ĐẠI HỌC QUỐC GIA THÀNH PHỐ HỒ CHÍ MINH
TRƯỜNG ĐẠI HỌC BÁCH KHOA
Khoa Khoa học và kĩ thuật máy tính

KIẾN TRÚC MÁY TÍNH

BÀI TẬP LỚN

NHÂN 2 SỐ NGUYÊN 32 BIT

GVHD : Nguyễn Xuân Minh

Sinh viên : Phạm Hà Bảo Long – 2211893
Lê Phú Cường – 2210425

TP. HỒ CHÍ MINH, THÁNG 5/2024

MỤC LỤC

1 Đề tài hợp ngữ: Nhân 2 số nguyên 32 bit

1.1 Mô tả đề:

Viết chương trình hiện thực giải thuật nhân số nguyên trong textbook (hình 3.4 hoặc 3.5), 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).

1.2 Yêu cầu:

• Sử dụng tập lệnh MIPS để thực hiện các thủ tục bên dưới.

• Thống kê số lệnh, loại lệnh (Instruction type) của mỗi chương trình.

• Tính và trình bày cách tính thời gian chạy của chương trình trên máy tính kiến trúc MIPS có tần số 3.4GHz.

• Code:

– Code style phải rõ ràng, có comment, phân hoạch công việc theo từng hàm, CHỈ DÙNG LỆNH MIPS CHUẨN.

– Truyền và nhận kết quả khi gọi hàm theo quy ước của thanh ghi (thanh ghi $a argument, thanh ghi $v giá trị trả về khi gọi hàm).

– Xuất kết quả để kiểm tra.

• Báo cáo:

– Trình bày giải pháp hiện thực.

– Giải thuật.

– 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.

2 Giải pháp hiện thực:

- Đầu tiên, ta cần khởi tạo các thanh ghi. Thanh ghi chứa số bị nhân (Multiplicand), thanh ghi chứa số nhân (Multiplier), thanh ghi chứa kết quả của phép nhân (Product).

- Kiểm tra từng bit của số nhân (Multilier), bắt đầu từ bit thấp nhất

- Nếu kết quả bit hiện tại là 1, cộng số bị (Multiplicand) vào kết quả (Product).

- Nếu kết quả bit hiện tại là 0, giữa nguyên kết quả.

- Dịch số nhân sang phải 1 bit, dịch số bị nhân sang trái 1 bit, và thực hiện lần lặp kế tiếp.

- Lặp lại các quá trình trên cho đến khi đủ 32 lần (tương ứng với 32 bit), lưu kết quả vào Product.

3 Chi tiết giải thuật:

Để thực hiện việc nhân hai số nguyên 32 bit chúng ta có thể xây dựng một giải thuật nhân hai số nguyên 32-bit trong MIPS sử dụng các phép dịch chuyển và phép cộng.

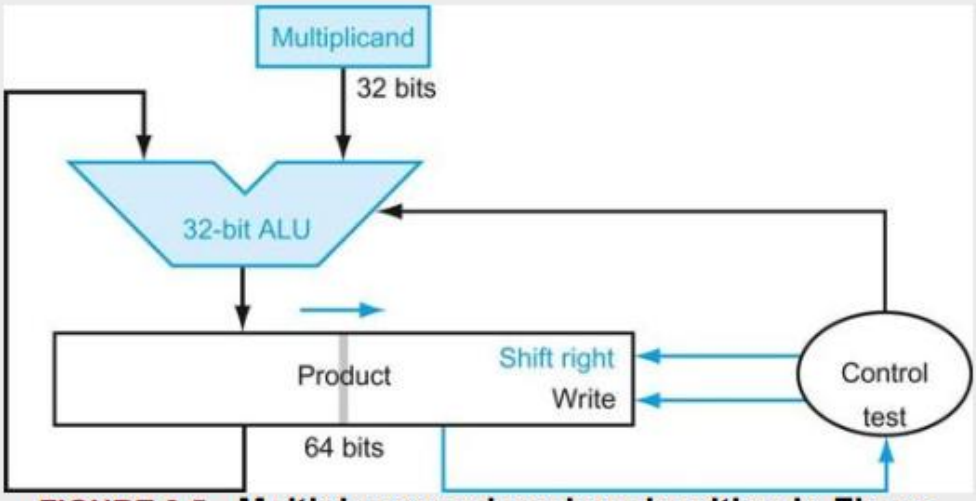


FIGURE 3.5 Multiply example using algorithm in Figure 3.4.
The bit examined to determine the next step is circled in color.

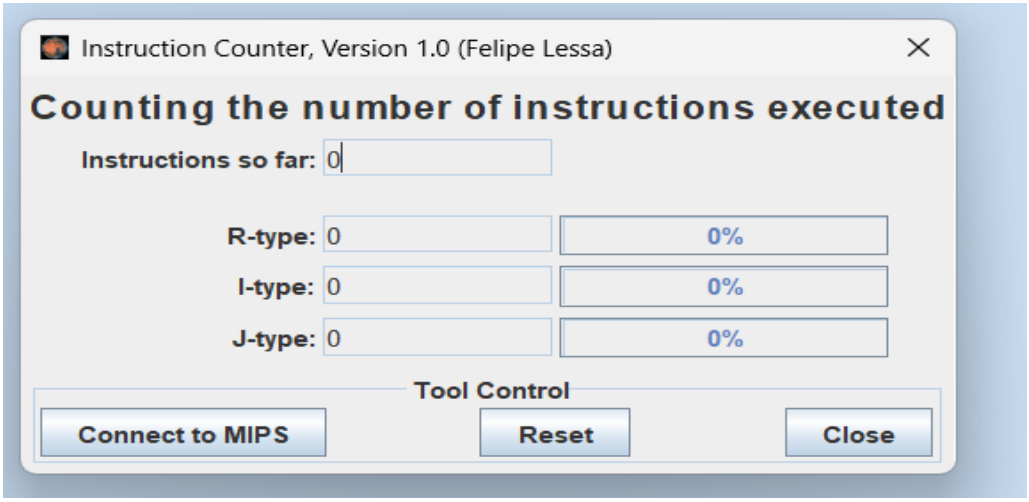
Hình 1: Giải thuật nhân hai số nguyên trong textbook.

1. Khởi tạo các thanh ghi:
    - Multiplicand (32 bit): thanh ghi chứa số bị nhân.
    - Multiplier (32 bit): thanh ghi chứa số nhân.
    - Product(64 bit): thanh ghi chứa kết quả của phép nhân, ban đầu ta đặt bằng 0.
2. Thực hiện các bước nhân:
    - Duyệt qua từng bit của số nhân (Multiplier), bắt đầu từ bit thấp nhất.
    - Kiểm tra bit hiện tại của số nhân (Multiplier).
    - Nếu bit hiện tại của số nhân (Multiplier) là 1, cộng số bị nhân (Multiplicand) vào kết quả (Product).
    - Nếu bit hiện tại của số nhân (Multiplier) là 0, giữ nguyên kết quả (Product).
    - Dịch số nhân (Multiplier) sang phải 1 bit (chia cho 2).
    - Dịch số bị nhân (Multiplicand) sang trái 1 bit (nhân với 2).

3. Lưu kết quả:
    - Lặp lại bước 2 cho đến khi đủ 32 lần (tương ứng với 32 bit), lưu kết quả vào Product.

4 Thống kê số lệnh, loại lệnh và thời gian thực thi:

Sử dụng công cụ Instruction Counter để đếm số lệnh trong chương trình:



Sử dụng công thức sau để tính thời gian chạy:

$$CPU\ time = \frac{CPU\ Clock\ cycles}{Clock\ rate} = \frac{Instruction\ Count \times CPI}{Clock\ rate}$$

Trong đó:

• CPU time là thời gian xử lý của chương trình (không tính thời gian giao tiếp I/O, thời gian chờ ...).

• CPU Clock cyles: Tổng số chu kỳ thực thi.

• Instruction Count là tổng số lệnh thực thi của chương trình.

• CPI (cycle per instruction) là số chu kỳ thực thi trên một lệnh.

• Clock rate là số chu kỳ thực thi trên một giây hay còn gọi là tần số, ví dụ: 3.4 GHz: có nghĩa trong một giây có 3.4×10^9 giao động.

5 Kết quả kiểm thử:

Tuỳ vào cách nhập hai số nguyên khác nhau để thực hiện phép toán nhân mà chương trình có tổng số lệnh, loại lệnh và thời gian thực thi khác nhau. Dưới đây là một số ví dụ cho thấy thời gian chạy khác nhau với CR = 1Ghz.

5.1 Ví dụ 1

Hai số nguyên a,b được đọc vào lần lượt là 123, 456.

Tổng số lệnh: 162

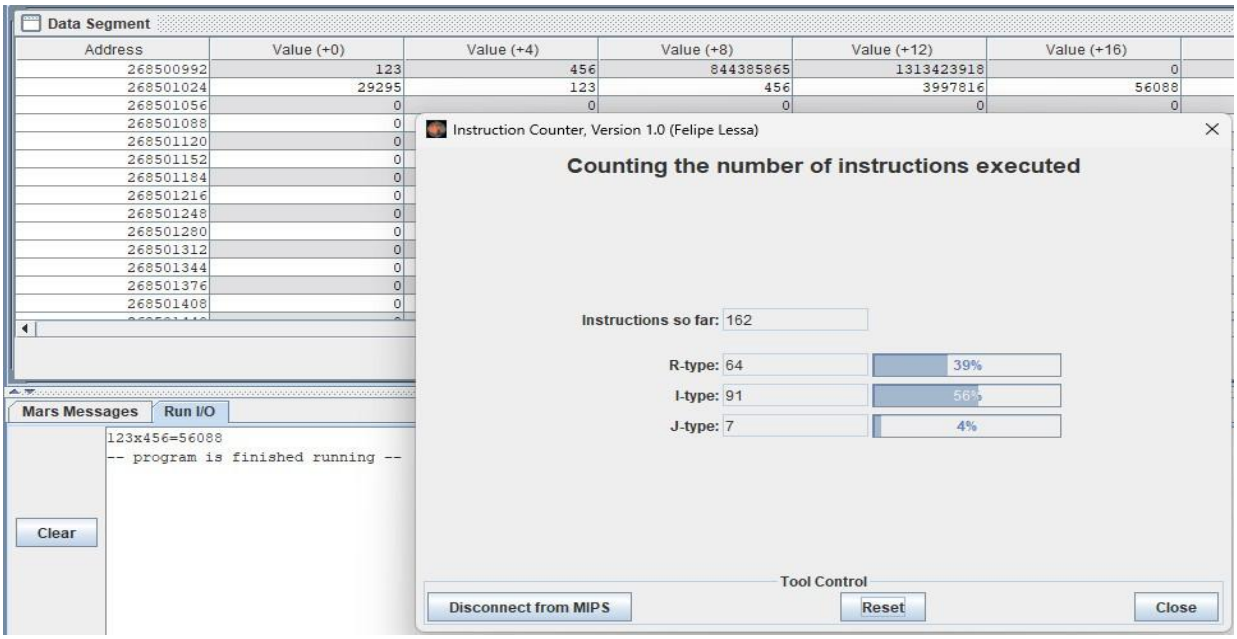
Trong đó:
- R-type: 64 lệnh chiếm 39%.

- I-type: 91 lệnh chiếm 56%.

- J-type: 7 lệnh chiếm 4%.

Tính toán thời gian thực thi:

$$CPU\ time = \frac{Instruction\ Count\ \times CPI}{Clock\ rate} = \frac{162\ \times 1}{1 \times 10^9} = 1.62 \times 10^{-7} s$$



Hình 2: Kết quả kiểm thử (1).

5.2 Ví dụ 2:

Hai số nguyên a,b được đọc vào lần lượt là -1234, 5678.

Tổng số lệnh: 203

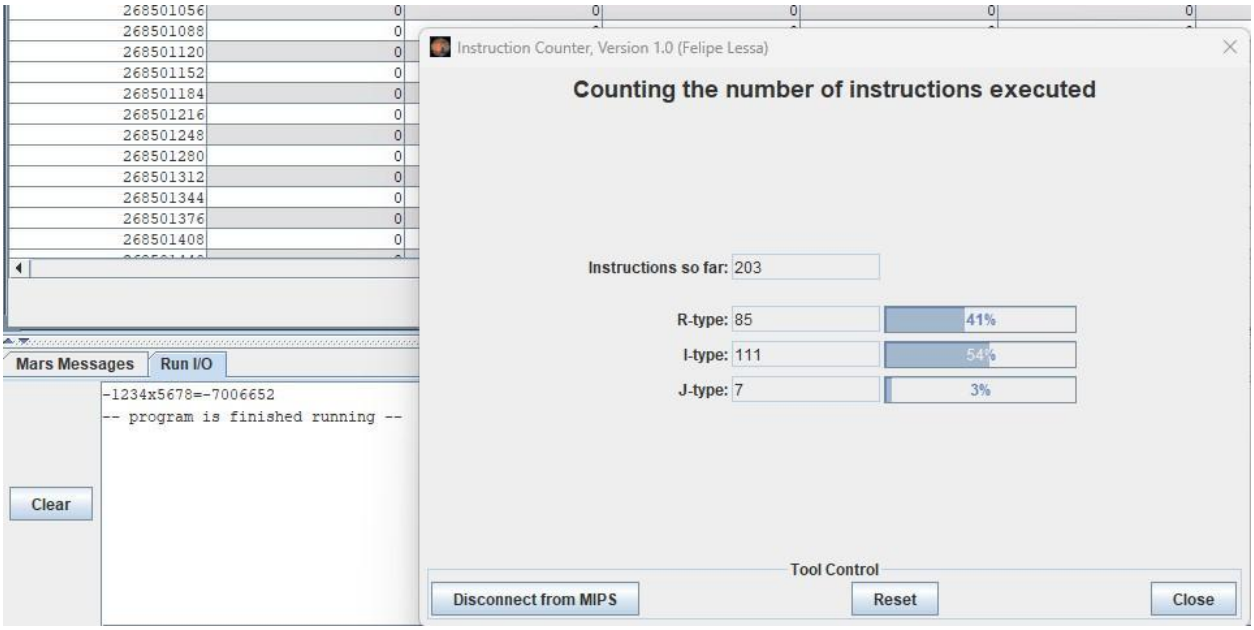
Trong đó:
- R-type: 85 lệnh chiếm 41%.

- I-type: 111 lệnh chiếm 54%.

- J-type: 7 lệnh chiếm 3%.

Tính toán thời gian thực thi:

$$CPU\ time = \frac{Instruction\ Count\ \times CPI}{Clock\ rate} = \frac{203\ \times 1}{1 \times 10^9} = 2.03 \times 10^{-7} s$$



Hình 3: Kết quả kiểm thử (2).

5.3 Ví dụ 3:

Hai số nguyên a,b được đọc vào lần lượt là -1234, -5678.

Tổng số lệnh: 406

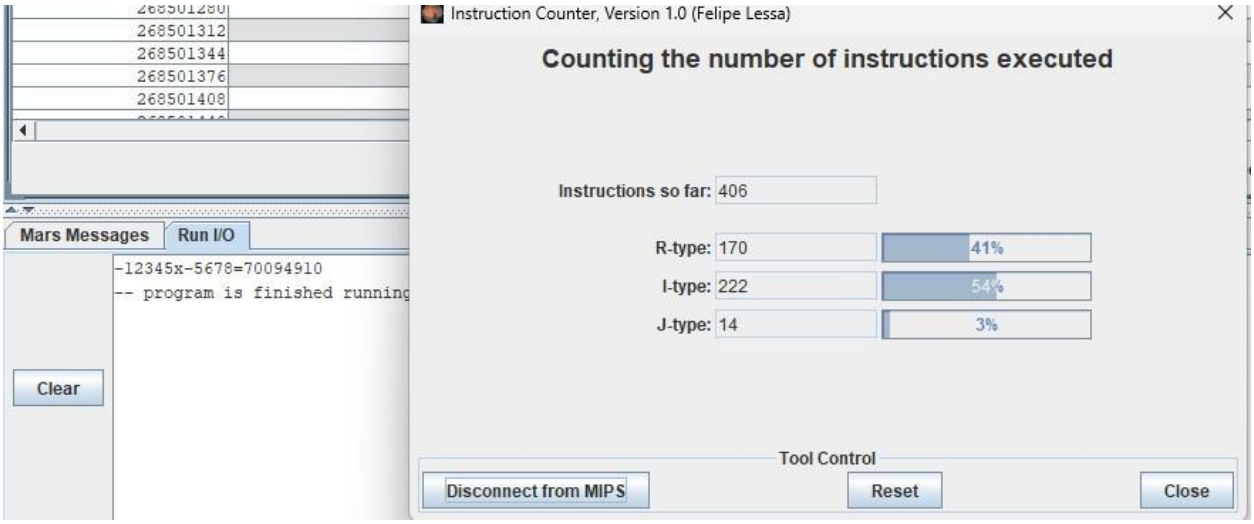
Trong đó:
- R-type: 170 lệnh chiếm 41%.

- I-type: 222 lệnh chiếm 54%.

- J-type: 14 lệnh chiếm 3%.

Tính toán thời gian thực thi:

$$CPU\ time = \frac{Instruction\ Count\ \times CPI}{Clock\ rate} = \frac{406\ \times 1}{1 \times 10^9} = 4.06 \times 10^{-7} s$$



Hình 4: Kết quả kiểm thử (3).

6 Kết luận:

Trong bài báo cáo này của nhóm em, chương trình hợp ngữ này có thể hiện thực hiện tính phép nhân 2 số nguyên 32 bit với 32 bit mà không sử dụng các lệnh nhân có sẵn một cách chính xác.

*Tuy nhiên, chương trình này còn hạn chế nếu phép nhân quá lớn kết quả vượt quá 32 bit thì chương trình sẽ cho ra kết quả không chính xác.