“Архітектура обчислювальних систем”.
Розділ 2. Програмування низького рівня.

Тема лекції:

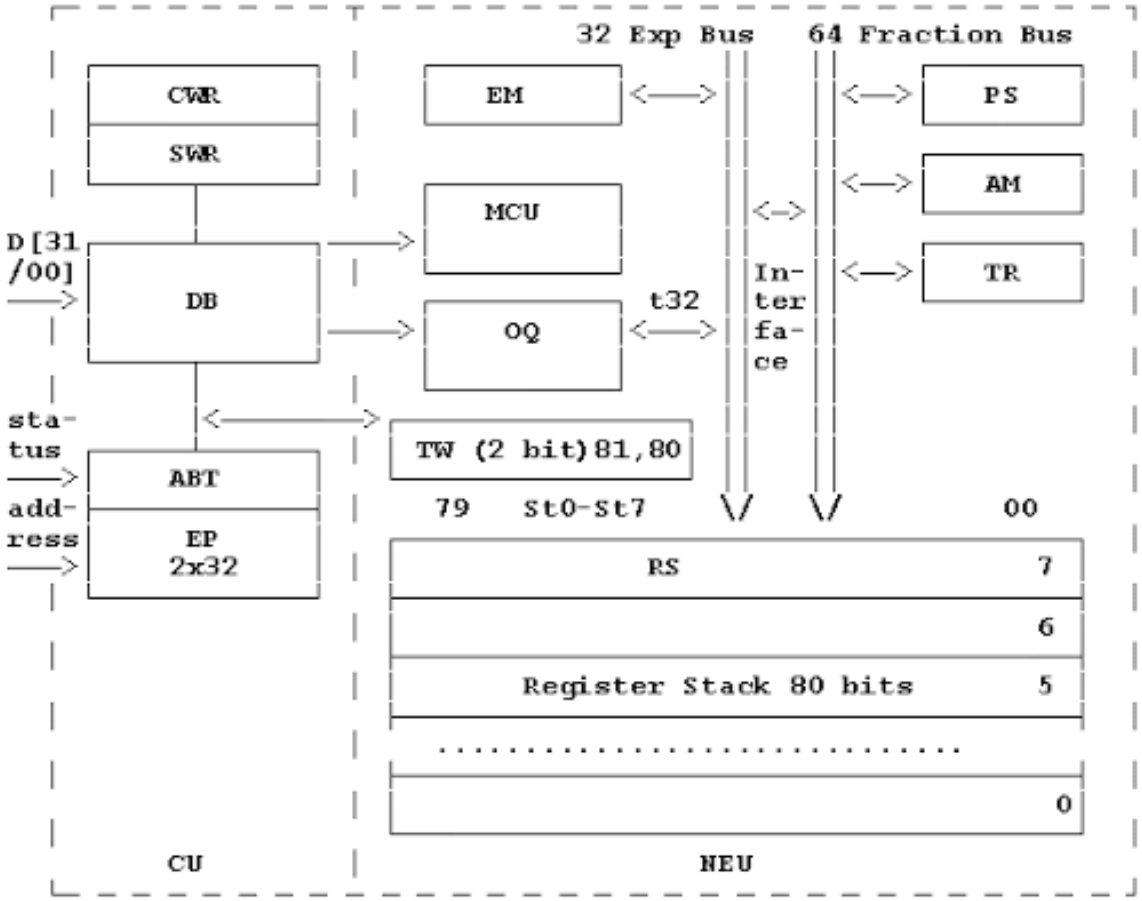
Архітектура і програмна модель співпроцесора, організація обчислень у FPU

План лекції

1. *Архітектура вузла операцій з плаваючою комою*
2. *Програмна модель співпроцесора*
3. *Система команд FPU*
4. *Порядок створення програм для співпроцесора*

МОДУЛЬ ЧИСЕЛ З ПЛАВАЮЧОЮ КОМОЮ FPU (Floating Point Unit)

1. Архітектура вузла операцій з плаваючою комою



Позначення на схемі:

CU - (Control Unit) пристрій управління,
NEU - (Numeral Execution Unit) пристрій цифрових процедур,
CWR - (Control Word Register) регістр керуючого слова,
SWR - (Status Word Register) регістр слова стану FPU,
DB - (Data Buffer) буфер даних,
ABT - (Addresing & Bus Tracking) адресація і стеження за станом системної шини.
EP - (Execution Pointer) покажчик процедури шини,
EM - (Exponent Module) модуль управління порядками чисел з FP,
MCU - (Microcode Control Unit) пристрій мікроуправління,
OQ - (Operands Queue) чергу операндів,
TW - (Tag Word) словник тегів і dirty bit в RS,
RS - (Register Stack) стекові регістри,
PS - (Programmable Shutter) програмований зсувач,
AM - (Arithmetic Module) модуль арифметичних процедур,
TR - (Temporary Registers) регістри тимчасового зберігання проміжних даних.

✎ **Співпроцесор** - спеціалізована інтегральна схема (або окремий вузол чи ядро), які працюють разом з ЦП, але призначені для виконання специфічного набору функцій.
Співпроцесор менш універсальний ніж ЦП, зокрема, не має лічильника команд.

Приклади: виконання операцій з дійсними числами - *математичний співпроцесор*, підготовка графічних зображень і тривимірних сцен - *графічний співпроцесор*, цифрова обробка сигналів - *сигнальний співпроцесор* тощо.

Можна виділити два способи обміну інформацією між ЦП і співпроцесором :

- пряме з'єднання вхідних і вихідних портів (ЦП має спеціальний інтерфейс для взаємодії із співпроцесором);
- з обміном через пам'ять (обмін інформацією між ЦП і співпроцесором відбувається завдяки доступу співпроцесора до оперативної пам'яті через системну магістраль).

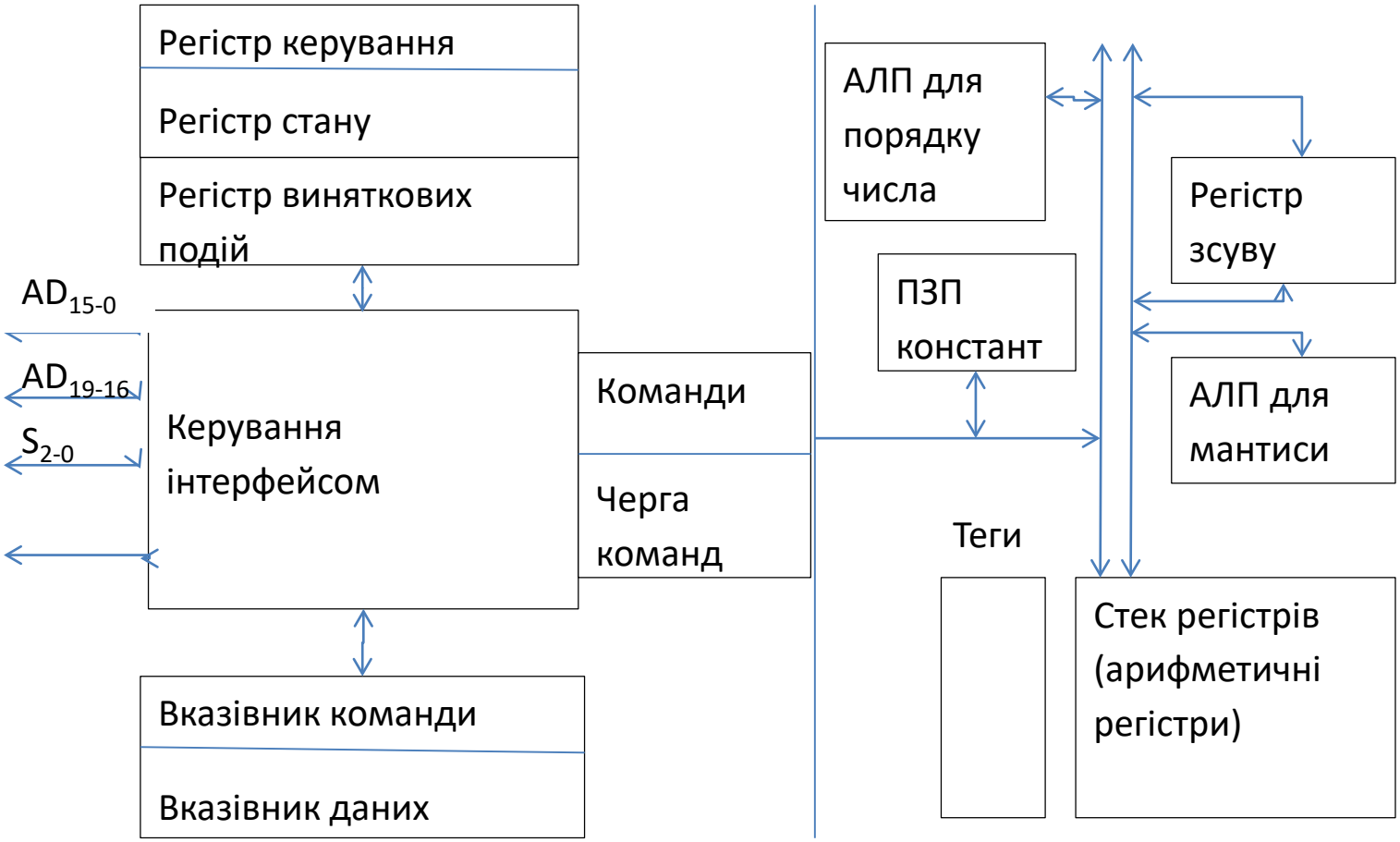


Рис. 1. Узагальнена структура співпроцесора

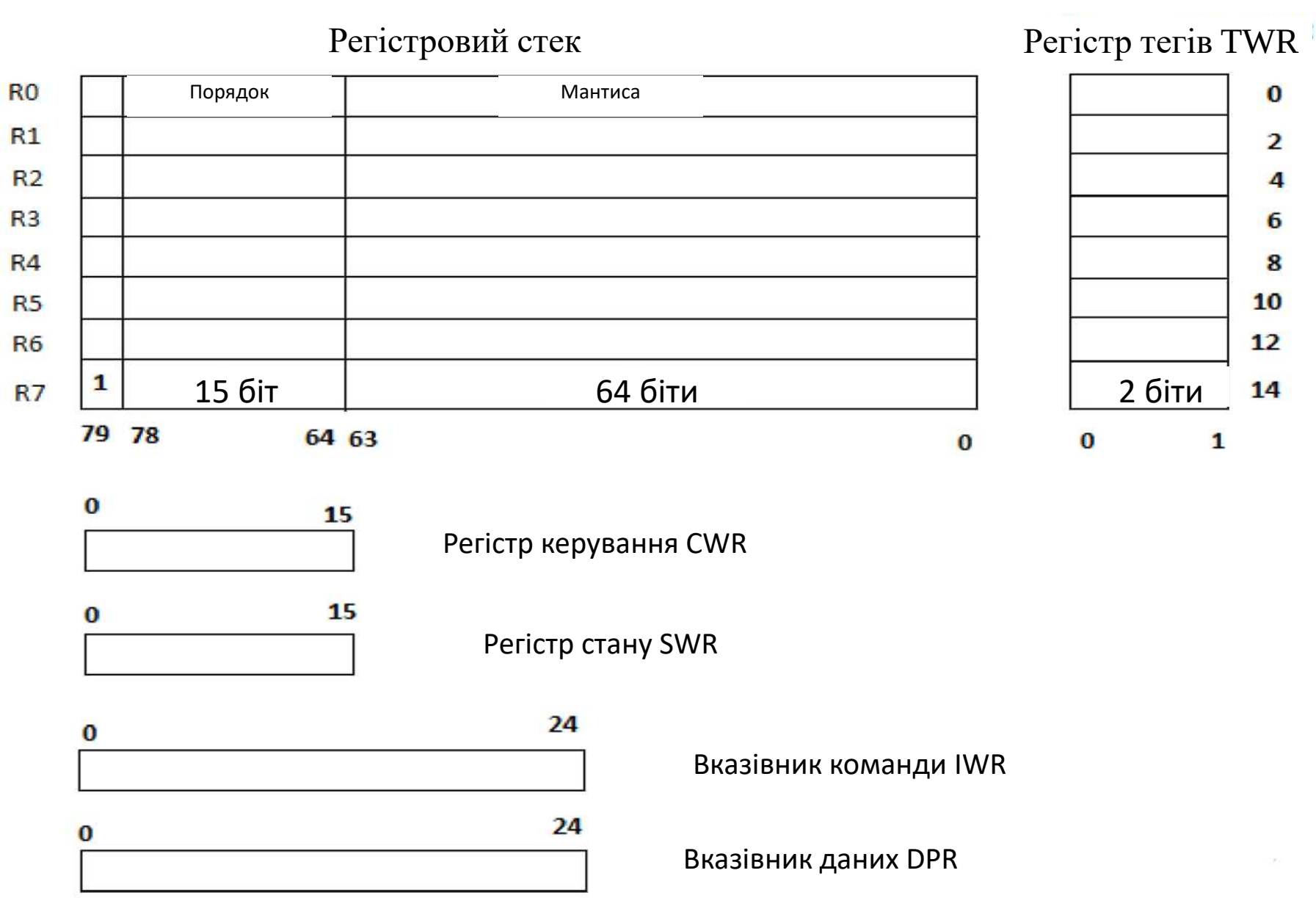


Рис. 2 - Регістри співпроцесора

Команди співпроцесора не оперують фізичними номерами регістрів стека R0 - R7. **Використовують логічні номери цих регістрів St(0) - St(7**).
Саме за допомогою логічних номерів реалізується відносна адресація регістрового стека співпроцесора.

Регістр стану співпроцесора SWR (Status Word Register) відбиває поточний стан співпроцесора після виконання команди.

У регістрі містяться поля, що дозволяють визначити: який регістр є вершиною стека, які винятки виникли після виконання команди, які особливості виконання команди.

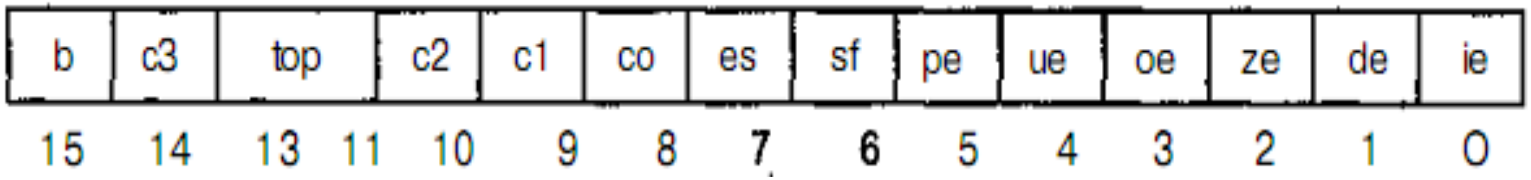
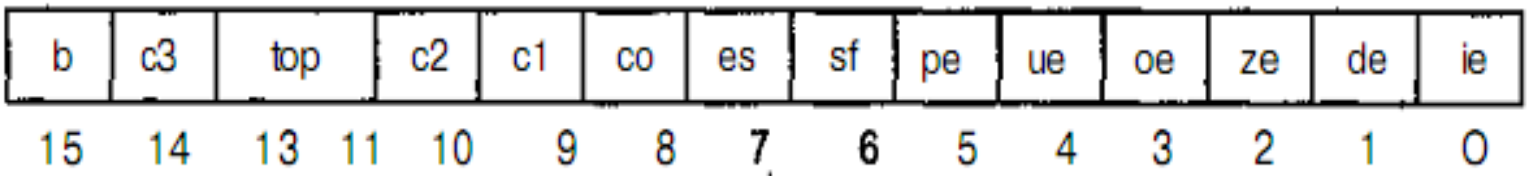


Рис. 3. Формат регістрів стану співпроцесора SWR

6 ознак виняткових ситуацій - усі можливі винятки зведені до шести типів, кожному з яких відповідає один біт регістра стану (прапорець):
біт 0 - недійсна операція (IE);
біт 1 - ненормалізований операнд (DE);
біт 2 - ділення на нуль (ZE);
біт 3 – переповнювання (OE);
біт 4 – антипереповнення (UE);
біт 5 - втрата точності (PE);

біт sf (Stack Fault) - помилка роботи стека співпроцесора, біт встановлюється в одиницю, якщо виникла одна з трьох виняткових ситуацій. Після аналізу цього біта його встановлюють в 0.
SF=1, C1=1 - переповнення стека;
SF=1, C1=0 - читання з порожнього стека;

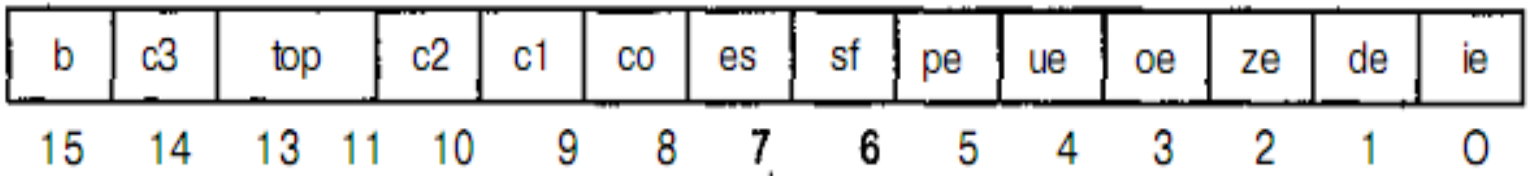
біт ES (Error Summary) - сумарна помилка співпроцесора - встановлюється в одиницю, якщо виникає будь-яка з шести перелічених вище виняткових ситуацій;



✎***коди умов (Condition Code) - 4 біти C0 -C3*** відображують результат виконання операції.
біт 8 - C0 (ознака переносу)
біт 9 - C1;
біт 10 - C2(ознака парності);
біт 14 - C3(ознака нуля);

✎**трибітове поле top** - *біти 11-13* - містить покажчик регістра поточної вершини стека;

біт 15 - біт зайнятості співпроцесора (B).



Регістр управління CWR(Control Word Register) визначає особливості обробки чисельних даних.

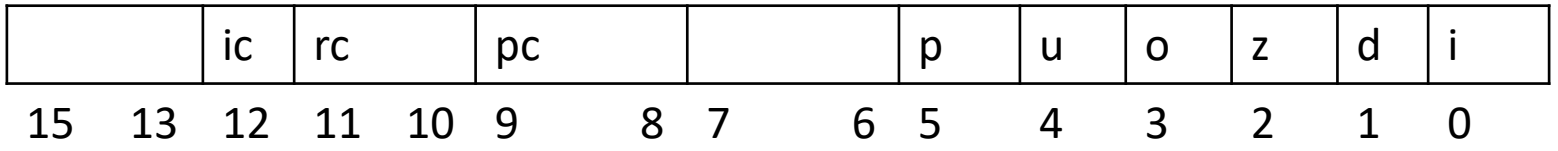
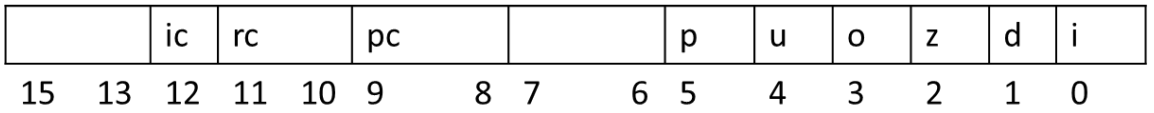


Рис. 4. Формат управління SWR

шість масок виключень - маски призначені для маскування виняткових ситуацій, виникнення яких фіксується за допомогою шести прапорів регістра стану.

біт 0 - виявлена недійсна операція ***(IM=0);***
біт 1 - виявлений ненормалізований операнд ***(DM=1);***
біт 2 - зафіксовано ділення ненульового числа на нуль ***(ZM=1);***
біт 3 - виявлено переповнювання ***(OM=1);***
біт 4 - виявлено антипереповнення(зникнення порядку ***UM=1);***
біт 5 - виявлена втрата точності ***(PM=1);***

Поле управління точністю(PC) - біти 8-9 - призначено для вибору довжини мантиси.



00 - 24 біта;
10 - 53 біта;
11 - 64 біта.
За замовченням значення поля встановлюється **PC**=11

Поля управління округленням (RC) - біти 10-11 - дозволяє управляти процесом округлення чисел під час роботи співпроцесора.
00- до найближчого цілого;
01 округлення в меншу сторону;
10 - округлення у більшу сторону;
11 - відкидання дробової частини (використовується для приведення числа до форми, яка використовується в операціях цілочисельної арифметики).

біт 12 - інтерпретації нескінченності **(IC=1).**
0 - беззнакова нескінченність; 1 - нескінченність зі знаком.

При ініціалізації співпроцесора сигналом RESET поля регістра управління встановлюються таким чином: IC = 0, RC = 00, PC == 11, EM = 0, усі маски встановлюються в одиницю.

Регістр тегів twr(Tag Word Register) — використовується для контролю стану кожного з регістрів R0 — R7.
Команда співпроцесора використовує цей регістр для визначення, наприклад, можливості запису в ці регістри.
Регістр відводить 2 біта на кожен регістр стека :
00- у регістрі не нульове дійсне число;
01- у регістрі істинний нуль;
10- у регістрі не число або нескінченність;
11- регістр порожній.

Регістр вказівник команди – ipr (Instraction Point Register) містить адресу команди, що викликала виняткову ситуацію, і 11 біт команди.

Регістр вказівник даних - dpr(data Point Register) містить адресу операнда (якщо використовувався) команди, що викликала виняткову ситуацію.

2. Основні правила створення програмних кодів для співпроцесора

Система команд співпроцесора включає близько 80 машинних команд.

Мінімальна довжина команди - 2 байти.

✎Усі команди умовно можна розбити на п'ять груп:

- передачі даних;
- арифметичні;
- порівняння;
- трансцендентних операцій;
- управління.

Мнемонічне позначення команд співпроцесора характеризує особливості їх роботи. У зв'язку з цим в мнемоніці прийняті наступні угоди:

- Перша буква завжди **F.**
- друга буква визначає тип операнда в пам'яті, з яким працює команда :

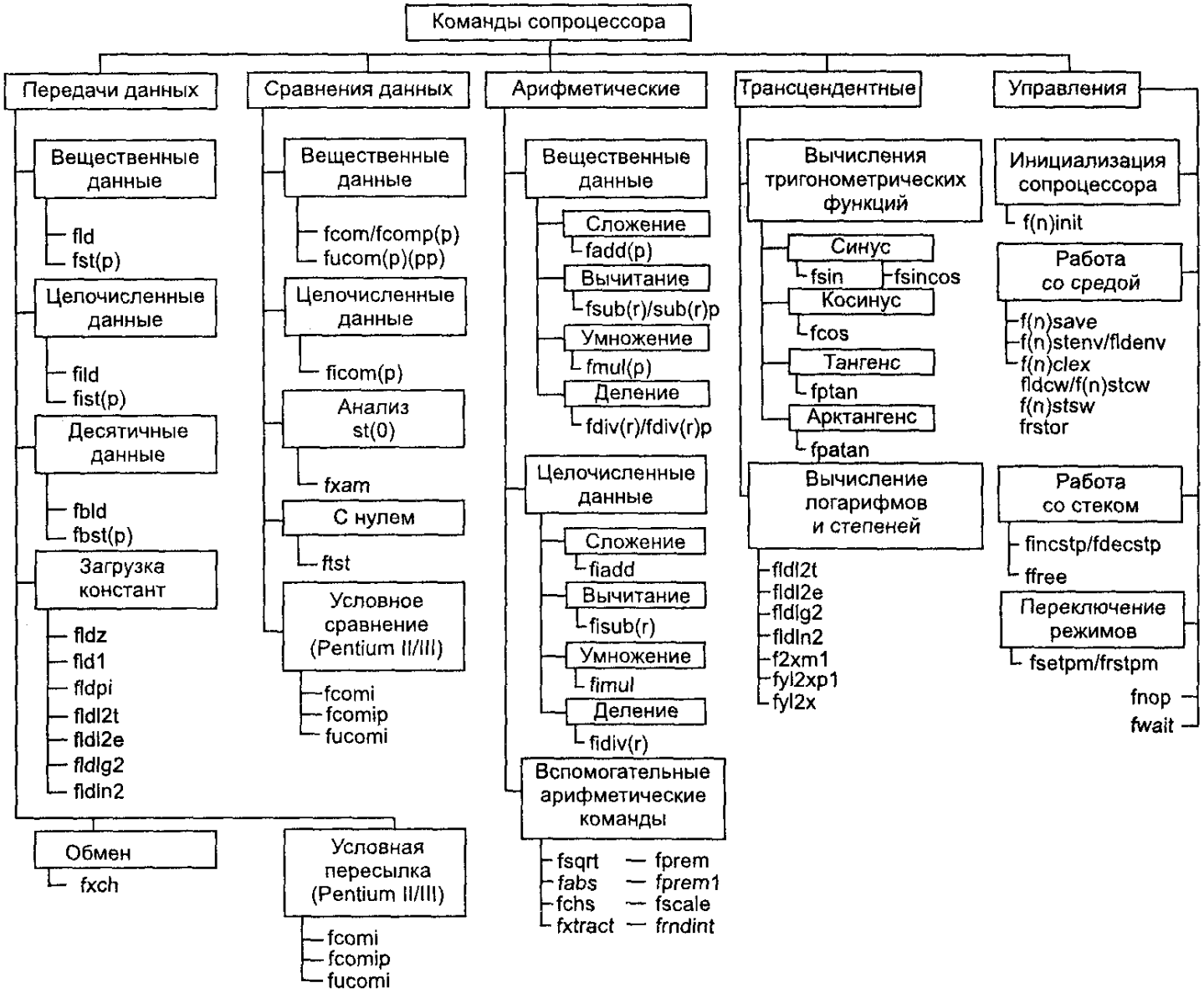
I - *означає операцію з цілим числом з пам'яті;*

B - *операцію з десятковим числом з пам'яті;*

за відсутності цих букв - операція виконується з дійсними числами.

- Передостання або остання букв **R(reversed)** вказує зворотний порядок операндів (для віднімання і ділення, оскільки для цих команд дуже важливий порядок дотримання операндів).
- Остання буква **P** ідентифікує команду, останньою дією якої є витягання із стека операнда.

Класифікація машинних команд співпроцесора



Класифікація машинних команд співпроцесора

Мнемонічне позначення команд співпроцесора характеризує особливості їх роботи і в зв'язку з цим може представляти певний інтерес. Тому коротко розглянемо основні моменти утворення назв команд:

- всі мнемонічні позначення починаються з символу **f** (`float`);
- друга буква мнемонічного позначення визначає тип операнда в пам'яті, з яким працює команда: - **i** - ціле двійкове число; - **b** - ціле десяткове число; - **відсутність букви** - дійсне число;
- остання буква мнемонічного позначення команди **p** означає, що останньою дією команди обов'язково є отримання операнда зі стека;
- остання або передостання буква **r** (reversed) означає реверсивний спосіб проходження операндів при виконанні команд віднімання і ділення, так як для них важливий порядок проходження операндів.

Система команд співпроцесора відрізняється великою гнучкістю у виборі варіантів завдання команд, що реалізують певну операцію, і їх операндів. Мінімальна довжина команди співпроцесора - 2 байта. Всі команди співпроцесора оперують регістрами стека співпроцесора. Якщо операнд в команді не вказується, то за замовчуванням використовується вершина стека співпроцесора (логічний регістр `st (0)`). Якщо команда виконує дію з двома операндами за замовчуванням, то ці операнди - регістри `st (0)` і `st (1)` .

Класифікація машинних команд співпроцесора

Команди передачі даних дійсного типу

Використовуються в разі якщо операнд, застосовуваний в команді, має дійсний тип (4, 8 або 10-байтний).

команда	операнди	пояснення	опис
FLD	src	TOP $_{SWR}$ - = 1; ST (0) = src;	Завантаження операнда в вершину стека
FST	dst	dst = ST (0);	Збереження вершини стека в пам'ять
FSTP	dst	dst = ST (0); TOP $_{SWR}$ + = 1;	Збереження вершини стека в пам'ять з виштовхуванням
FXCH	ST (i)	ST (0) ↔ ST (i)	Обмін значень ST (0) і ST (i)

Класифікація машинних команд співпроцесора

Команди передачі даних цілого типу
Використовуються в разі якщо операнд, застосовуваний в команді, має цілий тип (1, 2, 4 або 8-байтний).

команда	операнди	пояснення	опис
FILD	src	TOP $_{SWR}$ - = 1; ST (0) = src;	Завантаження операнда в вершину стека
FIST	dst	dst = ST (0);	Збереження вершини стека в пам'ять
FISTP	dst	dst = ST (0); TOP $_{SWR}$ + = 1;	Збереження вершини стека в пам'ять з виштовхуванням

Класифікація машинних команд співпроцесора

Команди передачі даних цілого типу
Використовуються в разі якщо операнд, застосовуваний в команді, має цілий тип (1, 2, 4 або 8-байтний).

команда	операнди	пояснення	опис
FILD	src	TOP $_{SWR}$ - = 1; ST (0) = src;	Завантаження операнда в вершину стека
FIST	dst	dst = ST (0);	Збереження вершини стека в пам'ять
FISTP	dst	dst = ST (0); TOP $_{SWR}$ + = 1;	Збереження вершини стека в пам'ять з виштовхуванням

Класифікація машинних команд співпроцесора

Арифметичні команди дійсного типу Схема розташування операндів речових команд традиційна для команд співпроцесора. Перший операнд за замовчуванням (якщо не вказано в команді) розташовується в вершині стека співпроцесора - регістрі `ST (0)`, І на його місце після виконання команди записується результат. Другий операнд може бути розташований або в пам'яті, або в іншому регістрі стека співпроцесора. За замовчуванням в якості другого операнда використовується регістр `ST (1)` . Допустимими типами операндів в пам'яті є дійсні формати простий і подвійної точності. На відміну від цілочисельних арифметичних команд, дійсні арифметичні команди допускають більшу різноманітність в поєднанні розташування операндів і самих команд для виконання конкретної арифметичної дії.

команда	операнди	пояснення	опис
FADD	dst, src	dst = dst + src;	додавання дійсних
FADDP	ST (i), ST (0)	ST (i) = ST (i) + ST (0); TOP $_{SWR}$ + = 1;	Додавання дійсних з виштовхуванням
FSUB	dst, src	dst = dst - src;	віднімання дійсних
FSUBP	ST (i), ST (0)	ST (i) = ST (i) - ST (0); TOP $_{SWR}$ + = 1;	Віднімання дійсних з виштовхуванням
FSUBR	dst, src	dst = src - dst;	Віднімання дійсних реверсивний
FSUBRP	ST (i), ST (0)	ST (i) = ST (0) - ST (i); TOP $_{SWR}$ + = 1;	Віднімання дійсних реверсивне з виштовхуванням
FMUL	dst, src	dst = dst * src;	множення дійсних
FMULP	ST (i), ST (0)	ST (i) = ST (i) * ST (0); TOP $_{SWR}$ + = 1;	Множення дійсних з виштовхуванням
FDIV	dst, src	dst = dst / src;	ділення дійсних
FDIVP	ST (i), ST (0)	ST (i) = ST (i) / ST (0); TOP $_{SWR}$ + = 1;	ділення дійсних з виштовхуванням
FDIVR	dst, src	dst = src / dst;	ділення дійсних реверсивне
FDIVRP	ST (i), ST (0)	ST (i) = ST (0) / ST (i); TOP $_{SWR}$ + = 1;	ділення дійсних реверсивне з виштовхуванням

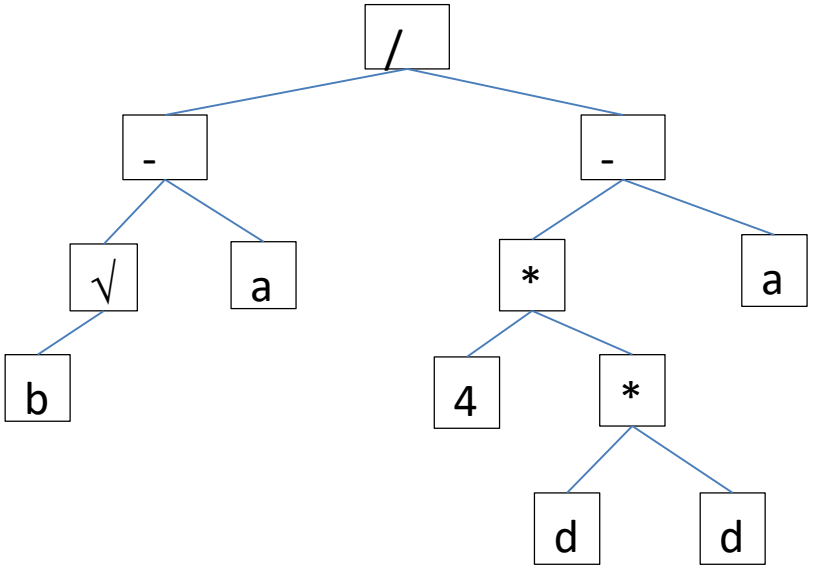
3. Порядок створення програм для співпроцесора

Для ефективного опису алгоритмів чисельних операцій використовують дерево обчислень. Обхід дерева з крайньої лівої вершини визначає послідовність обчислень. При цьому перехід до кореня виконується тільки тоді, коли результати усіх нижніх вершин пораховані.

✎ Алгоритм обробки дерева :

1. Якщо аналізована вершина операнд - занести його значення в стек.
2. Якщо чергова вершина - операція - виконати її над одним або двома операндами, визначуваними піддеревами. Результат операції помістити у вершину стека.
3. Якщо в дереві ще є вершини - перейти до кроку 1, інакше у вершині стека знаходиться результат.

Приклад 1. Нехай необхідно обчислити значення виразу
$Y= (a- \sqrt{b})/(4*d^2-a)$



Приклад1 :

```
finit       ; ініціалізація співпроцесора
fld b       ; занести в стек b
fsqrt       ;визначення квадратного кореня  b
fsub a      ; віднімання √b-a
fchs        ; зміна знаку виразу
fld d       ; занести в стек d
fmul d      ; розрахунок d* d
fld c4      ; занесення в стек 4
fmul        ; розрахунок 4* d* d
fsub a      ; розрахунок 4* d* d - a
fdiv        ; √b-a/4* d* d - a
fstp rez    ; занесення в пам’ять результату з
виштовхуванням зі стеку
```

Приклад 2:
Y= (tg(a) +b) (c*c - 1)

	St(0)	St(1)
Finit		
Fld pi	Pi	
Fld a	A	Pi
fprem	Залишок a/pi	
fptan	X	Y
fdiv	X/Y	
fadd b	X/Y+b	
fld c	C	X/Y+b
fmul c	C*c	X/Y+b
fsub one	C*c - 1	
fdivp st(1), st(0)	(X/Y+b) (c*c - 1)	
fstp y		
ret		