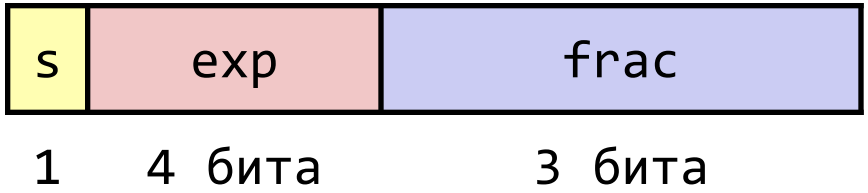
Лекция 16

4 апреля

Пример



- 8-разрядные числа с плавающей точкой
  - Знаковый бит – старший бит
  - Следующие четыре бита – порядок, смещение – 7
  - Последние три бита – дробная часть (мантисса)

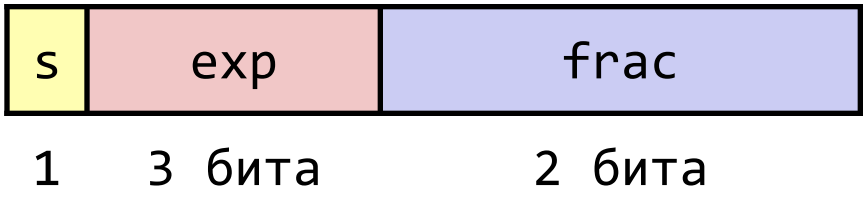
- Выполнены все требования стандарта IEEE 754 к формату числа
  - Реализованы нормализованные и денормализованные числа
  - Представлены значения 0, NaN, бесконечность

Диапазоны значений (только для положительных чисел)

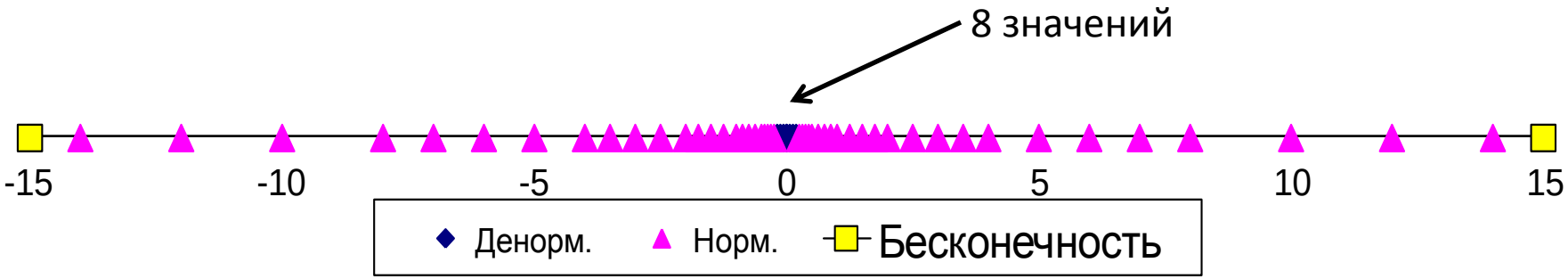
	s	exp	frac	E	Значения	
	0	0000	000	-6	0	
	0	0000	001	-6	1/8*1/64 = 1/512	Ближайшее к 0
	0	0000	010	-6	2/8*1/64 = 2/512	
Денормализованные числа	…					
	0	0000	110	-6	6/8*1/64 = 6/512	
	0	0000	111	-6	7/8*1/64 = 7/512	Наибольшее денорм.
	0	0001	000	-6	8/8*1/64 = 8/512	Наименьшее норм.
	0	0001	001	-6	9/8*1/64 = 9/512	
	…					
	0	0110	110	-1	14/8*1/2 = 14/16	
	0	0110	111	-1	15/8*1/2 = 15/16	Ближайшее к 1 «снизу»
	0	0111	000	0	8/8*1 = 1	
	0	0111	001	0	9/8*1 = 9/8	Ближайшее к 1 «сверху»
	0	0111	010	0	10/8*1 = 10/8	
Нормализованные числа	…					
	0	1110	110	7	14/8*128 = 224	
	0	1110	111	7	15/8*128 = 240	Наибольшее норм.
	0	1111	000	n/a	inf	

Распределение значений по числовой прямой

- 6-разрядный формат
  - e = 3 бита порядка
  - f = 2 бита мантиссы
  - Смещение $2^{3-1}-1 = 3$

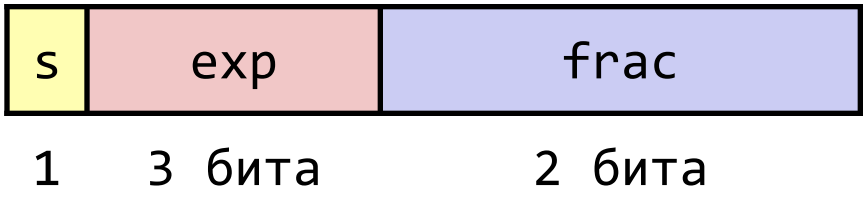


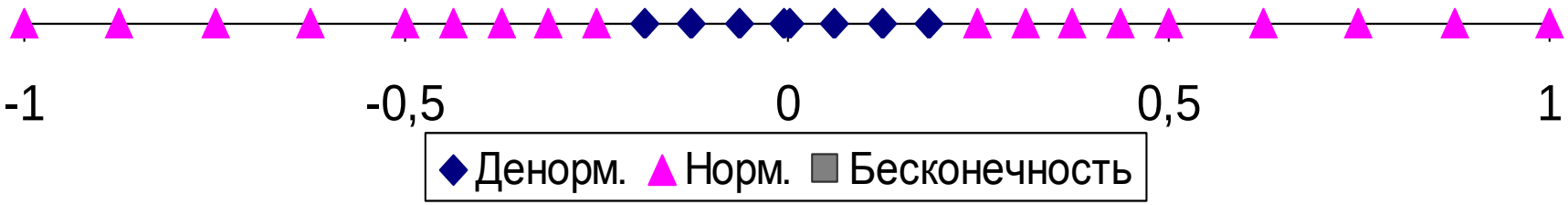
- Распределение сильно «сгущается» в окрестности 0



Распределение значений по числовой прямой (вид вблизи)

- 6-разрядный формат
  - e = 3 бита порядка
  - f = 2 бита мантиссы
  - Смещение 3





Некоторые числа

Описание	*exp*	*frac*	*Численное значение*
• Ноль	00…00	00…00	0.0
• Наименьшее «+» денорм.	00…00	00…01	$2^{-\{23,52\}} \times 2^{-\{126,1022\}}$
– Одинарная точность ≈ 1.4×10^{-45}			
– Двойная точность ≈ 4.9×10^{-324}			
• Наибольшее денорм.	00…00	11…11	$(1.0 - \varepsilon) \times 2^{-\{126,1022\}}$
– Одинарная точность ≈ 1.18×10^{-38}			
– Двойная точность ≈ 2.2×10^{-308}			
• Наименьшее «+» норм.	00…01	00…00	$1.0 \times 2^{-\{126,1022\}}$
– Немногим больше чем наибольшее денормализованное			
• Единица	01…11	00…00	1.0
• Наибольшее норм.	11…10	11…11	$(2.0 - \varepsilon) \times 2^{\{127,1023\}}$
– Одинарная точность ≈ 3.4×10^{38}			
– Двойная точность ≈ 1.8×10^{308}			

Точность
{одинарная,двойная}

Особенности кодировки

- FP ноль совпадает с целочисленным нулем
  - Все биты = 0
- Допустимо (в большинстве случаев) использовать беззнаковое целочисленное сравнение
  - Сперва сравниваем знаковые биты
  - Необходимо рассматривать −0 = 0
  - NaNs
    - В целочисленной интерпретации больше, чем любые другие числа
    - Что необходимо выдавать в качестве результата сравнения?
  - В противном случае …
    - Денормализованные vs. Нормализованные
    - Нормализованные vs. Бесконечность

Операции над числами с плавающей точкой

- $x +_f y = \mathrm{Round}(x + y)$
- $x \times_f y = \mathrm{Round}(x \times y)$
- Основная идея
  - Сперва вычислить точный результат
  - Поместить результат в требуемый размер точности
    - Переполнение, если порядок слишком большой
    - Возможно придется округлять поле frac

Округление

- Способы округления

	1.40	1.60	1.50	2.50	–1.50
– К нулю	1	1	1	2	–1
– К наименьшему (−∞)	1	1	1	2	–2
– К наибольшему (+∞)	2	2	2	3	–1
– К ближайшему (✓)	1	2	2	2	–2

Округление к ближайшему целому числу

- Основной способ округления
  - Все остальные способы дают статистическое смещение
    - Пример: суммирование положительных чисел будет давать устойчивую недо- или пере- оценку результата

- Применимо при округлении в произвольной позиции дроби
  - Когда число расположено точно посредине двух значений к которым можно округлить
    - Округляют к тому числу, у которого наименьшая значащая цифра четная
  - Например, округление до ближайших сотых

1.2349999	1.23	
1.2350001	1.24	
1.2350000	1.24	(середина — округляем к большему)
1.2450000	1.24	(середина — округляем к меньшему)

Округление двоичных чисел

- Двоичные дробные числа
  - “Четные” числа у которых младший значащий бит 0
  - “Середина” – когда биты справа от позиции к которой происходит округление = $100\ldots_2$

- Примеры
  - Округление до ближайшей 1/4 (2 бита справа от бинарной точки)

Число	Двоичное	Окр.	Действие	Окр. число
2 3/32	10.00011_2	10.00_2	(<1/2—down)	2
2 3/16	10.00110_2	10.01_2	(>1/2—up)	2 1/4
2 7/8	10.11100_2	11.00_2	(1/2—up)	3
2 5/8	10.10100_2	10.10_2	(1/2—down)	2 1/2

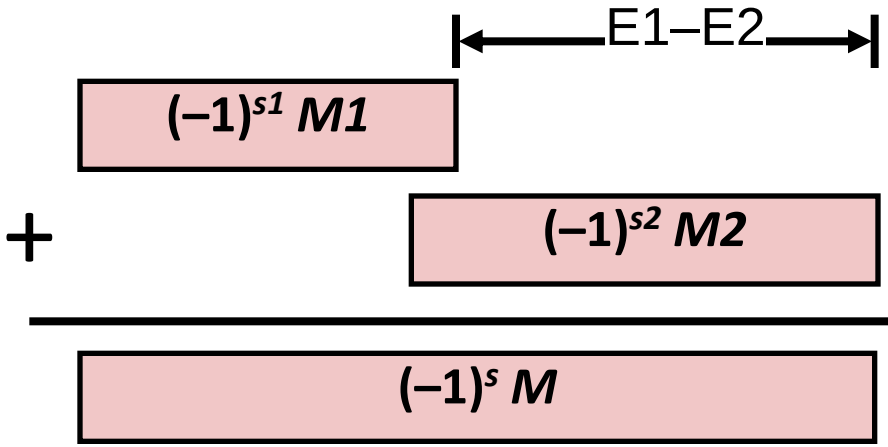
Умножение

- $(-1)^{s1}\ M1\ \ 2^{E1}\ \ \ x\ \ \ (-1)^{s2}\ M2\ \ 2^{E2}$
- Точный результат: $(-1)^{s}\ M\ \ 2^{E}$
  - Знаковый бит s: s1 ^ s2
  - Мантисса M: M1 x M2
  - Порядок E: E1 + E2

- Исправление
  - Если $M \geq 2$, сдвигаем M вправо (делим на 2), увеличивая E
  - Если E выходит за пределы, переполнение
  - Округляем M до соответствующего размера поля frac

Сложение

- $(-1)^{s1}\ M1\ \ 2^{E1}\ \ +\ \ (-1)^{s2}\ M2\ \ 2^{E2}$
  - Пусть E1 > E2
- Точный результат: $(-1)^{s}\ M\ \ 2^{E}$
  - Знаковый бит s, мантисса M:
    - Результат выравнивания и сложения
  - Порядок E: E1



- Исправление
  - Если M ≥ 2, сдвигаем M вправо, увеличивая E
  - Если M < 1, сдвигаем M влево на k позиций, уменьшая E на k
  - Переполнение если E выходит за пределы
  - Округляем *M* до соответствующего размера поля frac

Математические свойства сложения

- Выполняются ли свойства Абелевых групп
  - Замкнутость?
    - Результатом может быть бесконечность или NaN
  - Коммутативность?
  - Ассоциативность?
    - Переполнения и изменение результата при округлении
  - `0.0` – нейтральный элемент?
  - Каждый элемент имеет обратный
    - За исключением бесконечности и NaN
- Монотонность
  - $a \geq b \Rightarrow a+c \geq b+c$?
    - За исключением бесконечности и NaN

Математические свойства умножения

- Выполняются ли свойства коммутативных колец
  - Замкнуто ли относительно умножения?
    - Результат может быть бесконечность или NaN
  - Умножение коммутативно?
  - Умножение ассоциативно?
    - Возможность переполнения, неточности округления
  - 1.0 – мультипликативная единица?
  - Умножение дистрибутивно над сложением?
    - Возможность переполнения, неточности округления

- Монотонность
  - $a \geq b \ \& \ c \geq 0 \Rightarrow a * c \geq b * c$?
    - Исключение – бесконечность и NaN

Числа с плавающей точкой в языке Си

- Язык Си вводит два уровня точности
  - –float одинарная точность
  - –double двойная точность
- Приведение типа
  - –Приведение типа между int, float, и double включает изменение битового представления
  - – double/float → int
    - Отбрасывается дробная часть (аналогично округлению к нулю)
    - Поведение не определено, когда значение вне допустимого диапазона или NaN: как правило устанавливается TMin
  - – int → double
    - Точное приведение, поскольку long и int 32 бита ≤ 53 бита
  - – int → float
    - Будет округляться согласно принятым соглашениям

Задачи

- Для каждого Си-выражения объяснить:
  - почему оно верно для любого значения переменных, …
  - … либо почему ложно

```
int x = …;
float f = …;
double d = …;
```

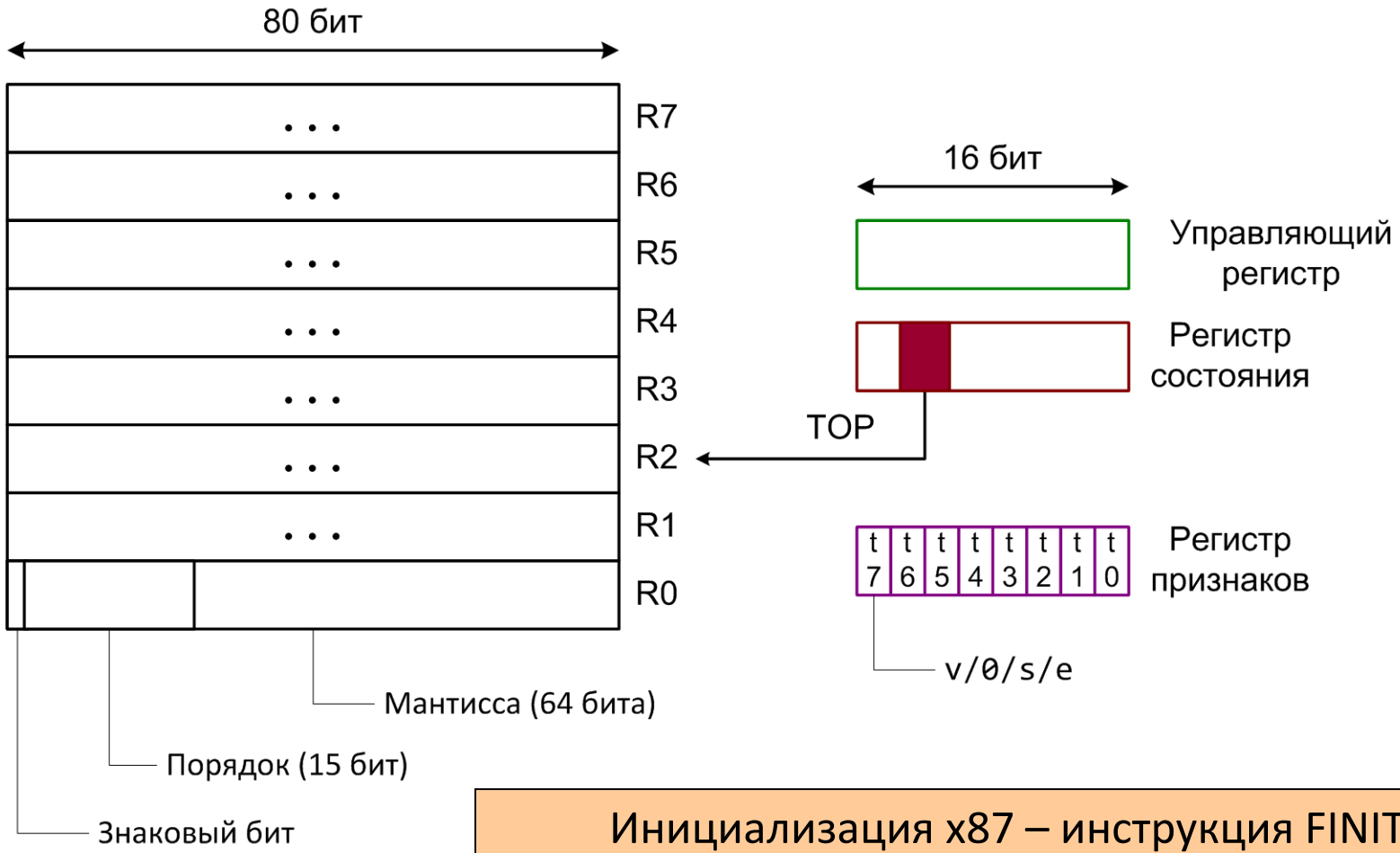
Предполагается, что d и f не являются NaN

- `x == (int)(double) x`
- `x == (int)(float) x`
- `f == (float)(double) f`
- `d == (float) d`
- `f == -(-f);`
- `2/3 == 2/3.0`
- `d < 0.0` ⇒ `((d*2) < 0.0)`
- `d > f` ⇒ `-f > -d`
- `d * d >= 0.0`
- `(d+f)-d == f`

Промежуточные итоги

- IEEE754 – четкое определение математических свойств
- Представляются числа вида $M \times 2^E$
- Семантика операций не зависит от особенностей аппаратуры
  - Сперва точное вычисление, затем округление
- Отличия от «настоящей» арифметики
  - Нарушаются свойства ассоциативности и дистрибутивности
  - Создаются сложности для компилятора и серьезных математических вычислений

Упрощенная схема x87

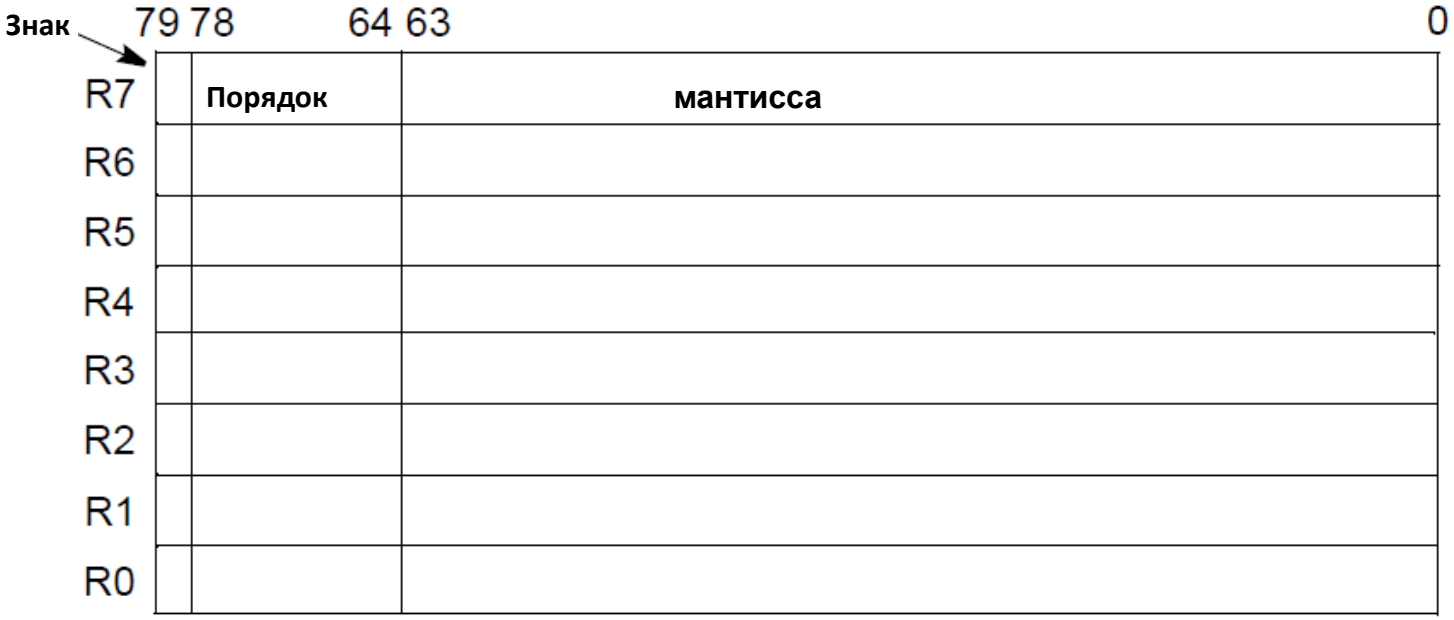


Инициализация x87 – инструкция FINIT

CW = 0x037F SW = 0x0000 Tag = 0xFFFF

Размер чисел с плавающей точкой

Регистры данных

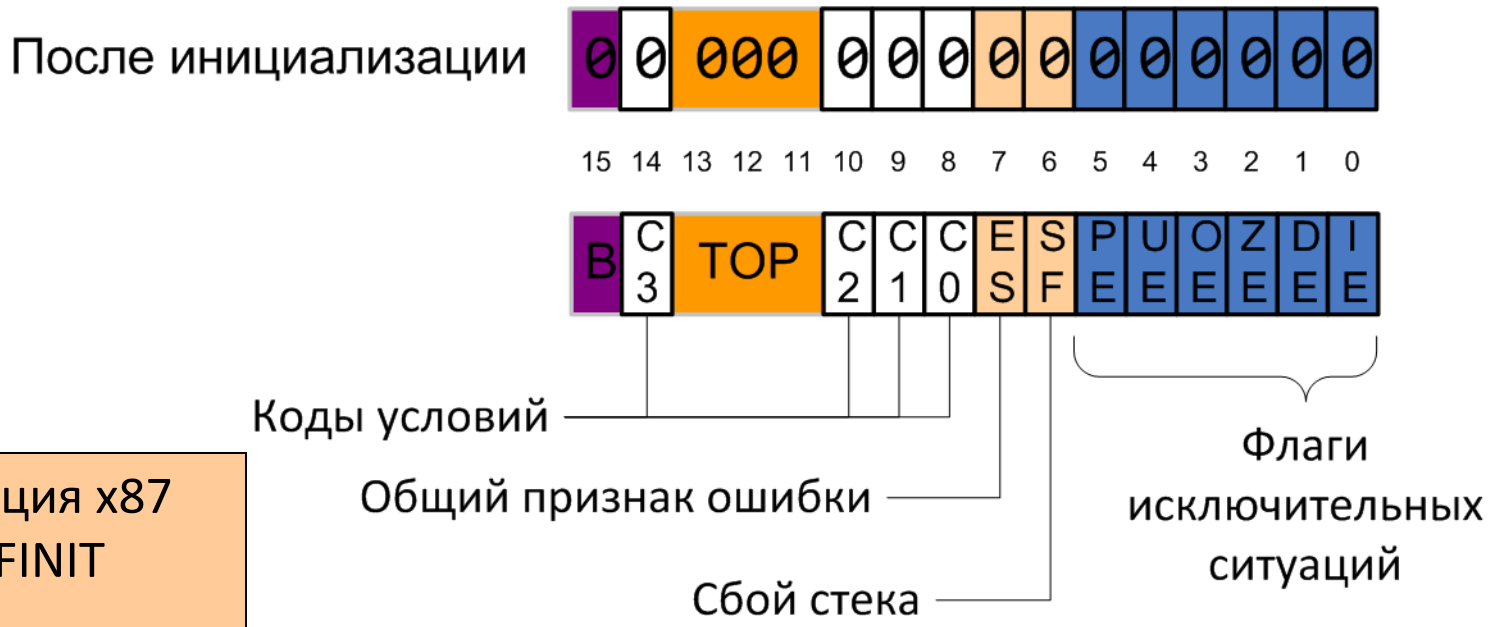


Обмен данными только с памятью.

```
dd 1.234567e20 ; Константы одинарной точности
dq 1.234567e20 ; Двойной точности
dt 1.234567e20 ; Расширенной точности
```

Слово (регистр) состояния

- SF – переполнение стека (C1 показывает направление)
- Исключительные ситуации: точность, переполнение, деление на ноль, денормализованный операнд, «неправильные» данные

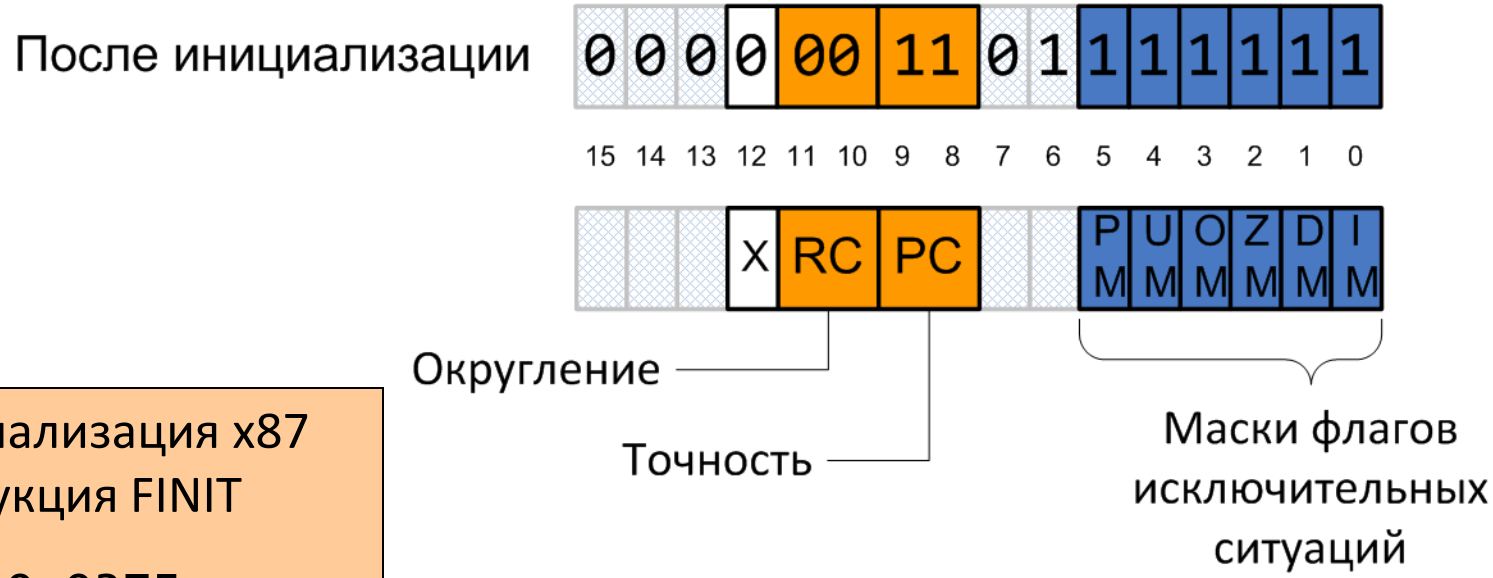


Инициализация x87 инструкция FINIT

SW = 0x0000

Управляющий регистр

- Точность: одинарная, двойная, расширенная
- Округление: к ближайшему четному, к нулю, к +/- бесконечности
- Флаг X – совместимость с 287
- Маски соответствуют исключениям в слове состояния



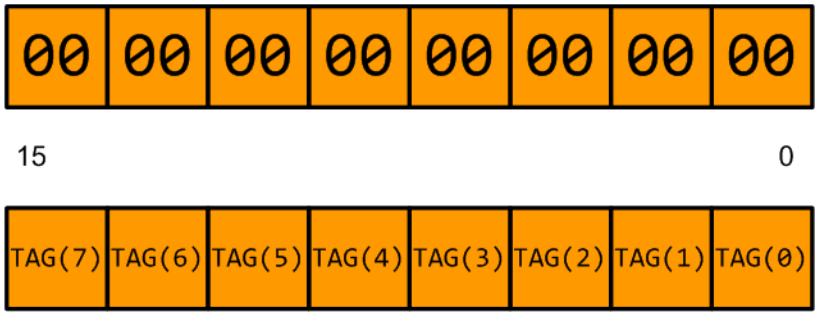
Инициализация x87 инструкция FINIT

CW = 0x037F

Регистр признаков (тагов)

- Состояние регистров
  - 0 – нормализованное число с плавающей точкой
  - 1 – число ноль
  - 2 – особые числа (NaN, ± ∞, денормализованное число)
  - 3 – регистр свободен
- Нумерация соответствует физическим регистрам

После инициализации



Инициализация x87 инструкция FINIT

`Tag = 0xFFFF`

NASM и числа с плавающей точкой

```
db –0.2                        ; «Четверть»
dw –0.5                        ; IEEE 754r/SSE5
                               ; половинная точность
dd 1.2                         ; одинарная точность
dd 1.222_222_222               ; допускается использовать
                               ; знак подчеркивания
dd 0x1p+2                      ; 1.0x2^2 = 4.0
dq 0x1p+32                     ; 1.0x2^32 = 4 294 967 296.0
dq 1.e10                       ; 10 000 000 000.0
dq 1.e+10                      ; синоним для 1.e10
dq 1.e–10                      ; 0.000 000 000 1
dt 3.14159265358979323846 2 ; число Пи
do 1.e+4000                    ; IEEE 754r четверная точность
```

- IEEE 754r – опубликован в 2008 году

NASM и числа с плавающей точкой

- __float8__
- __float16__
- __float32__
- __float64__
- __float80m__
- __float80e__
- __float128l__
- __float128h__

- __Infinity__
- __NaN__

```
dq +1.5, –__Infinity__, __NaN__
mov eax, __float32__(3.1415926)
```

Сложение двух чисел

```
%include 'io.inc‘

section .data
 x dd 11.2
 y dd 0.7

section .bss
 z resd 1

section .text
global CMAIN
```

```
CMAIN:
    finit
    fld dword [x]
    fld dword [y]
    faddp
    fstp dword [z]
    PRINT_HEX 4, z
    NEWLINE
    xor eax, eax
    ret
```

x

11.2 ~ 1.11999998092651367187 5E1

0x41333333 = 01000001 00110011 00110011 00110011

Сложение двух чисел

```
%include 'io.inc‘

section .data
 x dd 11.2
 y dd 0.7

section .bss
 z resd 1

section .text
global CMAIN
```

```
CMAIN:
    finit
    fld dword [x]
    fld dword [y]
    faddp
    fstp dword [z]
    PRINT_HEX 4, z
    NEWLINE
    xor eax, eax
    ret
```

y

0.7 ~ 6.99999988079071044921875E-1

0x3F333333 = 00111111 00110011 00110011 00110011

Сложение двух чисел

```
%include 'io.inc‘

section .data
 x dd 11.2
 y dd 0.7

section .bss
 z resd 1

section .text
global CMAIN
```

```
CMAIN:
    finit
    fld dword [x]
    fld dword [y]
    faddp
    fstp dword [z]
    PRINT_HEX 4, z
    NEWLINE
    xor eax, eax
    ret
```

z

0x413E6666 = 01000001 00111110 01100110 01100110

1.18999996185302734375E1 ~ 11.9

Печать числа

```
%include 'io.inc'

section .data
 x dd 11.2
 y dd 0.7

section .bss
 z resd 1

section .rodata
 lc db '%f', 10, 0

section .text
CEXTERN printf
global CMAIN
```

```
CMAIN:
    ; …
    ; пролог функции
    sub     esp, 20

    fld     dword [x]
    fld     dword [y]
    faddp
    fst     dword [z]
    fstp    dword [esp + 4]
    mov     dword [esp], lc
    call    printf

    add     esp, 20
    ; эпилог функции
    ; …
```

Печать числа

printf печатает мусор
где ошибка?!?!!1

```
%include 'io.inc'

section .data
 x dd 11.2
 y dd 0.7

section .bss
 z resd 1

section .rodata
 lc db '%f', 10, 0

section .text
CEXTERN printf
global CMAIN
```

```
CMAIN:
    ; …
    ; пролог функции
    sub     esp, 20

    fld     dword [x]
    fld     dword [y]
    faddp
    fst     dword [z]
    fstp    dword [esp + 4]
    mov     dword [esp], lc
    call    printf

    add     esp, 20
    ; эпилог функции
    ; …
```

Печать числа

ISO/IEC 9899:1999
§ 6.5.2.2 абзац №6

```
%include 'io.inc'

section .data
 x dd 11.2
 y dd 0.7

section .bss
 z resd 1

section .rodata
 lc db '%f', 10, 0

section .text
CEXTERN printf
global CMAIN
```

```
CMAIN:
    ; …
    ; пролог функции
    sub     esp, 20

    fld     dword [x]
    fld     dword [y]
    faddp
    fst     dword [z]
    fstp    qword [esp + 4]
    mov     dword [esp], lc
    call    printf

    add     esp, 20
    ; эпилог функции
    ; …
```

Порядок действий имеет значение

```
%include 'io.inc'

section .data
 x dq  3.14
 y dq  1e50
 z dq -1e50

section .bss
 r resq 1

section .rodata
 lc db '%f', 10, 0

section .text
CEXTERN printf
global CMAIN
```

```
CMAIN:
    ; … пролог функции
    sub     esp, 20

    fld     qword [x]
    fld     qword [y]
    fld     qword [z]
    faddp
    faddp
    fst     qword [r]
    fstp    qword [esp + 4]
    mov     dword [esp], lc
    call    printf

    add     esp, 20
    ; … эпилог функции
```

Порядок действий имеет значение

```
CMAIN:
    ; … пролог функции
    sub     esp, 20

    fld     qword [y]
    fld     qword [z]
    fld     qword [x]
    faddp
    faddp
    fst     qword [r]
    fstp    qword [esp + 4]
    mov     dword [esp], lc
    call    printf

    add     esp, 20
    ; … эпилог функции
```

```
CMAIN:
    ; … пролог функции
    sub     esp, 20

    fld     qword [x]
    fld     qword [y]
    fld     qword [z]
    faddp
    faddp
    fst     qword [r]
    fstp    qword [esp + 4]
    mov     dword [esp], lc
    call    printf

    add     esp, 20
    ; … эпилог функции
```

Распределение слагаемых

```
section .data
 x dq  1e200
 y dq  1e200
 z dq  1e200

section .bss
 r resq 1

section .rodata
 lc db '%lf', 10, 0
```

```
CMAIN:
    ; … пролог функции
    sub     esp, 20

    fld     qword [x]
    fld     qword [y]
    fsubp
    fld     qword [z]
    fmulp
    fst     qword [r]
    fstp    qword [esp + 4]
    mov     dword [esp], lc
    call    printf

    add     esp, 20
    ; … эпилог функции
```

Распределение слагаемых

```
section .data
 x dq  1e200
 y dq  1e200
 z dq  1e200

section .bss
 r resq 1

section .rodata
 lc db '%lf', 10, 0
```

```
CMAIN:
    ; … пролог функции
    sub     esp, 20

    fld     qword [x]
    fld     qword [z]
    fmulp
    fld     qword [y]
    fld     qword [z]
    fmulp
    fsubp
    ; вызов printf

    add     esp, 20
    ; … эпилог функции
```

Распределение слагаемых

```
section .data
 x dq  1e200
 y dq  1e200
 z dq  1e200

section .bss
 r resq 1
 cw resw 1

section .rodata
 lc db '%lf', 10, 0

CMAIN:
   ; … пролог функции
   sub     esp, 20
```

```
fstcw   word [cw]
and  word [cw], 11111111_11000000b
fldcw   word [cw]

fld     qword [x]
fld     qword [z]
fmulp
fld     qword [y]
fld     qword [z]
fmulp
fsubp
; вызов printf

add     esp, 20
; … эпилог функции
```

Польская обратная запись

- (w + x + y + z) / 4
- w x + y + z + 4 /

```
section .data
 w dq  1e10
 x dq  1e10
 y dq  1e10
 z dq  1e10
 d dd  4
```

```
section .text
    ;…
    fld     qword [w]
    fld     qword [x]
    faddp
    fld     qword [y]
    faddp
    fld     qword [z]
    faddp
    fild    dword [d]
    fdivp
    ;…
```

Предопределенные константы

- На «верхушку» стека регистров (ST0) помещается определенная константа
  - FLD1 +1.0
  - FLDL2T $\log_2 10$
  - L2E $\log_2 e$
  - FLDPI π
  - FLDLG2 $\log_{10} 2$
  - FLDLN2 $\log_e 2$
  - FLDZ +0.0

Сравнение чисел

```
_Bool isLe(double x, float y) {
   return x <= y;
}
```

Результат сравнения	C3	C2	C0
St0 > St1	0	0	0
St0 < St1	0	0	1
St0 == St1	1	0	0
неопределенно	1	1	1

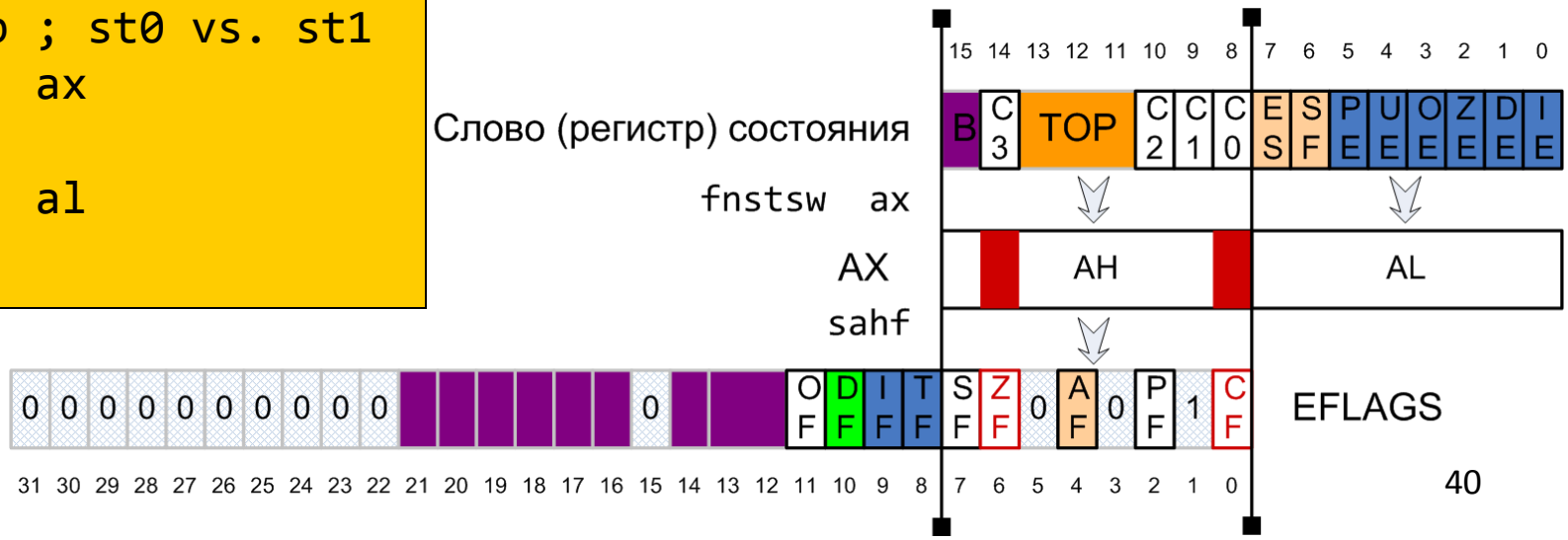
```
isLe:
        push    ebp
        mov     ebp, esp
        fld     dword [ebp+16]
        fld     qword [ebp+8]
        fucompp ; st0 vs. st1
        fnstsw  ax
        sahf
        setbe   al
        pop     ebp
        ret
```

Извлечение результатов сравнения

- C3 → ZF, C0 → CF
- Можно использовать условные коды, применяемые при сравнении беззнаковых чисел

Результат сравнения	C3	C2	C0
ST(0) > ST(i)	0	0	0
ST(0) < ST(i)	0	0	1
ST(0) == ST(i)	1	0	0
неупорядочены	1	1	1

```
isLe:
    ...
    fld     dword [ebp+16]
    fld     qword [ebp+8]
    fucompp ; st0 vs. st1
    fnstsw  ax
    sahf
    setbe   al
    ...
```



Функции: возвращение числа с плавающей точкой

```
void caller(double *p) {
   *p = inverse(*p);
}

caller:
   push    ebp
   mov     ebp, esp
   sub     esp, 8
   mov     eax, dword [ebp+8]
   fld     dword [eax]
   fstp    qword [esp]
   call    inverse
   mov     eax, dword [ebp+8]
   fstp    qword [eax]
   leave
   ret
```

```
float inverse(double x) {
   return 1/x;
}

inverse:
   push    ebp
   mov     ebp, esp
   fld1
   fld     qword [ebp+8]
   fdivp
   pop ebp
   ret
```