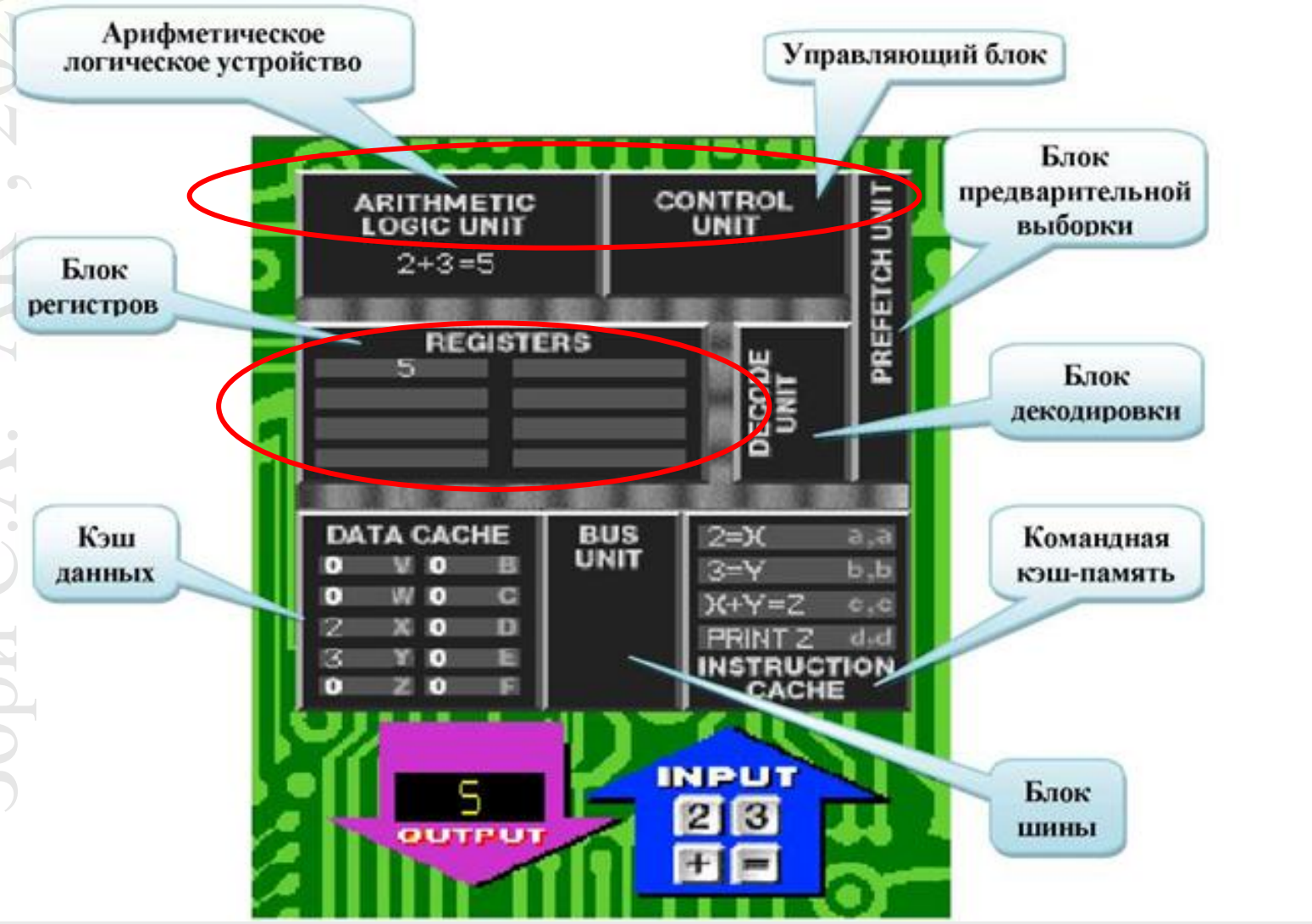
=

команды для работы с вычислительными устройствами (CPU, FPU, SIMD/AVX PU)

+

регистры (память)

Устройство CPU (укрупненно)



Устройство CPU (укрупненно)

1. Операционное устройство (ОУ, АЛУ)

- выполняет команды

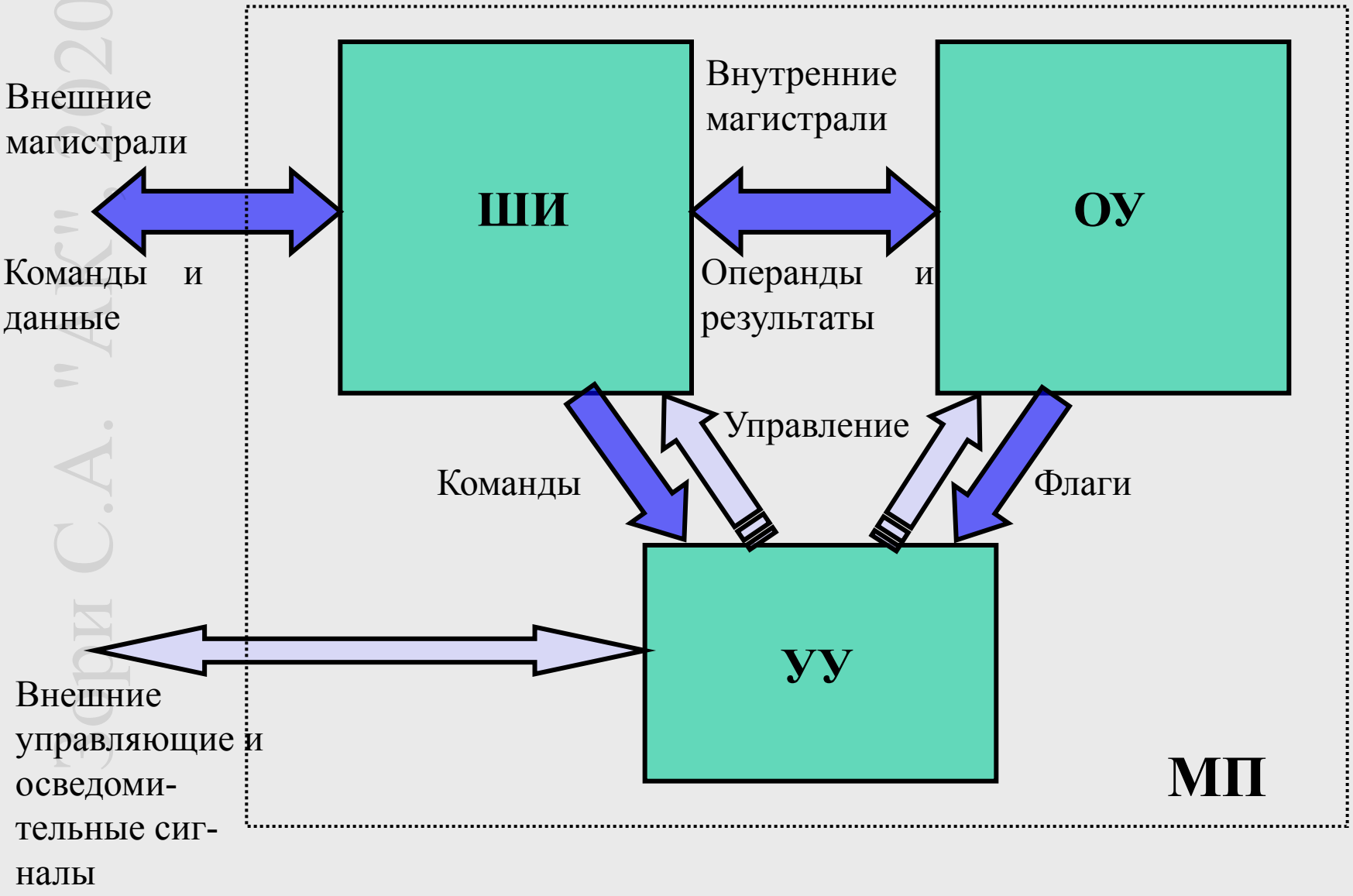
2. Шинный интерфейс (ШИ)

- выбирает команды из памяти
- читает операнды из памяти
- записывает результаты в память

3. Устройство микропрограммного управления

- управляет работой ОУ и ШИ, декодирует команды, анализирует флаги и управляет ходом программы, анализирует входные и генерирует выходные управляющие сигналы МП

Устройство CPU (укрупненно)



Устройства ОУ, ШИ и УУ работают параллельно во времени, что увеличивает производительность МП.

Для увеличения производительности ядра современные процессоры используют также:

- кэш*;
- конвейеризированную обработку*;
- суперскалярность*;
- встроенные FPU и SIMD Units*.

МП может работать в 2 основных режимах:

- реальный режим
- защищенный режим

В **реальном режиме** (режим работы i8086) МП работает в 16-разрядной системе команд и использует сегментные регистры при работе с памятью.

(В ОС Ms-DOS МП работает в реальном режиме).

В **защищенном режиме** МП работает в базовой, 32- (64-) разрядной системе команд, использует плоскую (логическую) модель памяти без применения сегментных регистров.

Память включает в себя ОЗУ и дисковую (виртуальную) память с единым механизмом адресации.

(В защищенном режиме МП работают в современных ОС – Windows, …).

• Для 16- разрядных ОС имеются специальные программные ср-ва для перевода МП в защищенный режим и возврата из него.

• Для защищенного режима также имеются программные ср-ва, позволяющие выполнять программы и переводить МП в реальный режим.

С точки зрения программы:

архитектура ЦП

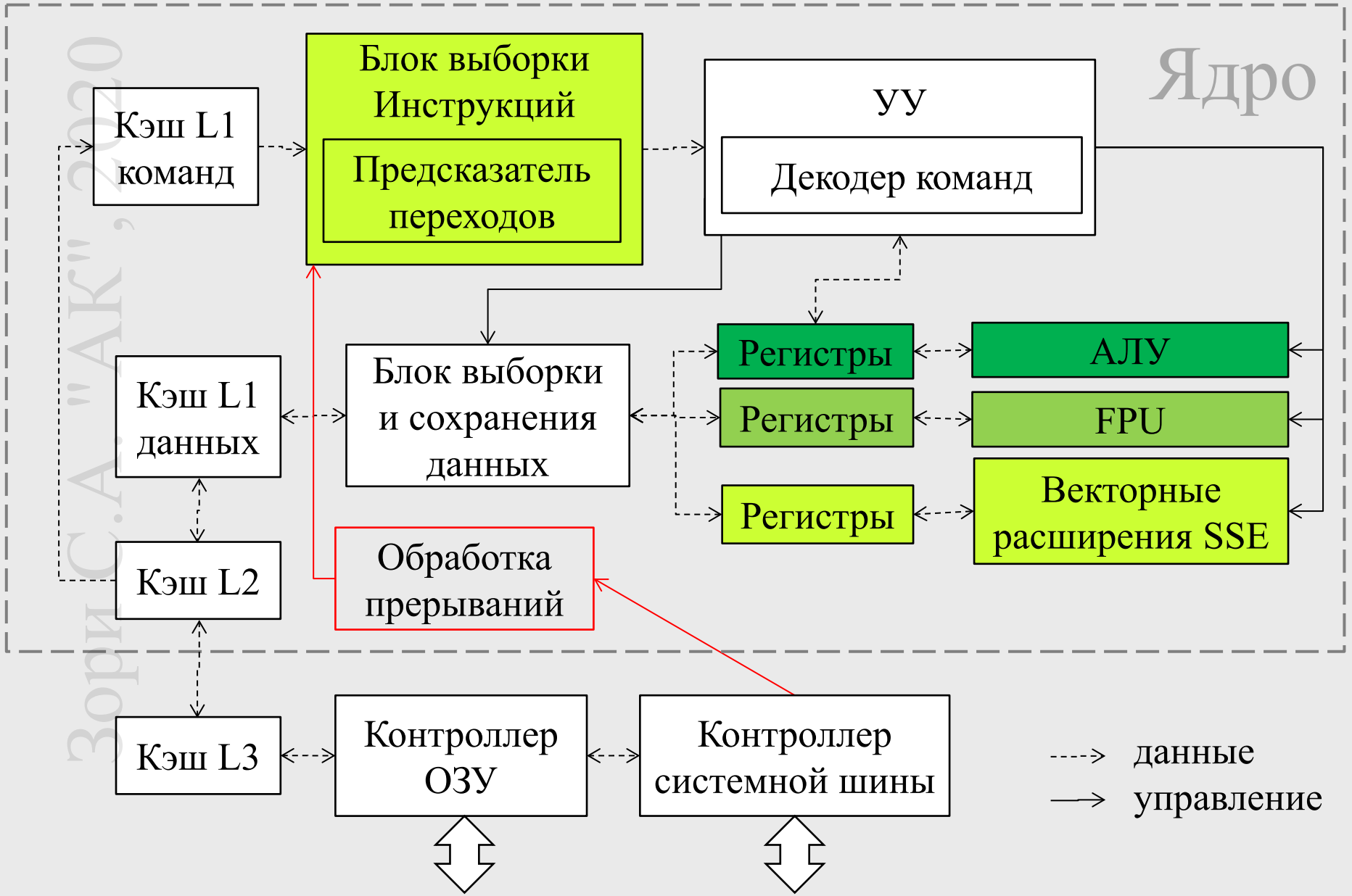
=

команды

+

регистры (память)

Что мы будем изучать?



В данном курсе ориентируемся на **32**-х разрядную архитектуру МП (IA-32)

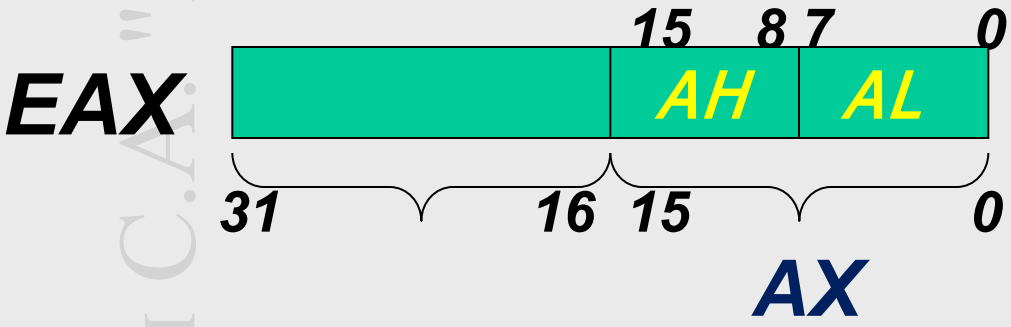
С точки зрения программиста МП содержит:

1. Блок **регистров общего назначения целочисленного* АЛУ** (в ОУ)

Общее назначение – хранение данных и адресов памяти.

Регистры микропроцессора – самая быстрая <u>память</u> (но очень маленькая), доступная пользователю.

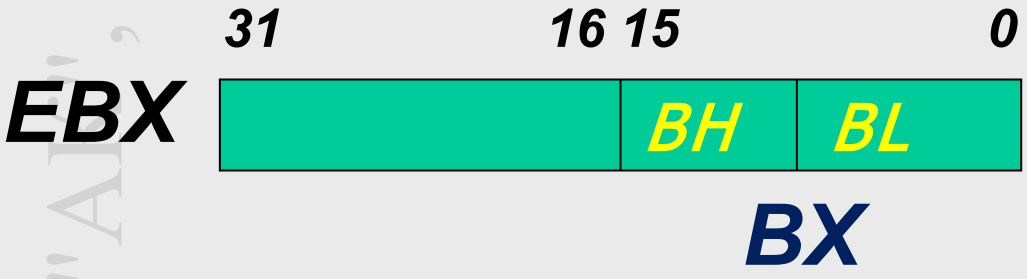
Регистр Аккумулятор (EAX):



A – Accumulator

Участвует в большинстве логич. и арифметич. команд в качестве приемника результата или источника данных

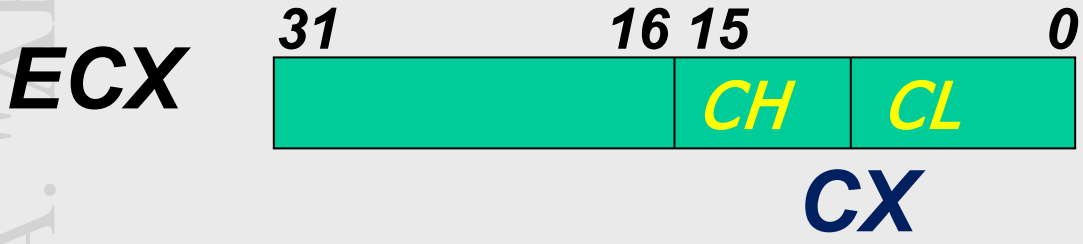
Базовый регистр (EBX):



B - Base

Кроме общего назначения используется при обращении к памяти (хранит базовый адрес памяти)

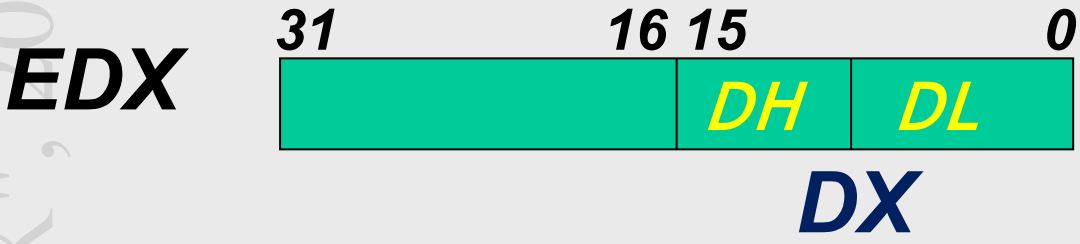
Счетчик:



C- Counter

Кроме общего назначения используется в командах управления циклами, сдвигов и работой с цепочками данных (счетчик)

Регистр данных:



D - Data

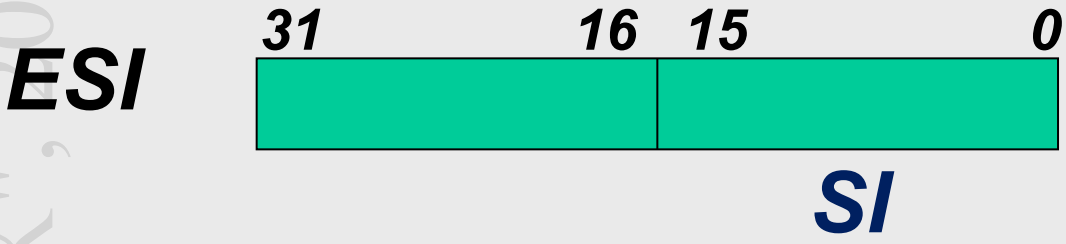
Кроме общего назначения используется в операциях умножения/деления с ФЗ и операциях ввода-вывода (хранит адрес порта УВВ)

Пары регистров

Если значение не помещается в одном регистре, его можно хранить в двух регистрах по частям

- Многие команды работают с такими парами
- Синтаксис: **<регистр 1>:<регистр 2>**
  - в <регистр 1> - старшая часть
  - в <регистр 2> - младшая часть
- Примеры: DX:AX, EDX:EAX

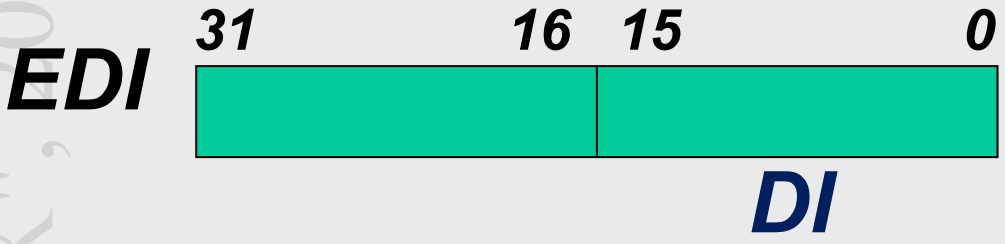
Регистр Индекса источника:



SI – Source Index

Специальный регистр для хранения указателя индекса при работе с цепочками (последовательностями) – адрес источника

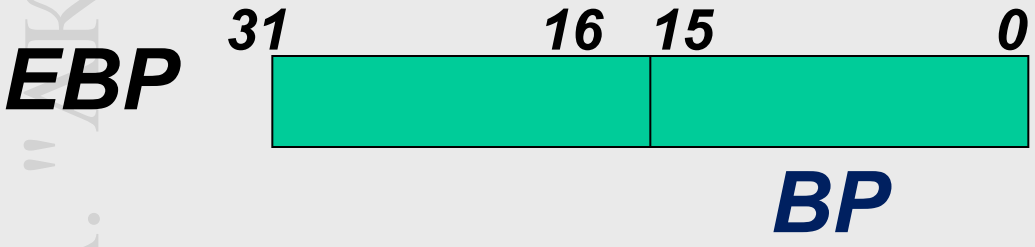
Регистр Индекса приемника:



DI – Destination Index

Специальный регистр для хранения указателя индекса при работе с цепочками (последовательностями) – адрес приемника

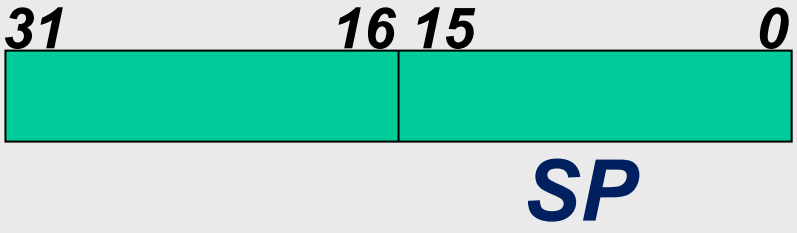
Базовый регистр стека:



BP – Base Pointer

Специальный регистр для хранения базового адреса стека

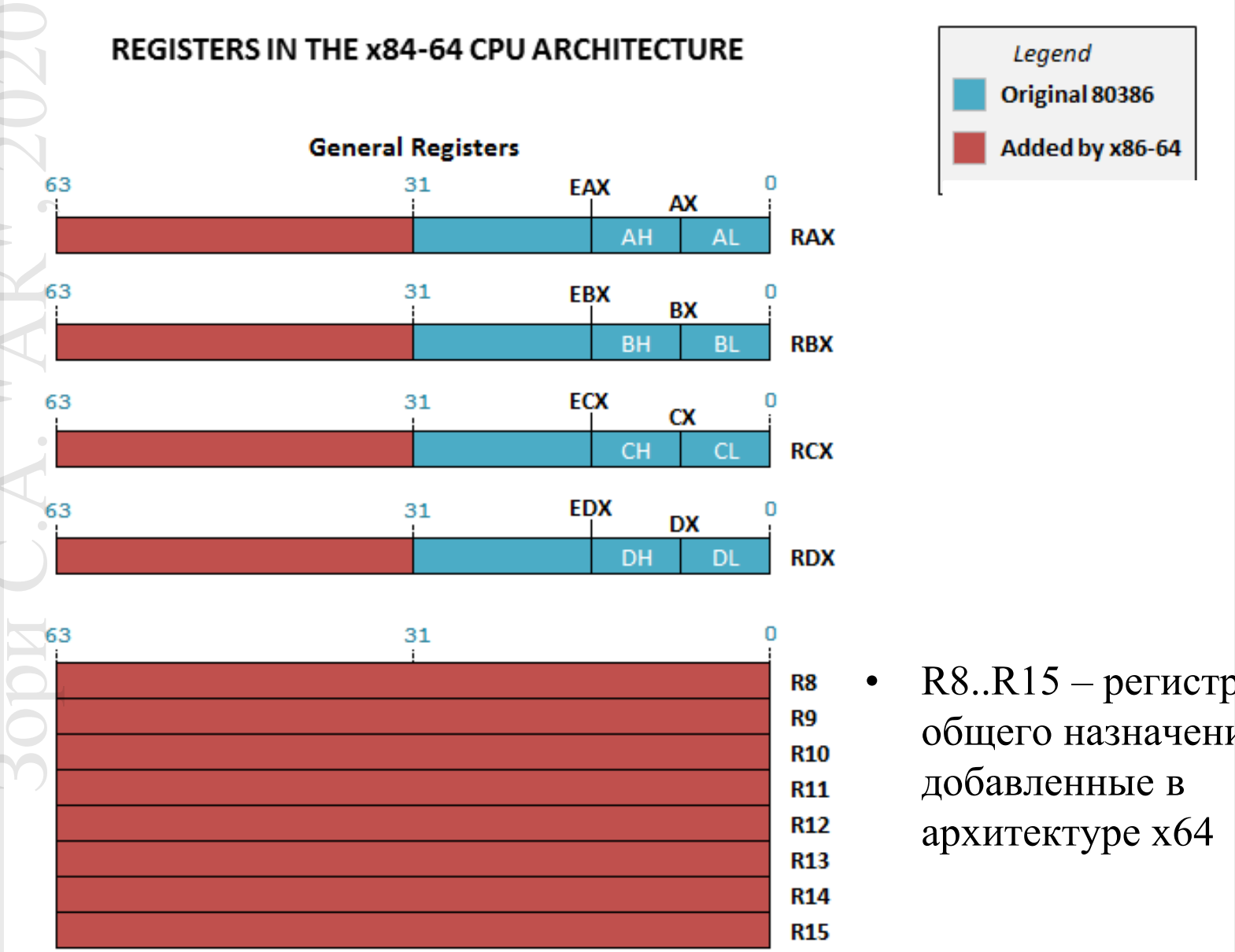
Регистр Указатель стека:



SP – Stack Pointer

Специальный регистр для хранения указателя стека

Расширение Регистров общего назначения в IA-64



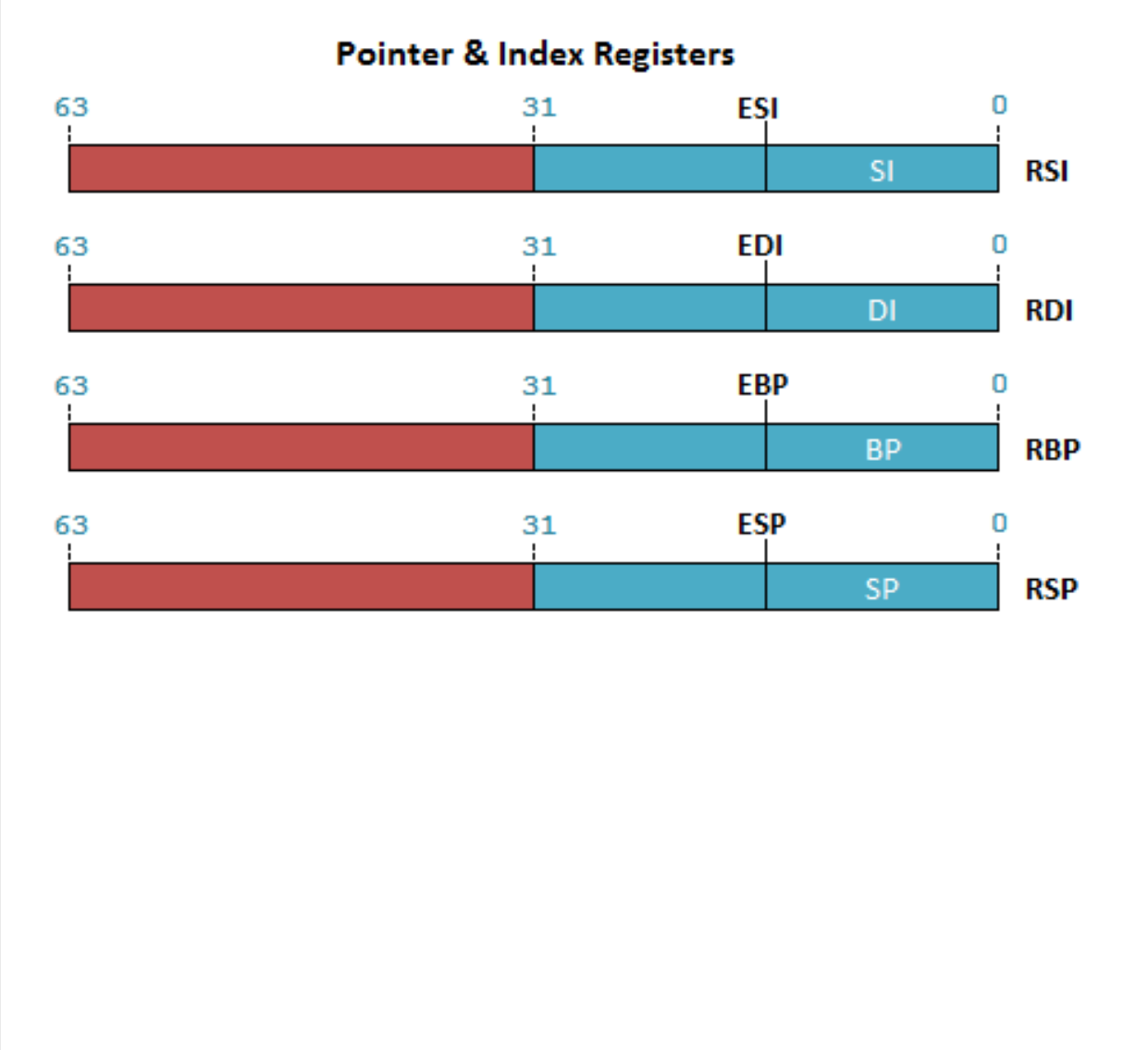
- R8..R15 – регистры общего назначения, добавленные в архитектуре x64

Назначение регистров общего назначения

- **AL/AH/AX/EAX/RAX** – аккумулятор
- **BL/BH/BX/EBX/RBX** – базовый индекс
- **CL/CH/CX/ECX/RCX** – счетчик (counter)
- **DL/DH/DX/EDX/RDX** – данные / двойная точность
- Во многих случаях можно использовать любой из этих регистров, игнорируя его назначения

- **R8..R15** – регистры общего назначения, добавленные в архитектуре x86-64

Указатели и индексные регистры IA-64



Назначение указателей и индексных регистров

- **SI/ESI/RSI** – индекс источника (Source index) для строковых команд
- **DI/EDI/RDI** – индекс приемника (Destination index) для строковых команд
- **SP/ESP/RSP** – указатель на вершину стека (Stack pointer)
- **BP/EBP/RBP** – указатель на начало текущего кадра стека (Stack base pointer)
- **IP/EIP/RIP** – адрес текущей инструкции (Instruction pointer)

Итак:

РОНы хранят:

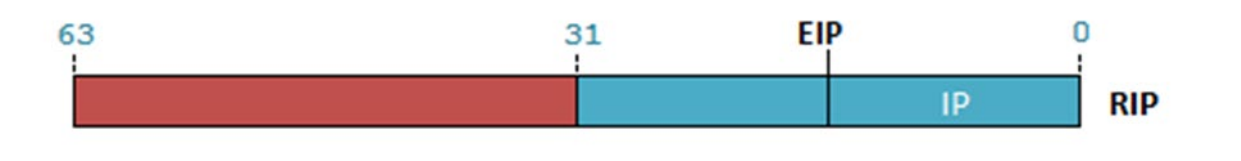
- ***операнды АО и ЛО;***
- ***составные части адреса ОЗУ;***
- ***указатели на ячейки ОЗУ.***

- Во многих случаях можно использовать любой из этих регистров, игнорируя его «предписанное» назначение

2. **Регистры специального назначения** (в ОУ)

Общее назначение – управление процессом обработки.

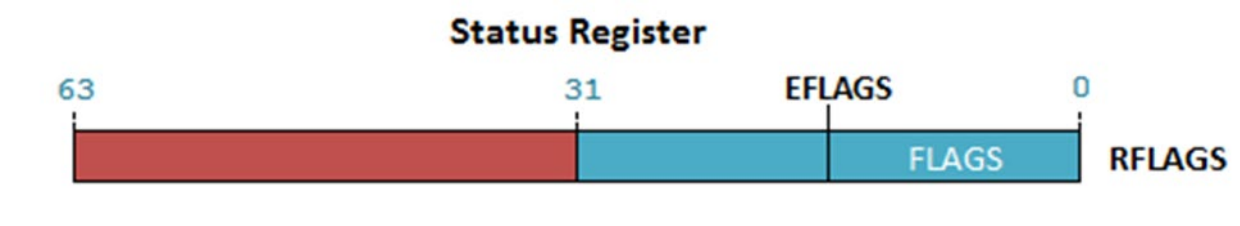
Указатель команд



Регистр **EIP** – ***Instruction Pointer*** – регистр адреса команды, хранит смещение (относительный адрес) в сегменте кода (выполняемой программы) на следующую команду, которая будет выполняться вслед за текущей.

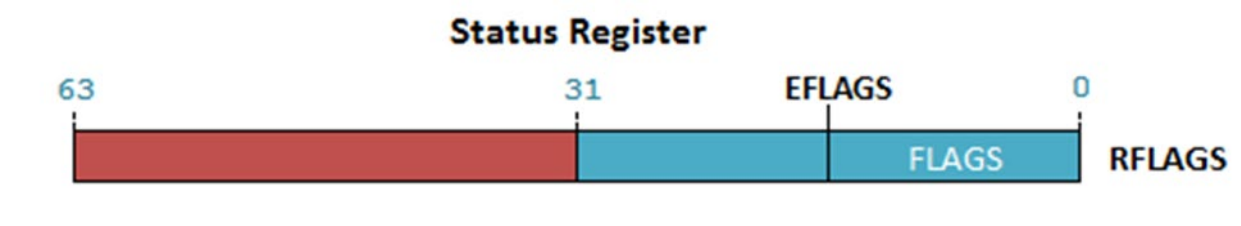
Программисту *непосредственно не доступен*, модифицируется автоматически только с помощью команд передачи управления.

Регистр состояния и управления (флагов)



- Флаги **состояния**
  - устанавливаются в процессе выполнения команд (например, арифметических или сравнения)
  - сигнализируют о ходе выполнения операции или свойствах результата
  - используются в основном для организации условных переходов (изменения последовательного хода программы)

Регистр состояния и управления



- *8 основных флагов состояния в FLAGS:*

№бита

0 CF-флаг переноса из 7/15/31 бита рез-та

2 PF-флаг паритета

4 AF-вспомогательный перенос из 3 бита

6 ZF-если результат=0,этот бит=1

7 SF-знак результата в бите 7/15/31

11 OF-переполнение разряда, если есть перенос в 7/15/31 бит.

Типы флагов +

Управляющий – явно устанавливается программой с целью повлиять на работу ЦП

Системный – то же, но используется ОС

- 1 флаг управления (управляет порядком обработки последовательности байт);
- 8 системных флагов:

  -трассировка;

  -прерывания;

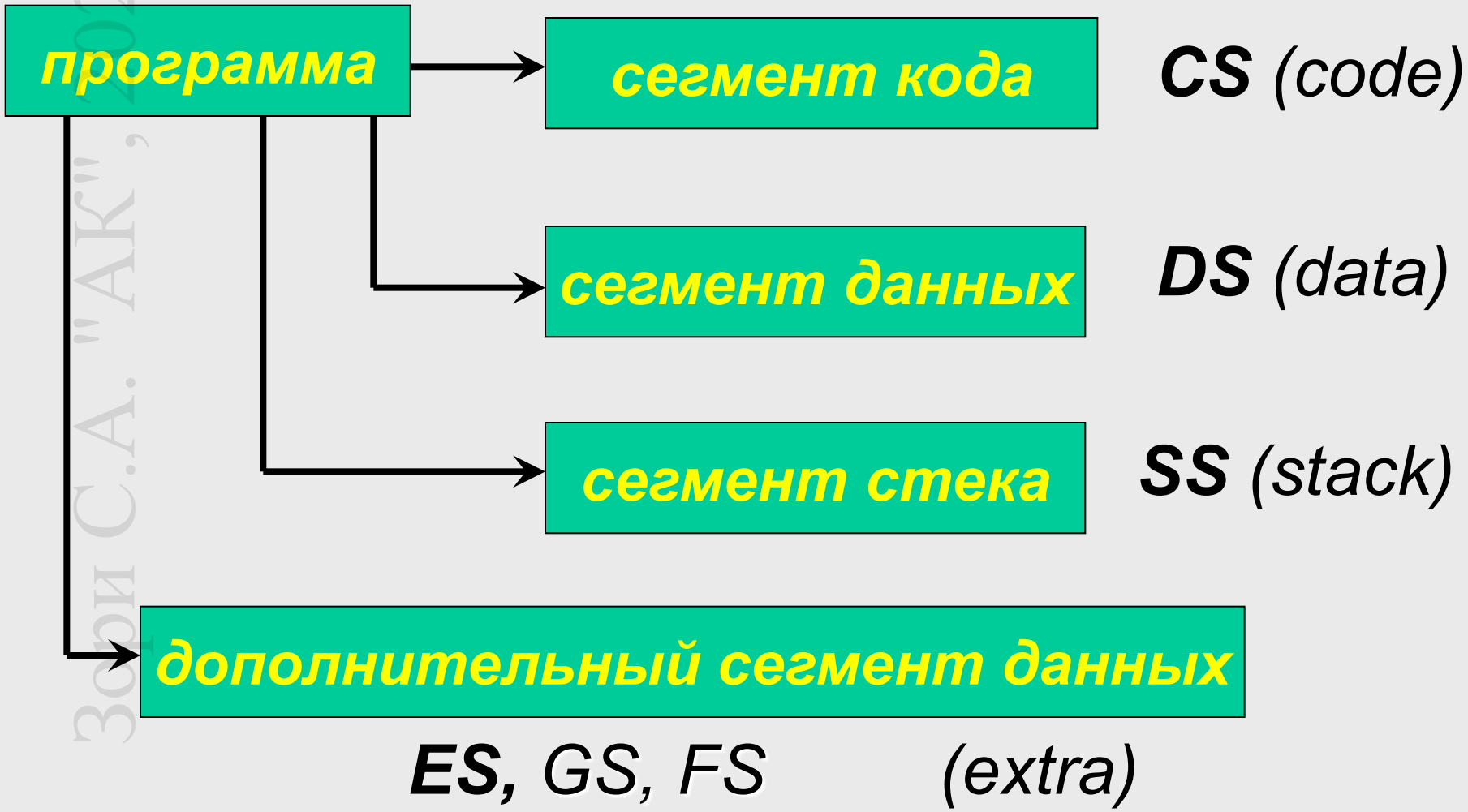
  -признак виртуального режима;

  -контроль выравнивания при обращении к ОЗУ (2/4 байта).

Бит, №	Обозначение	Назначение	Описание	Тип флага
0	CF	Carry Flag	Флаг переноса	Состояние
1	1		Зарезервирован	
2	PF	Parity Flag	Флаг чётности	Состояние
3	0		Зарезервирован	
4	AF	Auxiliary Carry Flag	Вспомогательный флаг переноса	Состояние
5	0		Зарезервирован	
6	ZF	Zero Flag	Флаг нуля	Состояние
7	SF	Sign Flag	Флаг знака	Состояние
8	TF	Trap Flag	Флаг трассировки (пошаговое выполнение)	Системный
9	IF	Interrupt Enable Flag	Флаг разрешения прерываний	Системный
10	DF	Direction Flag	Флаг направления	Управляющий
11	OF	Overflow Flag	Флаг переполнения	Состояние
12 13	IOPL	I/O Privilege Level	Уровень приоритета ввода-вывода	Системный
14	NT	Nested Task	Флаг вложенности задач	Системный
15	0		Зарезервирован	

Бит, №	Обозначение	Назначение	Описание	Тип флага
EFLAGS				
16	RF	Resume Flag	Флаг возобновления	Системный
17	VM	Virtual-8086 Mode	Режим виртуального процессора 8086	Системный
18	AC	Alignment Check	Проверка выравнивания	Системный
19	VIF	Virtual Interrupt Flag	Виртуальный флаг разрешения прерывания	Системный
20	VIP	Virtual Interrupt Pending	Ожидающее виртуальное прерывание	Системный
21	ID	ID Flag	Проверка на доступность инструкции `CPUID`	Системный
22 … 31	0		Зарезервированы	
RFLAGS				
32 … 63	0		Зарезервированы	

3. СЕГМЕНТНЫЕ РЕГИСТРЫ



Основное назначение – адресация данных и команд в памяти

Сегментные регистры предназначены для хранения указателей на участки оперативной памяти, называемые ***сегментами***.

В реальном режиме в них хранятся *начальные адреса* сегментов, в защищенном – *селекторы* сегментов – указатели на специальную таблицу, в которой хранится информация о размещении и других свойствах сегментов.

Сегментные регистры

Используются в адресации, размер – 16 бит

- **CS** – сегмент кода (code)
- **DS** – сегмент данных (data)
- **SS** – сегмент стека (stack)
- **ES** (extra)
- **FS**
- **GS**

} дополнительные, произвольного назначения

4. Конвейер выполнения команд (в ШИ)

– каждая команда выполняется в несколько этапов. Для реализации каждого этапа имеются специальные устройства. Команда последовательно проходит через эти устройства, за ней следует следующая команда, таким образом в итоге образуется конвейер команд.

Состоит:

Буфер команд – для хранения команд в процессе выполнения.

Кэш данных – для временного хранения считанных из ОЗУ данных (блока).

Кэш команд – для промежуточного хранения считанных из ОЗУ команд (блока).

Программисту непосредственно не доступны.

Таким образом организована

Многоуровневая (иерархическая) память:

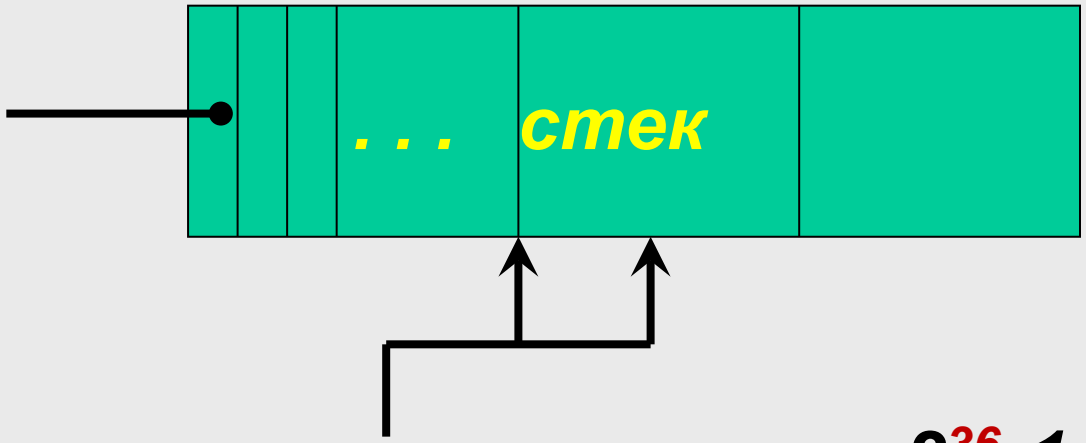
- **регистры** микропроцессора;
- кэш команд и кэш данных (**L1**);
- кэш второго уровня (**L2**);
- кэш третьего уровня (**L3**);
- оперативная **память**;
- устройство **ввода-вывода**.

Объем памяти сверху вниз увеличивается, но быстродействие существенно уменьшается!

Также существуют **отдельные** регистры (в соответствующих вычислительных устройствах (**PU**) МП) для:
- чисел с плавающей точкой (в **FPU**);
- векторов чисел (в **SIMD PU** – для MMX/SSE/AVX- расширений системы команд)

Сегментированная модель ОЗУ

АДРЕСНОЕ ПРОСТРАНСТВО ОЗУ



$2^{20}-1$

i8086 (1МБ)

$2^{32}-1$

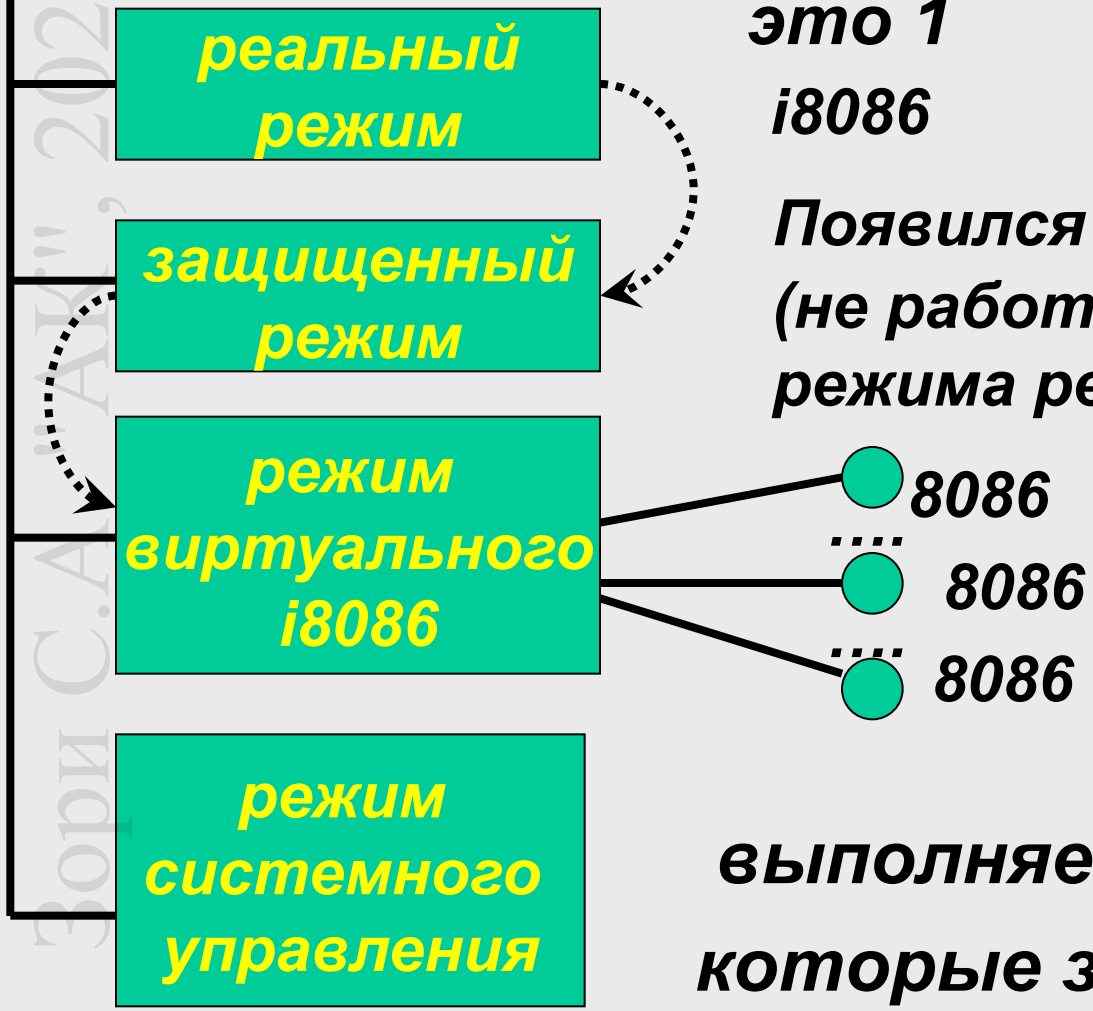
i486 (4ГБ)

$2^{36}-1$

с *Pentium Pro*

64 Гбайт

Режимы работы МП (влияют на организацию использования памяти)



это 1

i8086

Появился в i80286

(не работают программы для режима реального времени)

8086

....

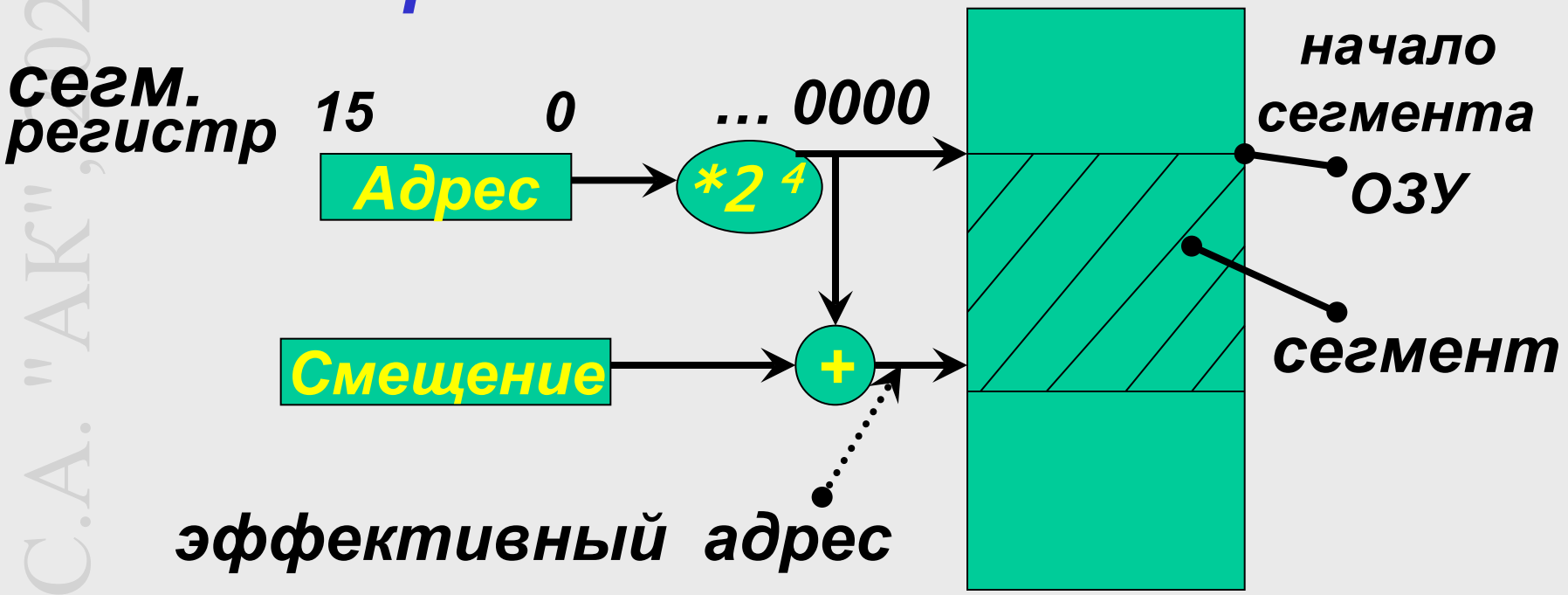
8086

....

8086

выполняет операции, которые зависят от состояния ПК

Реальный режим



Применение сегментации помогает сократить длину машинной команды => сократить объем программы.

Защищенный режим ***нужен:***

- для ограничения доступа прикладных программ к памяти, занимаемой ОС;
- для изоляции в памяти одновременно выполняемых команд.

В защищенном режиме каждому сегменту памяти соответствует ***набор свойств***: можно ли читать, выполнять, изменять сегмент, и кому доступны эти операции.

Эта информация хранится в ***дескрипторе сегмента***, а в команде, которая обращается к памяти, хранится селектор для дескриптора

Защищенный режим