ĐẠI HỌC QUỐC GIA THÀNH PHỐ HỒ CHÍ MINH

TRƯỜNG ĐẠI HỌC BÁCH KHOA

KHOA KHOA HỌC & KỸ THUẬT MÁY TÍNH

BÀI TẬP LỚN KIẾN TRÚC MÁY TÍNH

Đề 6: Chia hai số nguyên 32 bit

GVHD: Nguyễn Xuân Minh

Nhóm: 15

SV thực hiện: Dương Đức Nghĩa – 2011671

Lê Thị Hồng Thắm – 2012069

Thành phố Hồ Chí Minh, Tháng 6 / 2022

MỤC LỤC

1. Đề Bài

Viết chương trình hiện thực giải thuật chia số nguyên trong textbook (hình 3.9), áp dụng cho số có dấu. Dữ liệu đầu vào đọc từ file lưu trữ dạng nhị phân trên đĩa INT2.BIN (2 trị x 4 bytes = 8 bytes).

2. Yêu cầu

- Đề = ((Số nhóm-1) % 9) + 1
- Chương trình viết và chạy trên MARS MIPS 4.5.
- Code
  - Code style phải rõ ràng, có chú thích.
  - Phải có gọi hàm. Truyền tham số và trả kết quả khi gọi hàm theo quy ước của thanh ghi ($Ai chứa tham số, $Vi hoặc $fi chứa giá trị trả về).
  - In kết quả ra màn hình để kiểm tra.
- Nội dung báo cáo:
  - Trình bày giải pháp hiện thực.
  - Giải thuật (nếu có).
  - Thống kê số lệnh, loại lệnh (instruction type) sử dụng trong chương trình.
  - Tính thời gian chạy của chương trình (CR=1GHz).
  - Kết quả kiểm thử.
- Nộp báo cáo : 3 files
  - File báo cáo (không source code) định dạng .PDF (Bc_nhom##.pdf).
  - File mã nguồn (Mn_nhom##.asm).
  - File dữ liệu đầu vào (xxx.BIN).

3. Cơ sở lý thuyết

3.1. Số 32 bit

- Một thanh ghi 32-bit có thể lưu trữ 2^{32} giá trị khác nhau. Phạm vi có dấu của giá trị số nguyên có thể lưu trữ trong 32 bit là từ -2,147,483,648 đến 2,147,483,647 (không dấu: 0 đến 4,294,967,295)

3.2. Trừ hai số nhị phân

Quy tắc trừ hai số nhị phân

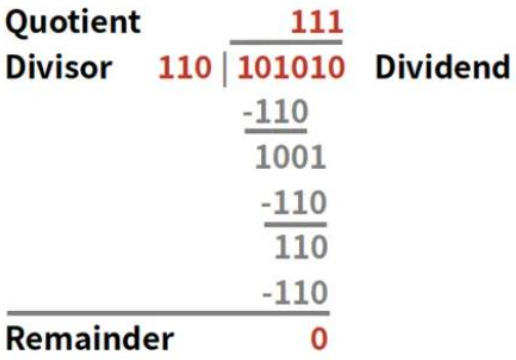
$0 - 0 = 0$

$0 - 1 = 1$ (mượn 1)

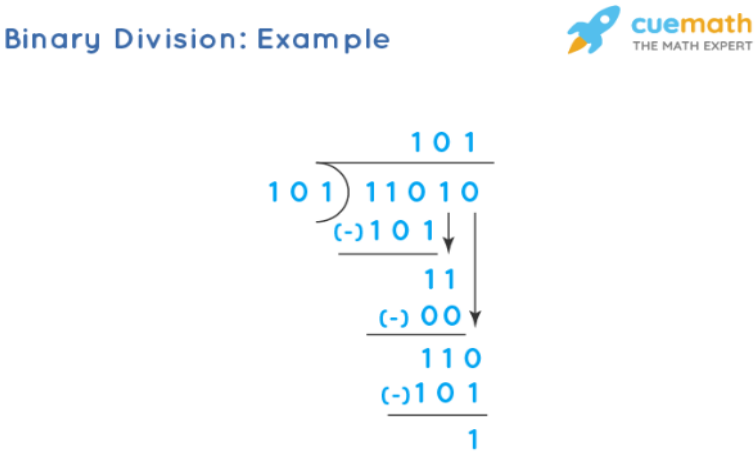
$1 - 0 = 1$

$1 - 1 = 0$

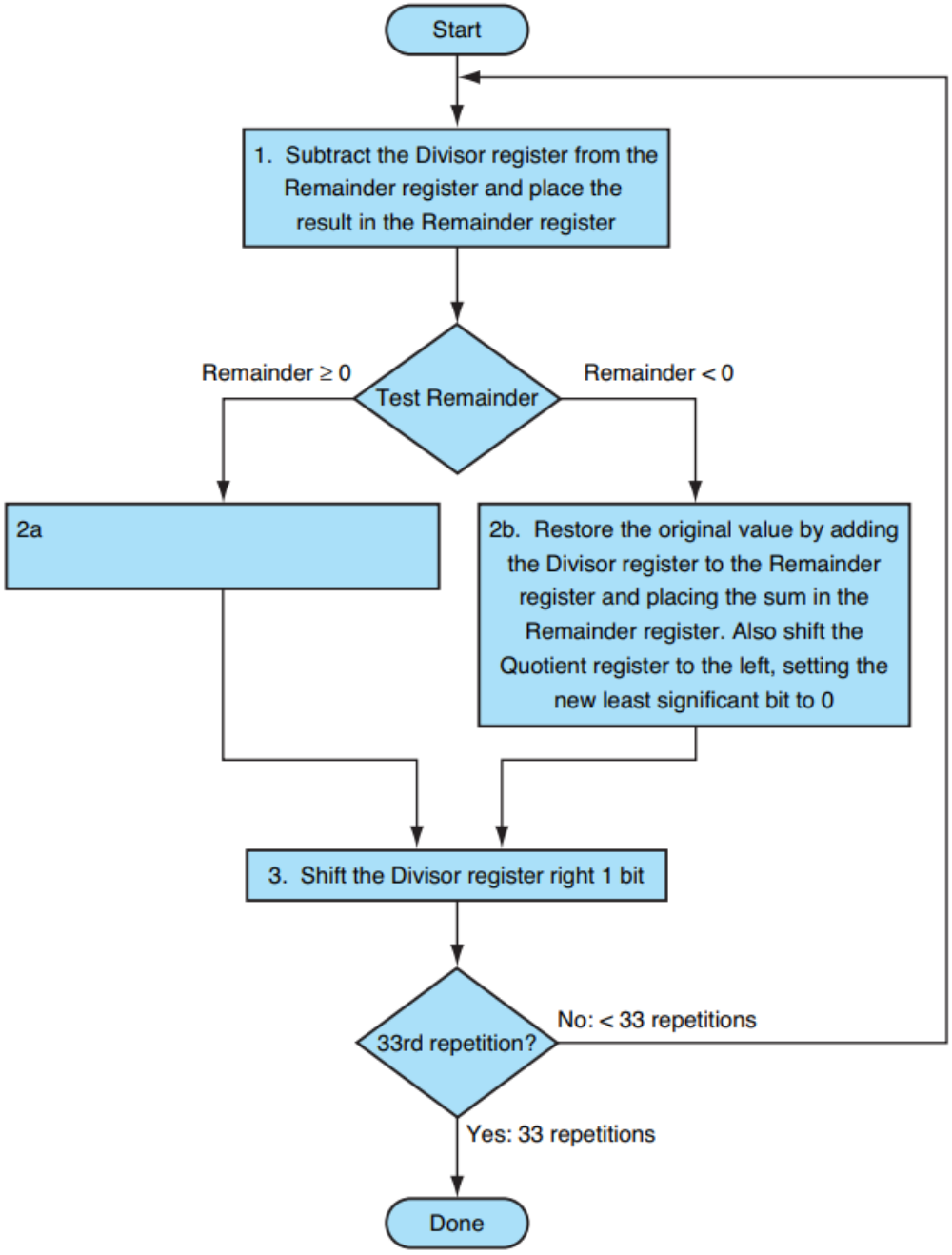
3.3. Chia hai số nhị phân



- Dividend: số bị chia
- Divisor: số chia
- Quotient: thương
- Remainder: số dư



4. Ý tưởng giải thuật



5. Các bước thực hiện

5.1. Đọc 2 số nguyên từ file .bin

Mở file (syscall 13) ➔ Đọc vào 2 số nguyên (syscall 14) ➔ đóng file (syscall 16)

5.2. Kiểm tra Divisor có bằng 0 hay không

Nếu Divisor bằng 0 lập tức in ra dòng thông báo “Chia cho 0” và kết thúc chương trình

5.3. Lấy giá trị tuyệt đối cho Dividend và Divisor

5.4. Mở rộng Dividend và Divisor

Đối với Dividend thì thêm 32 bit 0 vào đầu

Đối với Divisor thì thêm 32 bit 0 vào sau

➔ Kết quả là cả 2 số được mở rộng thành số 64 bit

5.5. Thực hiện phép trừ 64 bit

Phép trừ sẽ được chuyển thành phép cộng với số âm ở đây được chuyển về dạng 2’s complement

Sau đó thực hiện phép cộng 64 bit như sau:

Thực hiện phép cộng không tràn 2 thanh ghi thấp nếu kết quả xảy ra tràn thì nhớ 1 để cộng tiếp vào tổng của 2 thanh ghi cao

5.6. Kiểm tra nếu kết quả phép trừ

Nếu kết quả âm thì thực hiện restore lại số dư, dịch thanh ghi Quotient sang trái và gắn LSB của Quotient là 0

Nếu kết quả lớn hơn hoặc bằng 0 thì dịch thanh ghi Quotient sang trái và gắn LSB của Quotient là 1

5.7. Dịch thanh ghi Divisor sang phải

Vì thanh ghi Divisor đang ở dạng 64 bit nên trước tiên ta cần dịch phải 32 bit thấp rồi xét xem nếu LSB của 32 bit cao là 1 thì gán MSB của 32 bit thấp là 1 rồi dịch 32 bit cao

5.8. Kiểm tra vòng lặp

Nếu đã lặp đủ 33 lần thì dừng lặp và chuyển tiếp đến việc xét dấu

5.9. Xét dấu cho kết quả

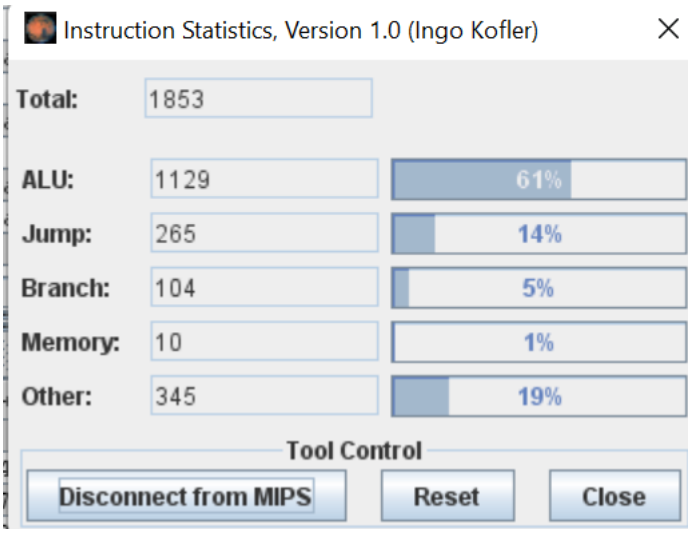
Ta thực hiện theo quy luật sau:

- Nếu Dividend và Divisor trái dấu thì đảo dấu của Quotient
- Dấu của Remainder trùng với dấu của Dividend

5.10. In kết quả và kết thúc chương trình

6. Thống kê số lệnh

Để thống kê số lệnh, loại lệnh (instruction type) trong Mips, ta chọn *Tools -> Instruction Statistics -> Connect to MIPS*. Sau đó chạy chương trình.



- Vậy trong chương trình, đã sử dụng tổng cộng 1853 lệnh, trong đó:
  - Lệnh ALU: 1129 lệnh
  - Lệnh Jump: 265 lệnh
  - Lệnh Branch: 104 lệnh

• Lệnh Memory: 10 lệnh

• Các loại lệnh khác: 345 lệnh

Nhận xét :

- Tổng số lệnh khá cao nguyên nhân do trong chương trình đã thực hiện 33 lần lặp giải thuật .
- Hầu hết các lệnh là lệnh ALU (chiếm 61%)
- Lệnh Memory chiếm rất ít khoảng 1% do không sử dụng nhiều câu lệnh lưu trữ và truy xuất dữ liệu

7. Thời gian chạy chương trình

- Giả sử số CPI của mỗi loại lệnh là:

• Lệnh ALU: 1

• Lệnh Jump: 3

• Lệnh Branch: 3

• Lệnh Memory: 5

• Các loại lệnh khác: 3

- Vậy CPI trung bình của chương trình là:

$$CPI = \frac{\Sigma_i (IC_i)(CC_i)}{IC} = \frac{1\times1129+3\times265+3\times104+10\times5+3\times345}{1853} = 1.79223$$

⇨ Thời gian chạy với CR = 1GHz là :

⇨ ET = $\frac{IC \times CPI}{CR} = \frac{1853\times1.79233}{10^9} = 3.3212\mu s$

8. Kết quả chạy thử

Nhóm đã cho chạy thử kết quả và sau đây là một số hình ảnh chụp từ console của MARS:

```
So nguyen thu nhat la: 7
So nguyen thu hai la: 2
Thuong so cua phep chia la: 3
So du cua phep chia la: 1
-- program is finished running --
```

Phép chia hai số dương

```
So nguyen thu nhat la: 7
So nguyen thu hai la: 0
Phep chia cho 0 !!!
-- program is finished running --
```

Phép chia cho 0

```
So nguyen thu nhat la: 2147483647
So nguyen thu hai la: 2147483647
Thuong so cua phep chia la: 1
So du cua phep chia la: 0
-- program is finished running --
```

Phép chia Max range 32 bit

```
So nguyen thu nhat la: 7
So nguyen thu hai la: -2
Thuong so cua phep chia la: -3
So du cua phep chia la: 1
-- program is finished running --
```

Phép chia một số dương cho một số âm

```
So nguyen thu nhat la: -7
So nguyen thu hai la: -2
Thuong so cua phep chia la: 3
So du cua phep chia la: -1
-- program is finished running --
```

Phép chia hai số âm

9. Kết luận

- Phép chia hai số có dấu có thể thực hiện bằng cách thực hiện phép chia hai số không dấu rồi xét dấu tương tự kết quả phép nhân.
- Thiết kế phần cứng tập trung vào hai số nguyên không dấu (số nguyên dương). Bởi vì bộ chia được thiết kế cho bộ xử lý MIPS 32 bit nên các toán hạng sẽ có kích thước 32 bit. Phép chia hai số 32 bit sẽ tạo ra kết quả là một thương số 32 bit và một số dư 32 bit.
- Chương trình có thể chạy với độ chính xác cao và tốc độ nhanh.

TÀI LIỆU THAM KHẢO

1. Patterson, D. A., & Hennessy, J. L. (2009). Computer organization and design: The hardware/software
   interface. Burlington, MA: Morgan Kaufmann Publishers.
2. 32 bit – Wikipedia : https://vi.wikipedia.org/wiki/32-bit
3. Binary Division – CueMath : https://www.cuemath.com/numbers/binary-division/