**ĐẠI HỌC QUỐC GIA THÀNH PHỐ HỒ CHÍ MINH**



**TRƯỜNG ĐẠI HỌC BÁCH KHOA**

**KHOA KHOA HỌC VÀ KỸ THUẬT MÁY TÍNH**

**BÀI TẬP LỚN**

**KIẾN TRÚC MÁY TÍNH**

**Đề 6: Chia 2 số nguyên 32 bit**

GVHD: Nguyễn Xuân Minh

Lớp: L01

Nhóm: 6

Danh sách thành viên:

Họ tên MSSV

1. Trần Đình Phong 2114404
2. Vũ Huỳnh Tấn Phát2114391

**TP Hồ Chí Minh, tháng 4 năm 2023**

🙞🙟 LỜI CẢM ƠN 🙝🙜

Lời đầu tiên, nhóm 6 chúng em xin gửi đến thầy Nguyễn Xuân Minh - giảng viên đã hướng dẫn cho nhóm hoàn thành đề tài này, một lời tri ân sâu sắc. Nhờ có những bài giảng cùng sự tâm huyết của thầy trên lớp mà tụi em nắm được cơ sở lí thuyết một cách khoa học, cụ thể nhất để giải quyết bài báo cáo trên theo đúng thời hạn đã giao.

Bên cạnh đó, nhóm chúng em cũng xin trân trọng cám ơn những ý kiến đóng góp, giúp đỡ và chỉ bảo tận tình của các thầy cô, bạn bè, đã giúp cho bài báo cáo trở nên hoàn thiện hơn.

Đồng thời, nhóm chúng em cũng xin chân thành cám ơn sự hỗ trợ, tương tác lẫn nhau của từng cá nhân trong nhóm. Qua quá trình làm việc nhóm đã dần hiểu rõ nhau hơn, đoàn kết hết trong học tập lẫn công việc.

Tuy nhiên, do chưa có nhiều kinh nghiệm làm việc bằng phần mềm Mars4\_5, cũng như những hạn chế về mặt kiến thức chắc chắn sẽ không tránh khỏi những thiếu sót. Rất mong nhận được sự nhận xét, ý kiến đóng góp từ phía thầy để bài báo cáo của chúng em được hoàn thiện hơn.

Lời cuối cùng, nhóm 6 chúng em xin gửi lời cám ơn chân thành nhất tới những người đã hỗ trợ nhóm trong quá trình làm bài báo cáo.

MỤC LỤC

[**1. Yêu cầu đề bài** 4](#_Toc132657653)

[**2. Giải pháp hiện thực** 4](#_Toc132657654)

[2.1. Giới thiệu bộ chia tuần tự 32 bit 4](#_Toc132657655)

[2.2. Giải pháp hiện thực 5](#_Toc132657656)

[**3. Thống kê cách lệnh đã sử dụng trong chương trình** 5](#_Toc132657657)

[ Nhóm lệnh R: 5](#_Toc132657658)

[ Nhóm lệnh I: 6](#_Toc132657659)

[ Nhóm lệnh J: 6](#_Toc132657660)

[**4. Kiểm tra chương trình** 7](#_Toc132657661)

[ Test case 1: 7](#_Toc132657662)

[ Test case 2: 8](#_Toc132657663)

[ Test case 3: 8](#_Toc132657664)

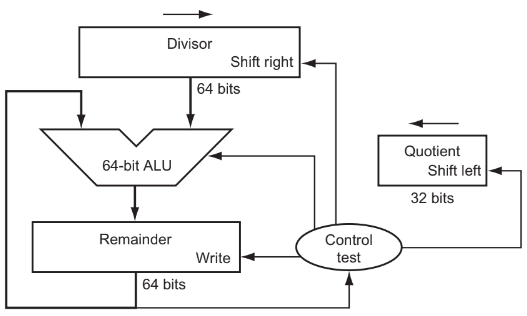
[ Test case 4: 9](#_Toc132657665)

# **1. Yêu cầu đề bài**

Chia 2 số nguyên 32 bit: 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. Giải pháp hiện thực**

## 2.1. Giới thiệu bộ chia tuần tự 32 bit



*Kiến trúc bộ chia tuần tự 32 bit*

Bộ chia tuần tự sử dụng 3 thanh ghi gồm 2 thanh ghi 64 và 1 thanh ghi 32 bit. Thanh ghi *Divisor* 64 bit sẽ chứa giá trị số chia ở 32 bit cao khi bắt đầu thực hiện phép chia. Sau mỗi bước tính toán giá trị trong thanh ghi này sẽ dịch sang phải 1 bit. Thanh ghi *Remainder* 64 bit chứa giá trị số bị chia ở 32 bit thấp khi bắt đầu thực hiện phép chia. Sau khi kết thúc phép chia thì giá trị chứa trong thanh ghi này sẽ là giá trị số dư. Thanh ghi *Quotient* 32 bit chứa giá trị thương sau khi phép chia kết thúc và tại thời điểm bắt đầu thực hiện phép chia sẽ được gán giá trị 0. Bộ ALU 64 bit sẽ thực hiện 2 công việc khác nhau là cộng và trừ trong từng trường hợp cụ thể.

## 2.2. Giải pháp hiện thực

Giải pháp hiện thực bài này được dựa trên hoạt động của bộ chia tuần tự 32 bit, thực hiện vòng lặp các bước sau 33 lần:

* Bước 1: Thực hiện phép trừ giá trị chứa trong thanh ghi *Remainder* với giá trị chứa trong thanh ghi *Divisor* và lưu kết quả vào thanh ghi *Remainder*.
* Bước 2: Kiểm tra giá trị chứa trong thanh ghi *Remainder* sau phép trừ, nếu giá trị này dương thì thực hiện bước 3, ngược lại thực hiện bước 4.
* Bước 3: Dịch trái thanh ghi *Quotient* và sửa bit thấp nhất trong thanh ghi này thành 1, chuyển sang bước 5.
* Bước 4: Cộng giá trị đang chứa trong thanh ghi *Remainder* với giá trị trong thanh ghi số *Divisor* và lưu kết quả vào thanh ghi *Remainder* đồng thời dịch trái thương 1 bit, chuyển sang bước 5.
* Bước 5: Dịch phải thanh ghi *Divisor* 1 bit.

# **3. Thống kê cách lệnh đã sử dụng trong chương trình**

* Nhóm lệnh R:

|  |  |  |
| --- | --- | --- |
| Lệnh | Ví dụ | Chức năng |
| add | add $a0, $s4, $0 | Cộng 2 thanh ghi $s4, $zero, kết quả lưu vào thanh ghi $a0. |
| sll | sll $v0, $v0, 1 | Dịch trái giá trị thanh ghi $v0 1 bit và lưu kết quả vào thanh ghi $v0. |
| slt | slt $t7, $s5, $0 | Nếu giá trị của thanh ghi $s5 nhỏ hơn $0, thì thanh ghi $t7 bằng 1, ngược lại bằng 0. |
| or | or $t2, $t2, $t7 | Thực hiện phép OR bit-wise trên giá trị của thanh ghi $t2 và thanh ghi $t7, và lưu kết quả vào thanh ghi $t2. |
| jr | jr $ra | Rẽ nhánh đến địa chỉ chứa trong thanh ghi $ra |
| srl | srl $s5, $s5, 1 | Dịch trái giá trị thanh ghi $s5 1 bit và lưu kết quả vào thanh ghi $s5. |
| addu | addu $s0, $s0, $t4 | Cộng không dấu 2 thanh ghi $t4, $s0, kết quả lưu vào thanh ghi $s0. |
| not | not $t4, $s4 | Thực hiện nghịch đảo bit-wise giá trị thanh ghi $s4 và lưu vào thanh ghi $t4. |

* Nhóm lệnh I:

|  |  |  |
| --- | --- | --- |
| Lệnh | Ví dụ | Chức năng |
| addi | addi $a2, $s2, 0 | Cộng giá trị của thanh ghi $s2 với giá trị 0 và lưu kết quả vào thanh ghi $a2. |
| lw | lw $ra, ($sp) | Lưu giá trị từ vùng nhớ được chỉ định của thanh ghi $sp vào thanh ghi $ra với offset = 0. |
| beq | beq $t2, $0, Carry0 | So sánh giá trị 2 thanh ghi $t2 và $0, nếu bằng nhau thì đi tới nhãn Carry0, ngược lại thực hiện lệnh tiếp theo. |
| addiu | addiu $t4, $t4, 1 | Cộng không dấu giá trị của thanh ghi $t4 với giá trị 1 và lưu kết quả vào thanh ghi $t4. |
| sw | sw $ra, ($sp) | Lưu giá trị của thanh ghi $ra vào địa chỉ hiện tại của con trỏ ngăn xếp $sp. |
| ori | ori $v0, $v0, 1 | Thực hiện phép toán OR giữa giá trị của thanh ghi $v0 và hằng số nguyên không dấu 1, sau đó gán kết quả vào thanh ghi $v0. |
| bne | bne $t4, $0, B4 | So sánh giá trị 2 thanh ghi $t4 và $0, nếu khác nhau thì đi tới nhãn B4, ngược lại thực hiện lệnh tiếp theo. |
| andi | andi $t1, $s4, 1 | Thực hiện phép toán AND giữa giá trị thanh ghi $s4 và hằng số nguyên không dấu 1, sau đó gán kết quả vào thanh ghi $t1. |
| bltz | bltz $v0, baoloi | Nếu giá trị $v0 nhỏ hơn 0, thực hiện nhãn baoloi, ngược lại thực hiện lệnh tiếp theo. |

* Nhóm lệnh J:

|  |  |  |
| --- | --- | --- |
| Lệnh | Ví dụ | Chức năng |
| j | j loop | Nhảy không điều kiện tới nhãn loop. |
| jal | jal Tien\_hanh | Gọi hàm được định nghĩa tại nhãn Tien\_hanh. |

# **4. Kiểm tra chương trình**

=

Trong đó:

* Clock Rate = 1 (GHz)
* CPI = 1
* Test case 1:
* Input: 1002 -53
* Output:

Text

Description automatically generated with low confidence

* Số lệnh thực hiện trong testcase:

Graphical user interface

Description automatically generated

* Thời gian thực thi testcase:

* Test case 2:
  + Input: -999 25
  + Output:

Text

Description automatically generated

* Số lệnh thực hiện trong testcase:

Graphical user interface, application

Description automatically generated

* Thời gian thực thi testcase:

* Test case 3:
* Input: 596 28
* Output:

Graphical user interface, table

Description automatically generated with medium confidence

* Số lệnh thực hiện trong testcase:

Graphical user interface, application

Description automatically generated

* Thời gian thực thi testcase:

* Test case 4:
* Input: -5579 -425
* Output:

Text

Description automatically generated with medium confidence

* Số lệnh thực hiện trong testcase:

Graphical user interface, application

Description automatically generated

* Thời gian thực thi testcase: