ĐẠI HỌC QUỐC GIA TP HỒ CHÍ MINH
TRƯỜNG ĐẠI HỌC BÁCH KHOA
KHOA KHOA HỌC VÀ KỸ THUẬT MÁY TÍNH

----------oOo----------

BÁO CÁO ASSIGNMENT
KIẾN TRÚC TẬP LỆNH MIPS

KIẾN TRÚC MÁY TÍNH

GVHD: THẦY TRẦN THANH BÌNH

Problem 2: Cộng, trừ 2 số thực	
Sinh viên thực hiện	**Mã số sinh viên**
Trần Phương Tĩnh	1927038
Lê Tất Thiện	1920058

TP. HỒ CHÍ MINH, tháng 12 năm 2020

Mục lục

1. Giới thiệu đề tài

- Cộng, trừ 2 số thực.
- Cho 2 số thực dạng chuẩn (Standard Floating Point IEEE 754) A và B với độ chính xác đơn (32 bit). Sử dụng hợp ngữ assembly MIPS, viết thủ tục cộng (trừ) hai số A, B.
- Giả sử tập lệnh hợp ngữ MIPS không hỗ trợ phép tính dấu chấm di động.
- Yêu cầu:
  - Thống kê số lệnh, loại lệnh của chương trình của nhóm
  - Tính và trình bày cách tính thời gian chạy của chương trình trên máy tính MIPS có tần số 2GHz
  - Code:
    - ✧ Code style phải rõ ràng, có comment, phân hoạch công việc theo từng hàm.
    - ✧ Truyền nhận và trả kết quả gọi hàm theo quy ước sử dụng thanh ghi ($a0,$a1, $a2, $a3 cho argument; $v0, $v1 cho kết quả trả về).
    - ✧ Xuất kết quả để kiểm tra (sử dụng các hàm hệ thống).

2. Giải thuật của chương trình

Gọi 2 số nhập vào là A và B.

2.1. Tính tổng

Gọi tổng là T.
Gọi sign là S, exponent là E, fraction là F.

Công thức số thực: $(-1)^S \times (1+F) \times 2^{E-bias}$ với bias = 127.
S gồm 1 bit, E gồm 8 bits, F gồm 23 bits.
Trong code hiện thực sẽ có 3 thanh ghi 32-bit để lưu giá trị S, E, F.
Fraction trong code hiện thực có cộng thêm 0x00800000, tức là làm cho bit thứ 23 (tính từ bit thứ 0 từ phải sang) từ 0 thành 1. Mục đích là để biểu diễn số 1 trong (1 + F), sẽ thuận lợi trong việc tính toán.

Kiểm tra trước những ngoại lệ:

Nếu hai số là -0 thì trả về -0 và kết thúc chương trình.
Nếu hai số là 0 thì trả về 0 và kết thúc chương trình.
Nếu một trong hai số là 0 hoặc -0 thì trả về kết quả là số còn lại và kết thúc chương trình.
Nếu hai số là Infinity và trái dấu thì trả về NaN và kết thúc chương trình.
Nếu một trong hai số là Infinity thì trả về Infinity và kết thúc chương trình.
Nếu một trong hai số là -Infinity thì trả về -Infinity và kết thúc chương trình.
Nếu không, tức là hai số đều khác 0 và khác Infinity thì vào bước 1.

- **Bước 1: Đưa hai số về dạng có exponent bằng nhau**
  - Nếu $E_A = E_B$ thì sang bước 2.
  - Nếu $E_A < E_B$ thì dịch phải F_A cho đến khi $E_A = E_B$ thì sang bước 2 (cứ dịch phải F_A một bit thì E_A sẽ cộng thêm một).
  - Nếu $E_A > E_B$ thì dịch phải F_B cho đến khi $E_A = E_B$ thì sang bước 2 (cứ dịch phải F_B một bit thì E_B sẽ cộng thêm một).

Kết thúc bước 1 ta có $E_T = E_A = E_B$.

- **Bước 2: Thực hiện phép cộng hai fraction**
  - **Bước 2.1:**
    - Nếu A và B cùng dấu thì $S_T = S_A = S_B$ và $F_T = F_A + F_B$ và sang bước 3.
    - Nếu A dương và B âm thì $F_T = F_A - F_B$ và sang bước 2.2.
    - Nếu A âm và B dương thì $F_T = F_B - F_A$ và sang bước 2.2.
  - **Bước 2.2:**
    - Nếu $F_T = 0$ thì trả về 0 (hai số nhập vào đối dấu và có giá trị tuyệt đối bằng nhau) và kết thúc chương trình.
    - Nếu $F_T > 0$ thì $S_T = 0$ (tổng dương) và sang bước 3.
    - Nếu $F_T < 0$ thì $S_T = 1$ (tổng âm), đảo dấu F_T về lại dương (tức là lấy bù 2 của F_T) và sang bước 3.

- **Bước 3: Chuẩn hoá tổng và kiểm tra overflow, underflow:**
  - **Bước 3.1:**
    - Nếu $F_T > 0\text{x}00\text{ffffff}$ thì dịch phải F_T cho đến khi $F_T \le 0\text{x}00\text{ffffff}$ thì sang bước 3.2 (cứ dịch phải F_T một bit thì E_T sẽ cộng thêm một).

- Nếu $F_T < 0x00800000$ thì dịch trái F_T cho đến khi $F_T \geq 0x00800000$ thì sang bước 3.2 (cứ dịch trái F_T một bit thì E_T sẽ trừ đi một).
- Nếu không, tức là $0x00800000 \leq F_T \leq 0x00ffffff$ thì sang bước 3.2.

- **Bước 3.2:**
  - Nếu $E_T < 0$ thì trả về NaN và kết thúc chương trình.
  - Nếu $E_T > 254$ thì trả về Infinity (dấu của Infinity là dấu của số có exponent lớn hơn) và kết thúc chương trình.
  - Nếu không, tức là $0 \leq E_T \leq 254$ thì sang bước 4.

- **Bước 4: Tổng hợp S_T, E_T, F_T lại thành két quả cuối cùng**
  - 1 bit đầu là S_T.
  - 8 bits tiếp theo là E_T.
  - 23 bits cuối cùng là F_T.

Kết thúc bước 4 ta có tổng T cần tính.

2.2. Tính hiệu:

Ta chuyển việc tính hiệu về tính tổng bằng cách lấy số thứ nhất cộng cho số đối của số thứ hai:

$$A - B = A + (-B)$$

3. Thống kê số lệnh, loại lệnh, thời gian chạy của chương trình

- Công thức tính thời gian chạy của chương trình:

$$CPUTime = \frac{InstructionCount \times CPI}{ClockRate}$$

- Trong đó:
  - CPU Time: Thời gian thực thi của chương trình.
  - Instruction Count: Tổng số lệnh thực thi của chương trình.
  - CPI: Số chu kỳ trung bình trên mỗi lệnh thực thi (CPI = 1).
  - Clock Rate: Tần số của máy tính (2 Ghz).
- Ví dụ: A = B = 0.0 thì IC = 120 nên

$$CPUTime = \frac{120 \times 1}{2 \times 10^9} = 60 \times 10^{-9}(s) = 60(ns)$$

- Test case:
  - 40 test case cho các trường hợp: 0, -0, số dương, số âm, Infinity, -Infinity.
  - 4 testcase cho số nguyên và số không biểu diễn chính xác.

STT	A	B	+	-	ALU	Jump	Branch	Memory	Other	IC	CPU Time (ns)
01	0.0	0.0	0.0	0.0	62	13	14	18	13	120	60
02	0.0	-0.0	0.0	0.0	62	13	14	18	13	120	60
03	0.0	0.125	0.125	-0.125	61	12	14	18	13	118	59
04	0.0	-0.125	-0.125	0.125	61	12	14	18	13	118	59
05	0.0	1e40	Inf	-Inf	61	12	14	18	13	118	59
06	0.0	-1e40	-Inf	Inf	61	12	14	18	13	118	59
07	-0.0	0.0	0.0	-0.0	62	11	16	18	13	120	60
08	-0.0	-0.0	-0.0	0.0	62	11	16	18	13	120	60
09	-0.0	1.25	1.25	-1.25	63	12	18	18	13	124	62
10	-0.0	-1.25	-1.25	1.25	63	12	18	18	13	124	62
11	-0.0	2e45	Inf	-Inf	63	12	18	18	13	124	62
12	-0.0	-2e45	-Inf	Inf	63	12	18	18	13	124	62
13	2.5	0.0	2.5	2.5	62	13	19	18	13	125	62.5
14	2.5	-0.0	2.5	2.5	62	13	19	18	13	125	62.5
15	2.5	3.75	6.25	-1.25	133	16	38	18	20	225	112.5
16	2.5	-3.75	-1.25	6.25	133	16	38	18	20	225	112.5
17	2.5	2.5	5.0	0.0	113	16	32	18	16	195	97.5
18	2.5	-2.5	0.0	5.0	113	16	32	18	16	195	97.5
19	2.5	3e50	Inf	-Inf	91	11	24	18	13	157	78.5
20	2.5	-3e50	-Inf	Inf	91	11	24	18	13	157	78.5
21	-4.375	0.0	-4.375	-4.375	62	13	19	18	13	125	62.5
22	-4.375	-0.0	-4.375	-4.375	62	13	19	18	13	125	62.5
23	-4.375	5.625	1.25	-10.0	133	16	39	18	21	227	113.5
24	-4.375	-5.625	-10.0	1.25	133	16	39	18	21	227	113.5
25	-4.375	-4.375	-8.75	0.0	113	15	32	18	16	194	97
26	-4.375	4.375	0.0	-8.75	113	15	32	18	16	194	97
27	-4.375	4e55	Inf	-Inf	91	11	24	18	13	157	78.5
28	-4.375	-4e55	-Inf	Inf	91	11	24	18	13	157	78.5
29	5e60	0.0	Inf	Inf	62	13	19	18	13	125	62.5
30	5e60	-0.0	Inf	Inf	62	13	19	18	13	125	62.5
31	5e60	6.6	Inf	Inf	91	14	20	18	13	156	78
32	5e60	-7.7	Inf	Inf	91	14	20	18	13	156	78
33	5e60	6e65	Inf	NaN	91	12	17	18	12	150	75
34	5e60	-6e65	NaN	Inf	91	12	17	18	12	150	75
35	-7e70	0.0	-Inf	-Inf	62	13	19	18	13	125	62.5
36	-7e70	-0.0	-Inf	-Inf	62	13	19	18	13	125	62.5
37	-7e70	8.1	-Inf	-Inf	91	16	24	18	13	162	81
38	-7e70	-9.2	-Inf	-Inf	91	16	24	18	13	162	81
39	-7e70	8e75	NaN	-Inf	91	13	20	18	12	154	77
40	-7e70	-8e75	-Inf	NaN	91	13	20	18	12	154	77
41	1	2	3.0	-1.0	136	16	42	18	22	234	117
42	-3	-4	-7.0	1.0	136	16	43	18	23	236	118
43	5.1	6.2	11.299999	-1.0999999	135	16	39	18	21	229	114.5
44	-7.3	-8.5	-15.8	1.1999998	138	16	44	18	24	240	120

4. Tài liệu tham khảo

- Slide bải giảng trên BKel.
- Sách Computer Organization and Design, The Hardware/ Software Interface by David A. Patterson and John L. Hennessy, Fifth Edition.