Лабораторна робота № 7

# ОБЧИСЛЕННЯ АРИФМЕТИЧНИХ ВИРАЗІВ І ТРАНСЦЕНДЕНТНИХ ФУНКЦІЙ (СПІВПРОЦЕСОР ix87)

*Мета заняття*: ознайомитися з основними командами мови Assembler для обчислення арифметичних виразів і трансцендентних функцій; набути практичних навичок в написанні програм, які обчислюють заданий змішаний арифметичний вираз, використовуючи арифметичні операції співпроцесора на мові Assembler.

## Теоретичні відомості

Математичний співпроцесор (FPU – Floating Point Unit) – спеціальний пристрій, який слугує для обробки дійсних чисел (чисел з плаваючою комою).

З моменту свого виникнення співпроцесор розширював обчислювальні можливості основного процесора i8086 (i80286, i82386, i80486) і спочатку був виконаний у вигляді окремої мікросхеми i8087 (i80287, i80387, i80487).Його присутність в перших моделях процесора була не обов'язковою. Починаючи з сімейства процесорів i486DX співпроцесор став складовою частиною основного процесора.

Сучасний співпроцесор забезпечує повну підтримку стандартів IEEE-754 і IEEE-854 по представленню і обробці чисел з плаваючою комою.

Він може виконувати трансцендентні операції (обрахування тригонометричних функцій, логарифмів і т.д.) з більшою точністю.



## *Типи даних*

Співпроцесор може виконувати операції з сімома звичайними типами даних, а також із спеціальними числами.

Звичайні дані можуть бути дійсними або цілими числами із знаком. Наявність цілочисельних даних дозволяє виконувати так звані змішані обрахування, коли в якості операндів використовуються і цілі, і дійсні числа.

Співпроцесор виконує всі обрахування в 80бітному розширеному дійсному форматі, а 16-, 32- і 64-бітні дані використовуються для обміну даними і можуть застосовуватись в командах співпроцесора як операнди.

Таблиця 8.1

**Типи звичайних даних співпроцесора**

| **Тип даних** | **Довжина (біт)** | **К-сть значимих цифр** | **Діапазон представлення** |
|---|---|---|---|
| Ціле слово | 16 | 4-5 | -32768…32767 |
| Коротке ціле | 32 | 9-10 | -2147483648 … -2147483647 |
| Довге ціле | 64 | 18-19 | -9223372036854775808 … 9223372036854775807 |

| | | | |
|---|---|---|---|
| Упаковане двійково-десяткове | 80 | 18 | -999999999999999999 … 999999999999999999 |
| Коротке дійсне | 32 | 7-8 | 1.175494351e-38 … 3.402823466e+38 |
| Довге дійсне | 64 | 15-16 | 2.2250738585072014e-308... 1.7976931348623158e+308 |
| Розширене дійсне | 80 | 19-20 | 3.3621031431120935063e-4932… 1.18973149535723176 5e+4932 |

Крім звичайних чисел, специфікація стандарту IEEE передбачає декілька спеціальних форматів, які можуть виникнути в результаті виконання математичних операцій співпроцесора і над якими можна виконувати окремі операції.

- **Додатний нуль –** всі біти числа скинуті до нуля.
- **Від'ємний нуль** – знаковий біт дорівнює 1, всі інші біти числа скинуті до нуля.
- **Додатна нескінченність –** знаковий біт дорівнює 0, всі біти експоненти установлені в 1, а біти мантиси скинуті до нуля.
- **Від'ємна нескінченність -** знаковий біт дорівнює 1, всі біти експоненти установлені в 1, а біти мантиси скинуті до 0.
- **Денормалізовані числа –** всі біти експоненти скинуті до 0. Ці числа дозволяють представляти дуже маленькі числа при обрахунках з розширеною точністю.

- **Невизначеність –** знаковий біт дорівнює1, перший біт мантиси дорівнює 1 (для 80-розрядних чисел перші два біта дорівнюють 11), інші біти мантиси скинуті до нуля, всі біти експоненти встановленні в 1.
- **Не-число типу SNAN (сигнальне) –** перший біт мантиси дорівнює 0 (для 80-розрядних чисел перші два біти дорівнюють 10), інші біти мантиси можуть містити 0 або 1, всі біти експоненти встановлені в 1.
- **Не-число типу QNAN (тихе) –** перший біт мантиси дорівнює 1 (для 80-розрядних чисел перші два біти дорівнюють 11), інші біти мантиси можуть містити 0 або 1, всі біти експоненти встановлені в 1.
- **Непідтримуване число –** всі інші ситуації.

### *Регістри*

У співпроцесора є 8 регістрів для зберігання даних R0-R7 і 5 допоміжних регістрів, які складають середовище співпроцесора.

Регістри даних співпроцесора R0-R7 мають довжину 80 біт (тобто 16-розрядних слів) і розглядаються як круговий стек, вершина якого (TOP) називається ST або ST(0) і є плаваючою.

Принцип роботи з круговим стеком співпроцесора аналогічний звичайному калькулятору. Будь-яка команда завантаження даних співпроцесора

автоматично переміщує вершину стеку співпроцесора: TOP = TOP + 1.

В таблиці 8.2 показана гіпотетична ситуація, коли в результаті виконання якоїсь команди вершиною стека став регістр R6. Інші регістри розподіляються по колу: R7-ST(1), R0-ST(2), …, R5-ST(7). Це ї є їх поточні імена ST(i), i=1,…,7 на момент виконання даної команди співпроцесора.

Якщо в цих регістрах є дані, то вони можуть слугувати операндами в командах співпроцесора. Звертатися ж напряму до регістрів R0-R7 не можна.

Таблиця 8.2

**Регістри співпроцесора і поняття вершини стеку співпроцесора**

| **Назва регістрів** | **Біти** | | | | | | | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
| CWR | Регістр управління (**Control Word Register**) | | | | | | | | | | | | | | | |
| SWR | Регістр стану співпроцесора **(Status Word Register)** | | | | | | | | | | | | | | | |
| TWR | Слово ознак-тегів **(Tags Word Register)** | | | | | | | | | | | | | | | |

Продовження табл.8.2

**Регістри співпроцесора і поняття вершини стеку співпроцесора**

| | |
|---|---|
| FIP (32 біти) | Регістр вказівника останньої виконаної команди |
| | **(Floating Instruction Pointer)** |
| FDP (32 ) | Регістр, де зберігається адреса операнда останньої виконаної команди |
| | **(Floating Data Pointer)** |
| R0 – ST (2) | |
| | |

| | |
|---|---|
| (80 біт) | Розширене дійсне чи будь-яке інше доступне дане співпроцесора |
| | |
| | |
| R1 – ST (3) | |
| | |
| (80 біт) | Розширене дійсне чи будь-яке інше доступне дане співпроцесора |
| | |
| | |
| …… | |
| R6 – (TOP) | |
| | |
| (80 біт) | Розширене дійсне чи будь-яке інше доступне дане співпроцесора |
| | |
| | |
| R7 – ST (1) | |
| | |
| (80 біт) | Розширене дійсне чи будь-яке інше доступне дане співпроцесора |
| | |
| | |

Регістр стану співпроцесора сигналізує співпроцесору про його стан і про те, як виконалась та чи інша арифметична команда. Таким чином, цей регістр виконує функції, аналогічні функціям регістра прапорців процесора FLAGS. За бітами регістра стану

співпроцесора закріплені відповідні імена, які полегшують розуміння їх призначення.

Таблиця 8.3

**Регістр стану співпроцесора SWR**

| **Регістр (слово) стану співпроцесора (SWR)** | | | | | | | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| **Старший байт** | | | | | | | | **Молодший байт** | | | | | | | |
| 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
| B | C3 | TOP | | | C2 | C1 | C0 | ES | SE | PE | UE | OE | ZE | DE | IE |
| **Аналогія з регістром прапорців процесора (Flags)** | | | | | | | | | | | | | | | |
| | ZF | | | | PF | | CF | | | | | | | | |

Регістр управління визначає для співпроцесора варіанти обробки дійсних чисел (таблиця 8.4).

Таблиця 8.4

**Регістр управління співпроцесора SWR**

| **Регістр (слово) управління співпроцесора (CWR)** | | | | | | | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| **Старший байт** | | | | | | | | **Молодший байт** | | | | | | | |
| 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
| | | | IC | RC | | PC | | IEM | | PM | UM | OM | ZM | DM | IM |

Молодші біти слова управління визначають маскування обрахованих виключень. Біти 0-5 містять індивідуальні маски для кожного із шести виключень, які розглядаються процесором при виконанні операцій

з плаваючою комою. Старші біти (8-12) управляючого слова визначають параметри роботи співпроцесора.

Регістр тегів містить 8 пар бітів, які описують вміст кожного регістра даних R0-R7:

00 – число;

01 – істинний нуль;

10 – особливе число; 11

– регістр пустий.

Таблиця 8.5

**Регістр (слово) тегів (TWR)**

| **Регістр (слово) тегів (TWR)** | | | | | | | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| **Старший байт** | | | | | | | | **Молодший байт** | | | | | | | |
| 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
| R7 | | R6 | | R5 | | R4 | | R3 | | R2 | | R1 | | R0 | |

### *Система команд*

Система команд співпроцесора досить проста, якщо знати ключ і розуміти англійську. Для їх підключення необхідно зробити наступне:

1) Скористатись однією із директив Асемблера: .8087, .287 (.286p), .387 (.386p.), .487(.486p). Необхідно мати на увазі, що не всі команди процесора сумісні зверху вниз.

2) Зробити ініціалізацію співпроцесора за допомогою команди FINIT.

3) При компіляції використовувати додатковий ключ.

Базовий співпроцесор i8087 має 69 базових команд, які підкоряються наступним закономірностям:

• Всі вони починаються на букву F (Floating).

• Друга буква може бути пов’язана з форматом оброблюваних чисел:

**I** (Integer) – цілі числа;

**B** (Binary-coded decimal) – упаковане двійководесяткове число (BCD).

Для дійсних чисел спеціальна буква не виділяється.

• Далі іде мнемонічний код операції (МКОП), наприклад:

**LD**(LoaD) – завантажити дане в вершину стека ST;

**ST**(STore) – вивантажити дане із вершини стека ST і уже відомі нам команди **ADD, MUL, DIV** і т.д. • Закінчується команда може буквою **R** (Reserved – реверсна операція) і/або **P** (Pop – виштовхнути результат із стека і звільнити вершину стека ST).

Таблиця 8.6

**Звичайні і реверсні операції**

| МКОП | Звичайна операція | Реверсна операція |
|---|---|---|
| SUB | Приймач=Приймач-Джерело | Приймач=Джерело-Приймач |
| DIV | Приймач=Приймач/Джерело | Приймач=Джерело/Приймач |

Співпроцесори пізніших моделей, починаючи з i80387, порушили цю систему умовних позначень, зате

збільшили функціональні можливості команд по обробці даних.

*1.1 Команди пересилки даних*

Всі команди цієї групи мають один операнд: або джерело, або приймач. Завантаження або вивантаження інформації відбувається в/з вершини стека
ST(0).

Зазвичай вершину стека позначають через ST, і в командах цієї групи вона явно в операндах не є присутньою.

Таблиця 8.7

**Команди передачі даних**

| Команда | Тип даних | Характеристика операнда | Приклад |
|---|---|---|---|
| **Команди завантаження даних в стек** | | | |
| FLD джерело | Дійсне | Змінна в ОЗП або ST(i), i=0,…7 | FLD a<br>FLD ST(5) |

Продовження табл.8.7

**Команди передачі даних**

| | | | |
|---|---|---|---|
| FBLD джерело | Типу BCD | Змінна в ОЗП | FBLD aBc |
| FILD джерело | Ціле | Змінна в ОЗП | FILD ka |
| **Команди кооперування даних з стеку** | | | |
| FST приймач | Дійсне | Змінна в ОЗП або ST(i), i=1,…7 | FST ap FST ST(1) |
| FIST приймач | Ціле | Змінна в ОЗП | FIST kab |
| **Команди вивантаження даних з стеку із звільненням вершини стеку** | | | |
| FSTP приймач | Дійсне | Змінна в ОЗП або ST(i), i=1,…7 | FSTP app<br>FSTP ST(5) |

| | | | |
|---|---|---|---|
| FBSTP приймач | Типу BCD | Змінна в ОЗП | FBSTP ppp |
| FISTP приймач | Ціле | Змінна в ОЗП | FISTP kabP |
| **Команда обміну вмісту джерела з ST(0)** | | | |
| FXCH (джерело) | Якщо джерело не вказане, то вважається, що він не відповідає ST (1) | ST(i), i=1,…7 | FXCH<br>FXCH ST(4) |

*1.2 Команди завантаження констант* Команди цієї групи поміщають в вершину стека ST(0) константи, які часто використовуються. Починаючи з співпроцесора i80987, ці константи зберігаються в більш точному форматі. Команди операндів не мають.

Таблиця 8.8

**Команди завантаження констант**

| **Команда** | **Призначення** |
|---|---|
| FLDI | Помістити в ST (0) число 1.0 |
| FLDZ | Помістити в ST (0) число +0.0 |
| FLDPI | Помістити в ST (0) число p |

Продовження табл.8.8

**Команди завантаження констант**

| | |
|---|---|
| FLDL2E | Помістити в ST (0) число, рівне $\log_2 e$ |
| FLDL2T | Помістити в ST (0) число, рівне $\log_2 10$ |
| FLDLN2 | Помістити в ST (0) , рівне ln 2 |
| FLDLG2 | Помістити в ST (0) , рівне lg 2 |

*1.3 Арифметичні команди*

Арифметичні команди реалізують базову арифметику співпроцесора (чотири основні арифметичні операції: +, -, *, /) і декілька спеціальних арифметичних команд.

Формат команд базової арифметики для дійсних чисел досить складний (операнди можуть бути задати в явному вигляді, а можуть і бути відсутні, тобто не явні), тому, щоб уникнути непорозумінь, в таблиці 8.9 даний явний формат.

Спеціальні арифметичні команди в своєму форматі не містять операндів, тобто всі їх операнди задані неявно.

Таблиця 8.9

**Команди базової арифметики**

| Команда | Тип даних | Алгоритм виконання |
|---|---|---|
| **Команди додавання** | | |
| FADD приймач, джерело | Дійсне | Приймач=приймач+джерело |
| FADDP приймач, джерело | Дійсне | |
| FIADD приймач | Ціле | |
| **Команди звичайного віднімання** | | |
| FSUB приймач, джерело | Дійсне | Приймач=приймач-джерело |
| FSUBP приймач, джерело | Дійсне | |
| FISUB джерело | Ціле | |

Продовження табл.8.9

**Команди базової арифметики**

| | | |
|---|---|---|
| **Команди реверсного (зворотного) віднімання** | | |
| FSUBR приймач, джерело | Дійсне | Приймач=джерело-приймач |
| FSUBPR приймач, джерело | Дійсне | |

| | | |
|---|---|---|
| FISUBR джерело | Ціле | |
| **Команди множення** | | |
| FMUL приймач, джерело | Дійсне | Приймач=приймач*джерело |
| FMULP приймач, джерело | Дійсне | |
| FIMUL джерело | Ціле | |
| **Команди звичайного ділення** | | |
| FDIV приймач, джерело | Дійсне | Приймач=приймач/джерело |
| FDIVP приймач, джерело | Дійсне | |
| FIDIV джерело | Ціле | |
| **Команди реверсного (зворотного) ділення** | | |
| FDIVR приймач, джерело | Дійсне | Приймач=джерело/приймач |
| FDIVRP приймач, джерело | Дійсне | |
| FIDIVR джерело | Ціле | |

Таблиця 8.10

## Спеціальні арифметичні команди

| Команда | Призначення | Алгоритм виконання |
|---|---|---|
| FPREM | Знаходження часткової остачі від ділення шляхом послідовного віднімання 64 рази вмісту ST(1) з ST(0) | ST(0)=остача[ST(0)/ST(1)] Формується прапорець C2 регістра управління CWR: C2=0 – отримана точна остача від ділення, тобто остача < ST(1) C2=1 – точна остача не отримана |
| FPREM1 (i80387) | Знаходження часткової остачі від ділення в стандарті IEEE – округлення до найбільшого цілого | |
| FABS | Знаходження абсолютного значення | ST(0)=abs(ST(0)) |
| FSQRT | Знаходження квадратного кореня | ST(0)=sqrt(ST(0)) |

Продовження табл.8.10

**Спеціальні арифметичні команди**

| | | |
|---|---|---|
| FCHS | Зміна знаку | ST(0)= -ST(0) |
| FRNDINT | Округлення до цілого | Вміст ST(0) округляється до цілого у відповідності з бітами RC регістра управління співпроцесором CWR |
| FXTRACT | Вивантажити експоненту і мантису числа | Число ===> ST(0)<br>ST(0) = мантиса числа<br>ST(1) = експонента числа |
| FSCALE | Команда, зворотна FXTRACT | ST(0) <=== мантиса числа<br>ST(1) <=== експонента числа<br>ST(0) = ST(0)*$2^{ST(1)}$ |

*1.4 Трансценденті операції*

Команди цієї групи в своєму форматі не містять операндів, тобто їх операнди задані неявно. Вони призначені для обрахування найбільш вживаних арифметичних функцій.

Таблиця 8.11

**Команди трансцендентних операцій**

| Команда | Призначення | Алгоритм виконання |
|---|---|---|
| FSIN (i80387) | Обрахування синуса sin(x), де значення аргумента x задається в радіанах | x ===> ST(0); ST(0) = sin(ST(0))<br>$-2^{63}<=x<=2^{63}$. Якщо значення x виходить за ці межі, можна скористатися командою FPREM з дільником 2П. Інакше прапорець C2=1 і значення ST(0) не змінюється. |

| | | |
|---|---|---|
| FCOS (i80387) | Обрахування синуса sin(x), де значення аргумента x задається в радіанах | x ===> ST(0); ST(0) = cos(ST(0))<br>$-2^{63}<=x<=2^{63}$. Якщо значення x виходить за ці межі, можна скористатися командою FPREM з дільником 2П. Інакше прапорець C2=1 і значення ST(0) не змінюється. |

Продовження табл.8.11

**Команди трансцендентних операцій**

| | | |
|---|---|---|
| FSINCOS (i80387) | Обрахування синуса і косинуса | x ===> ST(0);<br>ST(1) = sin(ST(0)); ST(0) = cos(ST(0))<br>$-2^{63}<=x<=2^{63}$. Якщо значення x виходить за ці межі, можна скористатися командою FPREM з дільником 2П. Інакше прапорець C2=1 і значення ST(0) не змінюється. |
| FPTAN | Обрахування часткового тангенса tg(a)=y/x, де значення аргумента a задається в радіанах | a ===> ST(0);<br>ST(0) = x; ST(1) = y;<br>0<a<П/4 (i8087, i80287).<br>$-2^{63}<=a<=2^{63}$. Якщо значення a виходить за ці межі, можна скористатися командою FPREM з дільником П. Інакше прапорець C2=1 і значення ST(0) не змінюється. |
| FPATAN | Обрахування арктангенса a=arctg (y/x) | ST(0) = x; ST(1) = y;<br>a <=== ST(0); знак співпадає із знаком ST(1)<br>0<\|y\|<\|x\|<нескінченність. \|a\| < П. |
| F2XM1 | Обрахування $2^X$-1 | X ===> ST(0);<br>ST(0) = $2^X - 1$<br>-1<X<1, інакше результат операції не визначений. |
| FYL2X | Обрахування $y*\log_2(x)$ | x ===> ST(0); y ===> ST(1);<br>0 < x < нескінченність,<br>-нескінченність <y< +нескінченність |
| FYL2XP1 | Обрахування $y*\log_2(c+1)$ | C ===> ST(0); y ===> ST(1);<br>$0 < \|c\| < 1-2^{0,5}/2$,<br>-нескінченність <y< +нескінченність |

*1.5 Команди порівняння*

Команди цієї групи виконують порівняння вмісту регістра ST(0) з джерелом і виставляють відповідні прапорці співпроцесора або процесора.

При порівнянні дійсних даних джерело може знаходитися або в регістрі ST(i), i=1,…,7, або в оперативній пам'яті. Якщо джерело в форматі команди не вказане, припускається, що воно зберігається в ST(1).

При порівнянні цілочисельних даних джерело може знаходитися тільки в оперативній пам'яті.

Таблиця 8.12

**Основні команди порівняння**

| Команда | Призначення |
|---|---|
| FCOM джерело | Порівняння дійсних даних |
| FCOMP джерело | Порівняння дійсних даних і звільнення вершини стека ST(0) |
| FCOMPP | Порівняння дійсних даних і звільнення вершини стека ST(0) і регістра ST(1) |
| FUCOM джерело (i80387) | Порівняння дійсних даних без врахуваня порядків |
| FUCOMP джерело (i80387) | Порівняння дійсних даних без врахуваня порядків і звільнення вершини стека ST(0) |
| FUCOMPP (i80387) | Порівняння дійсних даних без врахуваня порядків і звільнення вершини стека ST(0) і регістра ST(1) |
| FICOM джерело | Порівняння цілочисельних даних |
| FICOMP джерело | Порівняння цілочисельних даних і звільнення вершини стека ST(0) |
| FCOMI ST(0), ST(i) | Порівняння даних і встановлення EFLAGS |
| FCOMIP ST(0), ST(i) | Порівняння даних, встановлення EFLAGS і звільнення вершини стека ST(0) |

| | |
|---|---|
| FUCOMI ST(0), ST(i) | Порівняння даних бех врахування порядків і встановлення EFLAGS |
| FUCOMIP ST(0), ST(i) | Порівняння даних бех врахування порядків, встановлення EFLAGS і звільнення вершини стека ST(0) |
| FTST | Порівняння вмісту ST(0) з нулем |
| FXAM | Аналіз вмісту і визначення типу числа в ST(0) |

### *1.6 Команди управління співпроцесором*

В командах цієї групи неявних операндів нема. Приймач або джерело відповідають ділянці в оперативній пам'яті необхідної довжини.

Таблиця 8.13

**Команди управління співпроцесором**

| Команда | Призначення |
|---|---|
| FLDCW джерело | Завантаження регістрів управління CWR з джерела |
| FSTCW приймач | Запис вмісту слова управління CWR в приймач. Дана команда повністю еквівалентна команді WAIT FNSTCW |
| FNTCW приймач | Запис вмісту слова управління CWR в приймач без очікування закінчення обробки виключних ситуацій |
| FSTSW приймач (i80287) | Запис вмісту слова стану SWR в приймач. Дана команда повністю еквівалентна команді WAIT FNSTSW |
| FNSTSW приймач (i80287) | Запис вмісту слова стану SWR в приймач без очікування закінчення обробки виключних ситуацій |
| FSAVE приймач | Збереження повного стану FPU в приймачі. Дана команда повністю еквівалентна команді WAIT FNSAVE |

| | |
|---|---|
| FNSAVE приймач | Збереження повного стану FPU без очікування обробки виключних ситуацій |
| FXSAVE приймач | Швидке збереження повного стану FPU в приймачі. Дана команда не сумісна з командами FSAVE/FRSTOR |
| FRSTOR джерело | Відновлення повного стану FPU. Дана команда є зворотною до команди FSAVE |
| FXSTOR джерело | Швидке відновлення повного стану FPU. Дана команда є зворотною до команди FXSAVE |
| FLDENV джерело | Завантаження з джерела 5 допоміжних регістрів оточення співпроцесора (CWR, SWR, TWR, FIP, FDP) |
| FSTENV приймач | Збереження 5-ти допоміжних регістрів. Ця команда повністю еквівалентна команді WAIT FNSTENV і є оберненою до команди FLDENV |

Продовження табл.8.13

**Команди управління співпроцесором**

| | |
|---|---|
| FNSTENV приймач | Збереження 5-ти допоміжних регістрів без очікування закінчення обробки виключних ситуацій. Дана команда є оберненою до команди FLDENV |
| FWAIT<br>WAIT | Очікування готовності співпроцесора |
| FINIT | Ініціалізація співпроцесора. Дана команда повністю еквівалентна команді WAIT FNINIT |
| FNINIT | Ініціалізація співпроцесора без очікування обробки виключних ситуацій |
| FENI (тільки в i8087) | Включення виключень. В старших версіях співпроцесора сприймається як команда FNOP. |

| | |
|---|---|
| FDISI (тільки в i8087) | Заборона виключень. В старших версіях співпроцесора сприймається як команда FNOP. |
| FNOP | Відсутність операції |
| FCLEX | Обнулення прапорців виключень в регістрі стану SWR (PE, UE, OE, ZE, DE, IE, ES, SE, B). Дана команда повністю еквівалентна команді WAIT FNCLEX |
| FNCLEX | Обнулення прапорців виключень без очікування |
| FINCSTP | Поле TOP регістра стану співпроцесора збільшується на 1 |
| FDECSTP | Поле TOP регістра стану співпроцесора зменшується на 1 |
| FFREE ST(i) | Звільнення регістра даних ST(i), i=0,…,7. В регістрі тегів TWR даних регістр позначається як пустий. |

### *Основні особливості програмування*

До основних особливостей програмування співпроцесора, пов’язаних з операндами, належать:

1) При анонімних зверненнях до пам’яті бажано задавати тип операнда через директиву **PTR**. Наприклад: FLD QWORD PTR[BX].

2) Консанти в операндах не допускаються. Їх портрібно задавати.

3) Наявність або відсутність операнда пов’язана не тільки з самою командою, але і з

допустимими форматами команд співпроцесора. Наприклад, для арифметичних команд можливі декілька форматів.

Таблиця 8.14

**Формати команд базової арифметики**

| **Формат** | **Мнемокод** | **Допустимі операнди** | **Приклад** |
|---|---|---|---|
| Стековий | $F_{op}$ | ST(1), ST(0) | FSUB |
| Регістровий | $F_{op}$ | ST(i), ST(0)<br>ST(0), ST(i),<br>i=0,…,7 | FMUL ST(5), ST<br>FADD ST,ST(2) |
| Регістровий з витягуванням із стека | $F_{op}P$ | ST(i), ST(0),<br>i=0,…,7 | FMULP ST(3),<br>ST(0) |
| Дійсні дані в пам'яті | $F_{op}$ | ST(0), $mem_{32}$<br>ST(0), $mem_{64}$ | FADD a |
| Цілі дані в пам'яті | $FI_{op}$ | ST(0), $mem_{16}$<br>ST(0), $mem_{32}$<br>ST(0), $mem_{64}$ | FIDIV k |

*Приклад:* Написати програму для обчислення заданого змішаного арифметичного виразу для даних у форматі float, використовуючи команди співпроцесора. Виконати покрокове виконання асемблерного коду та навести значення регістрів при їх виконанні. Відмітити нормальні та аномальні результати, зробити аналіз результатів. Вираз:

$$x = \frac{\frac{c}{b} + \mathrm{a} - \sqrt{24 + \mathrm{d}}}{2ac - 1}$$

# Лістинг програми:

```
#include <iostream>
#include <stdio.h>
#include <math.h>
#include <conio.h>


int main()
{
	float a, b, c, d, res_c, res_asm, z = 24, x = 2, k = 1, n = 1;
	do
	{
		printf("Enter the values:\n");
printf("a = "); scanf_s("%f", &a);
printf("b = "); scanf_s("%f", &b);
printf("c = "); scanf_s("%f", &c);
printf("d = "); scanf_s("%f", &d);
	} while (a > 2147483647.0 || a < -2147483648.0 || b > 2147483647.0
|| b < -2147483648.0 || c > 2147483647.0 || c < -2147483648.0 || d >
2147483647.0 || d < -2147483648.0);


	if (b == 0)
	{
		printf("Error! Division by 0.\n");

	}


	else
	{
		res_c = (c / b + a - sqrt(24 + d)) / (2 * a * c - 1);

		printf("Result_C = %f\n", res_c);

	}
	__asm
	{
		finit  // ініціалізація співпроцесора

		fld c  // завантаження в вершину стека

		fdiv b // ділення (c/b)

		fld d  // завантаження в вершину стека

		 fadd z // додавання (24+d)
fsqrt  // квадратний корінь sqrt(24+d)		fsub   //
віднімання c/b-sqrt(24+d)		fadd a // додавання
```

```
c/b-sqrt(24+d)+a                fld a  // завантаження в
вершину стека           fmul x // множення (2*a)
fmul c // множення (2*a*c)              fsub k // віднімання
(2*a*c-1)               fdiv   // ділення (c/b+a-
sqrt(24+d))/(2*a*c-1)
                fstp res_asm // res_asm=(c/b+a-sqrt(24+d))/(2*a*c-1)
        }
        printf("Result_ASM = %f\n", res_asm);
        _getch();
return 0;
}
```

Проведемо тестову перевірку роботи програми(рис.8.1).

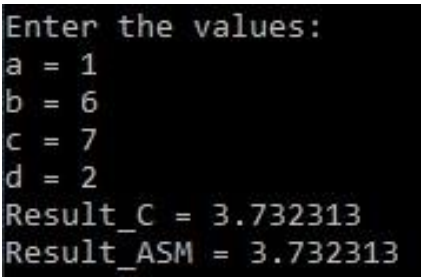


Рисунок 7.1 – Перевірка роботи програми

В даному виразі може виникнути помилка ділення на 0(рис.7.2):

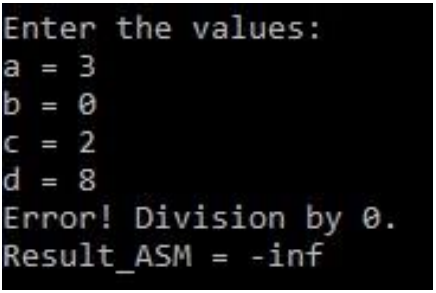


Рисунок 7.2 – Помилка при діленні на 0

## Значення регістрів співпроцесора при пороковому виконанні

| Команда | Опис | Стан регістрів | Коментар |
|---|---|---|---|
| finit | Ініціалізація співпроцесора | cwr=037Fh;<br>swr=0;<br>twr=FFFFh;<br>dpr=0; ipr=0; | |
| fld c | Завантаження в вершину стека | src | c |
| fdiv b | Ділення | dst=dst/src | c/b |
| fld d | Завантаження в вершину стека | src | d |
| fadd z | Додавання | dst=dst+src | d+24 |
| fsqrt | Квадратний корінь | St(0)=$\sqrt{St(0)}$ | sqrt(24+d) |
| fsub | Віднімання | dst=dst-src | c/b-sqrt(24+d) |
| fadd a | Додавання | dst=dst+src | c/b-sqrt(24+d)+a |
| fld a | Завантаження в вершину стека | src | a |
| fmul x | Множення | dst=dst*src | 2*a |
| fmul c | Множення | dst=dst*src | 2*a*c |
| fsub k | Віднімання | dst=dst-src | 2*a*c-1 |
| fdiv | Ділення | dst=dst/src | (c/b-sqrt(24+d)+a)/(2*a*c-1) |
| fstp res_asm | Збереження у вершині стеку | dst= St(0) | |

## Завдання на лабораторну роботу

1. Написати програму для ообчислення заданих змішаних арифметичних виразів(табл.8.15) для даних у форматі float, використовуючи команди співпроцесора. Виконати покрокове виконання асемблерного коду та навести стан регістрів при їх

виконанні. Відмітити нормальні та аномальні результати, зробити аналіз результатів.

Таблиця 8.15

**Дані для розрахунку**

| № Варіанту | Вираз 1 | Вираз 2 |
|---|---|---|
| 1 | $\frac{2*c-d+\sqrt{23*a}}{\frac{a}{4}-1}$ | $\frac{2*b-38*c}{\frac{arctg(b+a)}{c}+1}$ |
| 2 | $\frac{c+4*d-\sqrt{123*c}}{1-\frac{a}{2}}$ | $\frac{a+tg\left(\frac{b}{4}-1\right)}{\frac{c}{3}-a*d}$ |
| 3 | $\frac{-2*c+d*82}{tg\left(\frac{q}{4}-1\right)}$ | $\frac{\lg\left(25+2*\frac{a}{d}\right)}{c+a-b-1}$ |
| 4 | $\frac{\lg(2*c-a)+d-152}{\frac{a}{4}+c}$ | $\frac{\frac{b}{2}-\frac{53}{c}}{arctg(b-a)*c+1}$ |
| 5 | $\frac{tg\left(a+\frac{c}{4}\right)-12*d}{a*b-1}$ | $\frac{4*\ln\left(\frac{a}{b}\right)+1}{c+b-d+a}$ |

| | | |
|---|---|---|
| 6 | $\frac{-2*c-\sin\left(\frac{a}{d}\right)+53}{\frac{a}{4}-b}$ | $\frac{\frac{4}{c}+tg(3*a)}{\frac{c}{a}-b-1}$ |
| 7 | $\frac{2*c-\lg\left(\frac{d}{4}\right)}{a*a-1}$ | $\frac{c+23-b+4}{a-\ln\left(a+\frac{c}{b}-1\right)}$ |
| 8 | $\frac{tg(c)-d*23}{2*a-1}$ | $\frac{arctg\left(\frac{c}{4}+28\right)*d}{\frac{a}{d}-c-1}$ |
| 9 | $\frac{2*c-\frac{d}{23}}{\ln\left(1-\frac{a}{4}\right)}$ | $\frac{\sqrt{2*b}-a+\frac{b}{c}}{a-\frac{c}{4}+1}$ |
| 10 | $\frac{4*c+d-1}{c-tg\frac{a}{2}}$ | $\frac{8*\lg(b-1)-c}{a*2+\frac{b}{c}}$ |
| 11 | $\frac{2*c-d*\sqrt{\frac{42}{d}}}{c+a-1}$ | $\frac{\frac{arctg(b-c)}{b}+\frac{a}{4}}{a+b-1}$ |
| 12 | $\frac{\sqrt{\frac{25}{c}-d+2}}{d+a-1}$ | $\frac{c*b-24+a}{b-\lg(2*c-1)+a}$ |
| 13 | $\frac{arctg\left(c-\frac{d}{2}\right)}{2*a-1}$ | $\frac{\frac{4*\ln(b)}{c}+1}{2*c+a-b}$ |

| | | |
|---|---|---|
| 14 | $\frac{4*\lg(c)-\frac{d}{2}+23}{a^2-1}$ | $\frac{\sqrt{\frac{41*d}{a}+1}}{a-\frac{c}{b}+d}$ |
| 15 | $\frac{c*tg(b+23)}{\frac{a}{2}-4*d-1}$ | $\frac{\ln(a*b+2)*c}{41-\frac{b}{c}+1}$ |
| 16 | $\frac{\frac{c}{d}+\ln\left(3*\frac{a}{2}\right)}{c-a+1}$ | $\frac{arctg(a-c)*b+28}{4*\frac{b}{a}+1}$ |
| 17 | $\frac{2*c+\lg(d)*51}{d-a-1}$ | $\frac{b*a+\frac{c}{2}}{c-tg(b+1)}$ |
| 18 | $\frac{2*c+\ln\left(\frac{d}{4}\right)+23}{a^2-1}$ | $\frac{2*c+tg(a-21)}{\frac{c}{a}+d+1}$ |
| 19 | $\frac{42*c-\frac{d}{2}+1}{a^2-\ln(b-5)}$ | $\frac{\lg(4*b-c)*a}{b+\frac{c}{d}-1}$ |
| 20 | $\frac{\frac{arctg(2*c)}{d}+2}{d-a-1}$ | $\frac{1+a-\frac{b}{2}}{\sqrt{24+d-c}+\frac{a}{b}}$ |
| 21 | $\frac{arctg\left(\frac{12}{c}\right)+73}{a^2-1}$ | $\frac{a*\frac{b}{4}-1}{\sqrt{41-d}-b*a+c}$ |
| 22 | $\frac{2*\frac{c}{a}-d^2}{d+tg(a-1)}$ | $\frac{c-\frac{\ln(33+b)}{4}}{a-\frac{c}{b}-1}$ |

| | | |
|---|---|---|
| 23 | $\frac{\sqrt{\frac{53}{a}}+d-4*a}{1+a*c}$ | $\frac{\lg(21-a)*\frac{c}{4}}{1+\frac{c}{a}+b}$ |
| 24 | $\frac{\sqrt{15*a}+b-\frac{a}{4}}{c*d-1}$ | $\frac{2*b-\ln(a+b)*c}{\frac{c}{4}-1}$ |
| 25 | $\frac{-\frac{25}{a}+c-tg(b)}{1+c*\frac{d}{2}}$ | $\frac{\lg\left(\frac{b}{a}+4\right)+d}{c-b+1}$ |
| 26 | $\frac{\lg(4*a-1)+\frac{b}{2}}{d*c-5}$ | $\frac{a-b*4-1}{\frac{c}{31}+tg(a*b)}$ |
| 27 | $\frac{8*\lg(b+1)-c}{\frac{a}{2}+d*c}$ | $\frac{c*4+28*\frac{d}{c}}{5-arctg(a*d-c-1)}$ |
| 28 | $\frac{4*a-\ln(b-1)}{\frac{c}{d}+18}$ | $\frac{41-\frac{\frac{d}{4}-1}{c}}{tg(b+a)}-d$ |
| 29 | $\frac{\frac{arctg(4*b)}{c}-1}{12+a-b}$ | $\frac{\frac{c}{b}-\sqrt{24+d}+a}{2*a*c-1}$ |
| 30 | $\frac{arctg(b)+c*b-\frac{a}{4}}{a*d-1}$ | $\frac{a+\frac{c}{b}-\sqrt{28*d}}{4*b*a+1}$ |

## Контрольні питання

1. Що таке математичний співпроцесор?

2. Типи звичайних даних співпроцесора.
3. Спеціальних формати даних співпроцесора.
4. Назвіть регістри співпроцесора.
5. Звичайні і реверсні операції.
6. Команди пересилки даних.
7. Команди завантаження констант.
8. Арифметичні команди
9. Трансценденті операції
10. Команди порівняння та управління співпроцесором