

Обзор архитектуры IA32/EM64T

Юрий Долгов, Дмитрий Шкурко

Архитектура

Микроархитектура

- Набор внутренних устройств
- Взаимодействие устройств

Интерфейс для работы с процессором

- Набор используемых команд
- Набор правил исполнения команд
- Форматы и типы данных

Ассоциативная память

В каждой ячейке хранится ключ и данные

- Выборка данных осуществляется по заданному ключу
- Каждая ячейка может хранить любую пару ключ-значение

Адрес	Данные
Адрес	Данные
Адрес	Данные

С точки зрения реализации:

- ассоциативная память дорогая
- с увеличением объема уменьшается скорость доступа

Устройство кэша (частично ассоциативная память)

Линия (блок)

Данные

Теги линий

Yellow			
Green	Green	Green	Green

Ассоциативное
множество
(4-way)

Адрес

Значение из
тега

Номер линии

Смещение внутри
линии

Optimization of applications for Intel* platforms

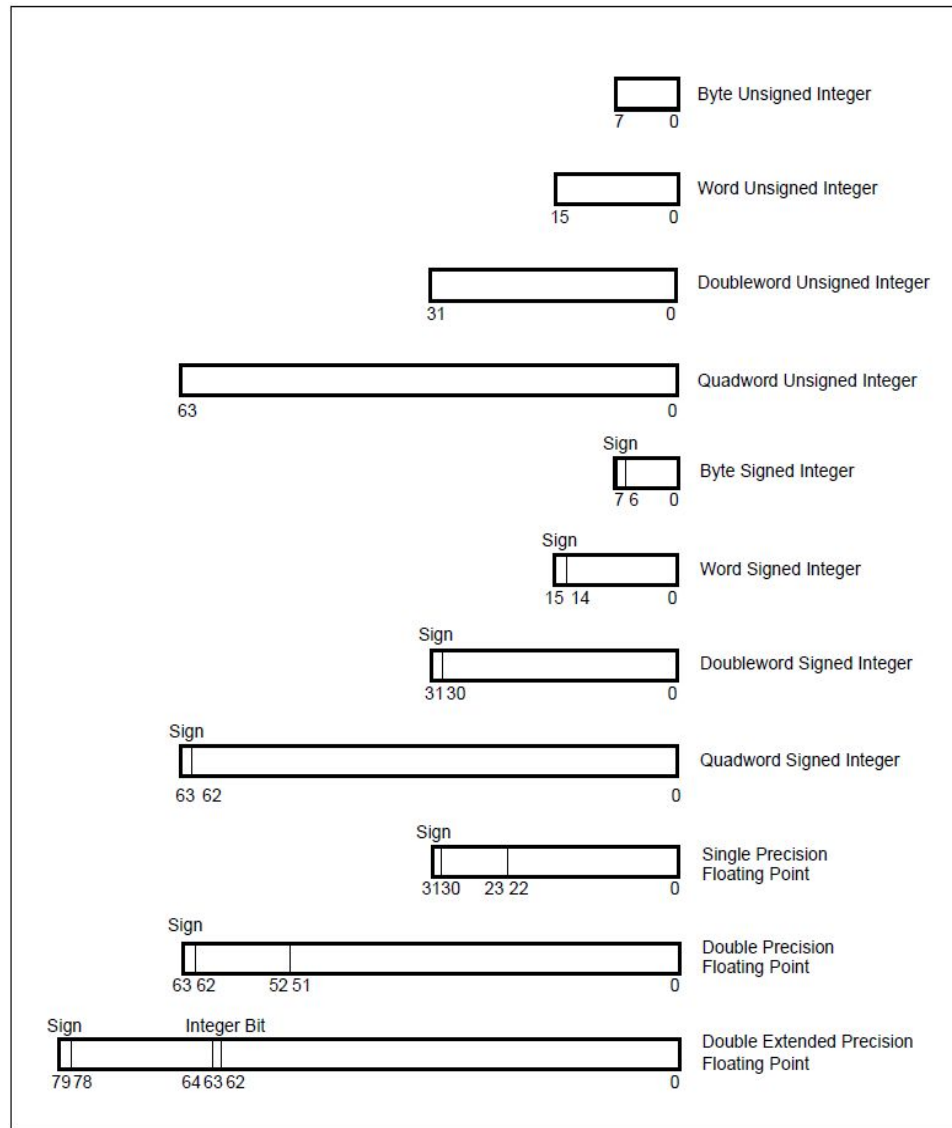
MESI (Modified Exclusive Shared Invalid) протокол когерентности WB памяти

Начальное состояние линии кеша	Состояние после чтения	Состояние после записи
M	M	M
E	E	M
S	S	M (RFO)
I	S	M (RFO, RI)

Процессор содержит несколько специальных буферов для уменьшения нагрузки на шину, возникающей за счет запросов RFO

Типы данных

- Целые (8, 16, 32, 64 бита)
 - без знака
 - со знаком
- Числа с плавающей точкой
 - Одинарной точности (32 бита)
 - Двойной точности (64 бита)
 - Расширенной точности (80 бит)
- Упакованные типы
 - Несколько базовых типов, упакованных в 128 или 64 бита
- Указатели (64 бита или 32 бита)



Optimization of applications for Intel* platforms



Целочисленные регистры (32-битный режим)

- Регистры общего назначения(**General-purpose registers**):
 - 8 32-битных регистров используются 32-битном режиме для обращения к операндам в памяти (EAX, EBX, ECX, EDX, EBP, ESI EDI, и ESP).
- Сегментные регистры (**Segment registers**):
 - 6 16-битных сегментных регистров содержат части указателей, служат для обращения к памяти (CS,DS, SS, ES, FS, и GS)
- EFLAG регистр (**EFLAG register**):
 - Этот 32-битный регистр служит для предоставления статуса и контроля над выполнением базовых арифметических операций, сравнения и системных операций.
- EIP регистр (**EIP register**):
 - Этот 32-битный регистр содержит указатель на текущую инструкцию.

Целочисленные регистры (64-битный режим)

- Регистры общего назначения(**General-purpose registers**):
 - 16 64-битных регистров используются 32-битном режиме для обращения к в памяти в памяти (RAX, RBX, RCX, RDX, RBP, RSI, RDI, RSP и R8-R15 – используются с префиксом REX).
- Сегментные регистры (**Segment registers**):
 - устанавливаются уникальным образом
- RFLAG регистр (**RFLAG register**):
 - Этот 64-битный регистр служит для предоставления статуса и контроля над выполнением базовых арифметических операций, сравнения и системных операций.
- RIP регистр (**RIP register**):
 - Этот 64-битный регистр содержит указатель на текущую инструкцию.

Двоичная арифметика

Арифметика целых чисел без знака по модулю 16

•Вычитание

- $7 - 3 = 0111 - 0011 = 0111 + 1101 = 10100 = 4$ (есть перенос (CF=0), результат > 0)
- $3 - 7 = 0011 - 0111 = 0011 + 1001 = 1100 = 12$ (нет переноса (CF=1), результат < 0)
- Чтобы сравнить числа нужно вычесть одно из другого и проверить наличие переноса

•Сложение

- $7 + 3 = 0111 + 0011 = 1010 = 10$ (нет переноса (CF=0), нормальная ситуация)
- $7 + 10 = 0111 + 1010 = 10001 = 1$ (есть перенос (CF=1), ненормальная ситуация)

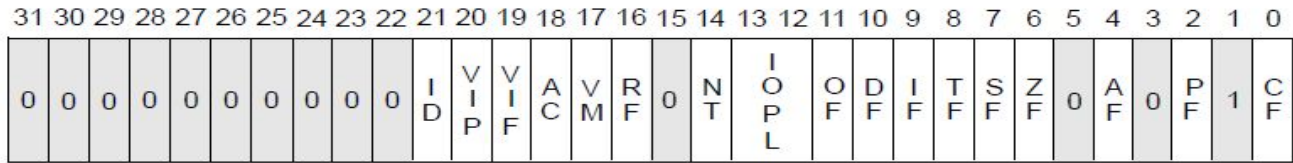
Арифметика целых чисел со знаком по модулю 16

•Сложение

- $(-7) + 3 = 1001 + 0011 = 1100$ (переносы равны (OF=0), знак равен 1, результат < 0)
- $7 + (-3) = 0111 + 1101 = 10100$ (переносы равны (OF=0), знак равен 0, результат > 0)
- $2 + 1 = 0010 + 0001 = 0011$ (переносы равны (OF=0), результат > 0)
- $(-2) + (-1) = 1110 + 1111 = 11101$ (переносы равны (OF=0), результат < 0)

Переполнения

- $7 + 3 = 0111 + 0011 = 1010$ (переносы разные (OF=1), результат > 0)
- $-7 + (-3) = 1001 + 1101 = 10110$ (переносы разные (OF=1), результат < 0)

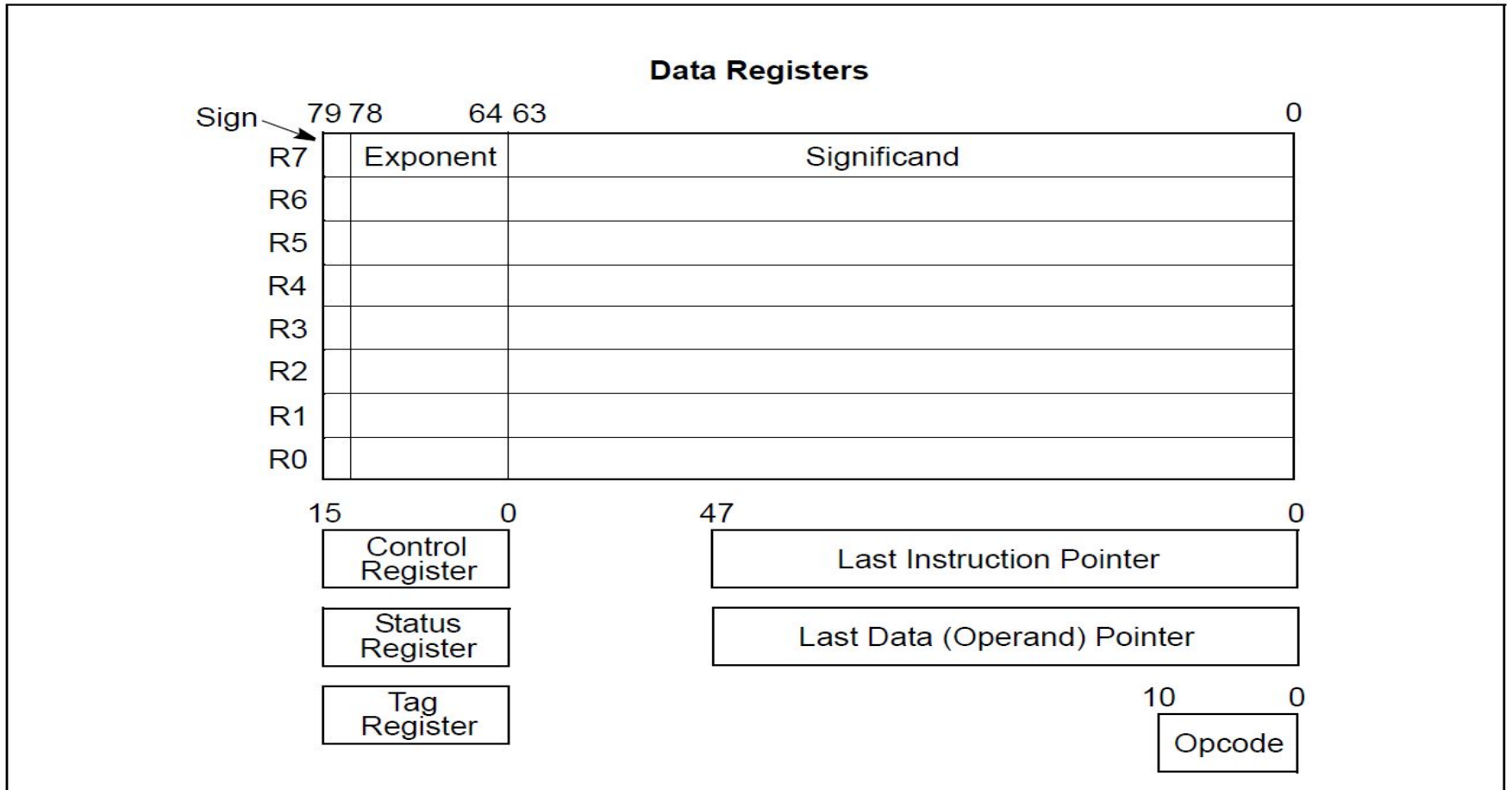


S Indicates a Status Flag
 C Indicates a Control Flag
 X Indicates a System Flag

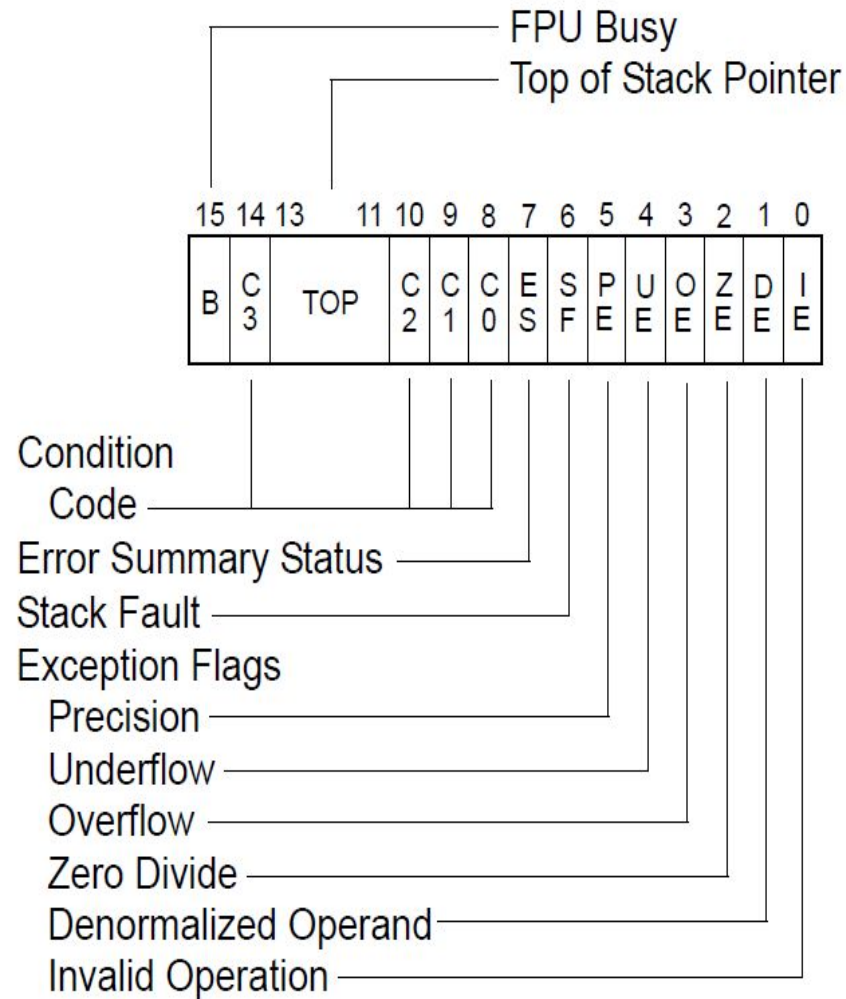
Reserved bit positions. DO NOT USE.
 Always set to values previously read.



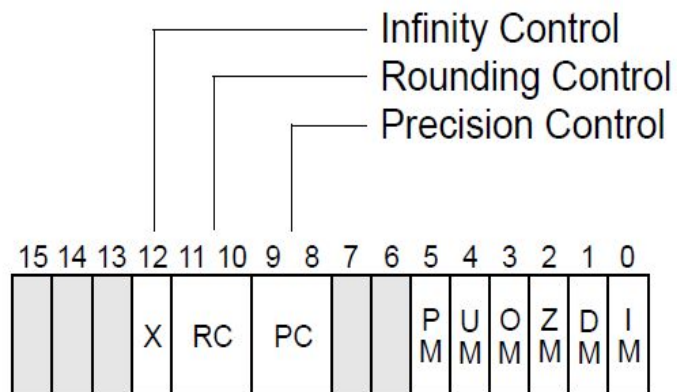
x87 регистры



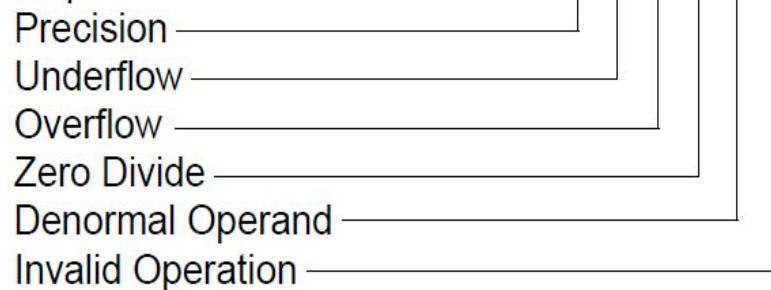
Слово статуса x87



Контрольное слово x87



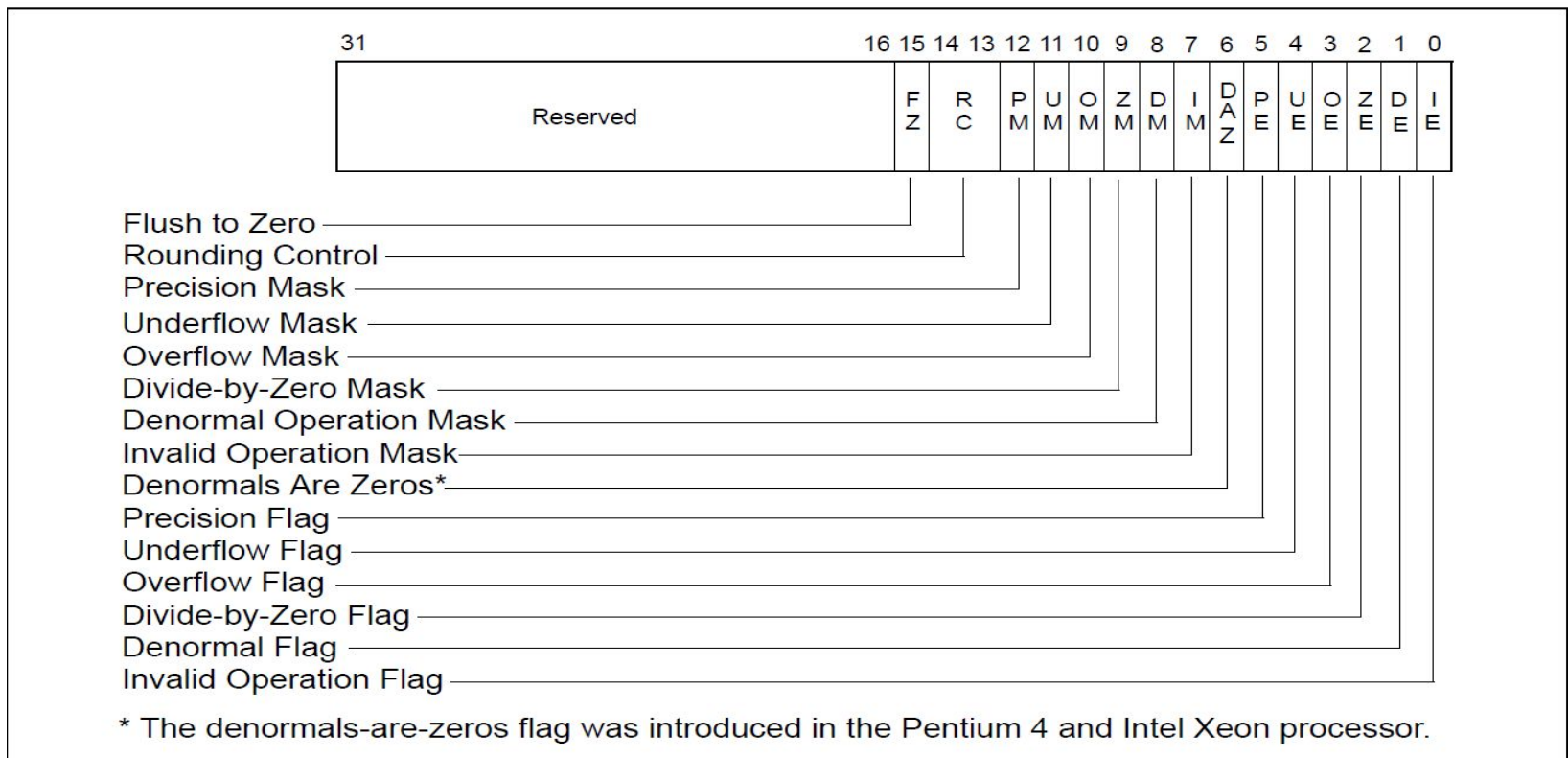
Exception Masks



Reserved

SSE

- 8 или 16 128-битных регистров (xmm00-xmm07)
- Регистр статуса и контрольной информации (mxcsr)



Формат инструкции

<Мнемоника> <входной и выходной операнд> <входной операнд>

- Операнд может быть следующих типов
 - Регистр
 - Память
 - База + индекс * множитель + смещение
 - Смещение относительно указателя на команду
 - Константа
- Некоторые инструкции имеют неявные операнды
<jump/call> <адрес>
- Адрес может быть взят
 - Из регистра
 - Непосредственно из команды

Примеры инструкций

- Add rax, [rbx + 2*rdx + 8] (rax <- rax + [rbx + 2*rdx + 8]) (sub)
- Mul rcx (rdx:rax <- rax*rcx) (imul, div, idiv)
- Cmp rcx, rdi (изменяет флаги)
- Mov rcx, rbx + 2*rdx + 8] (чтение в регистр)
- Push rcx (pop rcx)
- Lea rax, [rbx + 2*rdx + 8] (rax <- rbx + 2*rdx + 8)
- Jmp [rax]
- Call <label>
- Ret
- Fsub st1, st0 (fadd)

SSE1, 2, 3

- Paddusw xmm1, xmm2 (paddsq, paddb)
- Addps xmm1, xmm2 (subpd, mulps, divps)
- Cvttpd2pi xmm1, xmm2 (cvttpd2dq)
- Movhlps xmm1, xmm2 (movhpd, movlps)

Q&A

Optimization of applications for Intel* platforms



Thank you.

Optimization of applications for Intel* platforms

