1. Phần giải thích bài làm

a) Xử lý đối với số thập phân:

- Trước hết nhập 2 số đầu vào là $s0,$s1. Ta đi thực hiện việc lấy giá trị tuyệt đối của chúng (nếu số đó âm thì lấy 0 trừ đi số đó), và xét phần dấu của chúng lưu vào $t0,$t1 (bit 0 : âm, bit 1 : dương).
- Thực hiện hàm DIV, đầu vào là 2 giá trị tuyệt đối vừa tìm được ở trên, vì giá trị tuyệt đối nên dĩ nhiên tối đa chỉ có 31 bits (vì đầu vào là số có dấu 32bits). Ở hàm DIV thực hiện thuật toán phép chia tuần tự tối ưu (được trình bày ở textbook). Kết quả trả về là phần thương $v0, dư $v1
- Tiếp đó ta đi thực hiện nhãn Print_RESULT_DIV để thực hiện việc in đúng chuẩn đầu ra là phần nguyên và phần được in theo số nguyên có dấu. Ta xét dấu của thương = (dấu của số bị chia) xor (dấu của số chia), dấu của phần dư = (dấu phần thương) xor (dấu của số chia) (cách xét dấu tham khảo ở cuốn CS422ComputerArchitectureComputerOrganizationAndDesign5thEdition2014 , ở cuốn này dấu phần dư có thể là số âm (thay vì ta dùng cơ sở lý thuyết của Euclid thì dấu phần dư luôn dương https://vi.wikipedia.org/wiki/Ph%C3%A9p_chia_c%C3%B3_d%C6%B0)). Vì là phép chia nên phần trị tuyệt đối của phần thương và phần dư chỉ có tối đa 31 bits)
- Tiếp theo ta thực hiện phép nhân là hàm Multiply, đầu vào là 2 thanh ghi là 2 số lấy trị tuyệt đối như đã nói ở trên đưa vào $a0,$a, đầu ra là là thanh ghi có giá trị là $v0=a,$v1=b, biểu diễn số z.

Số có dạng $z = a * 2^{32} + b \; với \; a, b \leq 2^{32} - 1$
(Phần này tham khảo thuật toán nhân tuần tự tối ưu trong textbook)

Sau đó ta thực hiện nhãn Print_RESULT_MUL để in kết quả phép nhân.
Thực tế ta chỉ thực hiện được phép nhân tối đa với 2 số trị tuyệt đối 31 bit là

$$(2^{31} - 1) * (2^{31} - 1) = 2^{62} - 2^{32} + 1 = (2^{30} - 1) * 2^{32} + 1$$

đúng theo chuẩn của số z nêu ở trên, ta nhận xét rằng, a= 2^{30}-1 thì chỉ có tối đa 30 bits khi biểu diễn số dương. Cho nên thực tế số a sẽ có tối đa 30 bits và số b có tối đa 32 bits. Để có thể áp dụng phép chia ở trên trong lúc tính toán ta cần đưa mỗi số về dạng số nguyên không dấu 31 bits tức là (a,b)=(30 bits;32 bits)=(31 bits, 31 bits). Phần này dễ dàng thực hiện bằng các phép dịch bits.

Khi đó $z = a * 2^{31} + b$ (a, b <= 2^{31} - 1)

Sau khi chuyển đổi xong ta chưng minh được phép lặp sau.
DO_Multiply:

Ta áp dụng phép chia phân tích được: a = 10x + y; b = 10u + v

(với x, y, u, v >= 0; y, v < 10)

Ta tính $z' = \frac{(10x+y)*2^{31}+(10u+v)}{10} = x * 2^{31} + u + \frac{y*2^{31}+v}{10}$

Ta thấy về lý thuyết $\frac{y*2^{31}+v}{10} + u \leq \frac{9*2^{31}+9}{10} + 9 < 2^{31} - 1$

Điều này chứng tỏ phần nguyên của $u + \frac{y*2^{31}+v}{10}$ trở thành phần 31 bits thấp trong phép lặp. Còn phần dư sẽ là các ký số của kết quả nhân này, ta lưa vào mảng $s7. Đặt $a = x; b = u + \frac{y*2^{31}+v}{10}$, Ta tiếp tục biểu diễn số z như vậy và tiếp tục vòng lặp. Điều kiện dừng khi a = 0 and b = 0.

Dấu của phép nhân thì ta dễ thấy là bằng (dấu của số bị nhân) xor (dấu của số nhân). Và ta in mảng $s7 theo thứ tự ngược lại thì ta được kết quả phép nhân.

b) Chuyển Hex sang Dec:

Ở phần nhân định dạng số hệ 10, việc in kết quả 2 thành ghi 32 bits thành số nguyên 64 bits khá là phức tạp. Thì phần nhân và chia định dạng hệ 16, đòi hỏi quá trình đổi một chuỗi kí tự hợp lệ thành số nguyên lưu trong thanh ghi và ngược lại. Tuy nhiên, điều này khá đơn giản khi chúng ta thao tác trên từng bit dữ liệu. Cụ thể:

- Chuyển số hệ 10 sang hệ 16
- Chuyển cơ số 16 sang cơ số 10:

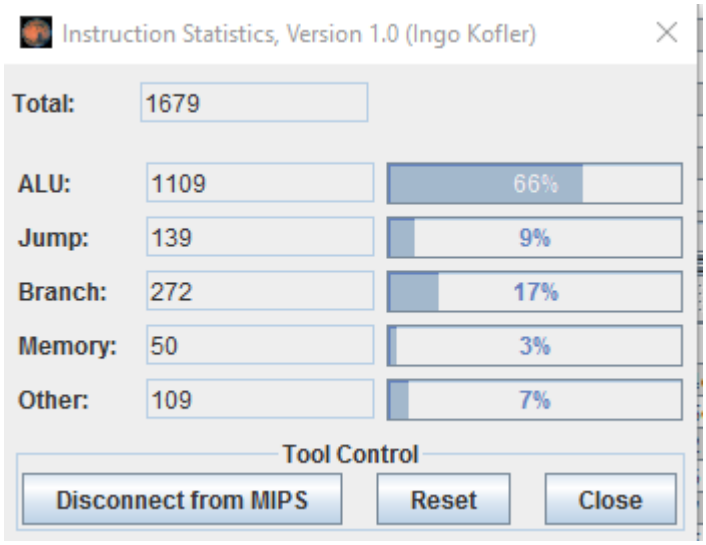
Để trình bày đẹp mắt và ngắn gọn code, thì ở phần hàm chuyển cơ số 16 sang cơ số 10 đỏi hỏi đầu vào chúng ta phải hợp lệ (kí tự chữ phải là in hoa).
Một chữ số hệ 16 được biểu diễn bởi 4 bit (0->F tương ứng với 0000 -> 1111: 0 -> 15 ở dạng cơ số 10) . Mặt khác, ở dạng kí tự ‘0’ -> ‘9’ được quy ra 48 -> 57, ‘A’-> ‘F’ được qui ra 65->70 ở cơ số 10.
Hay nói cách khác, để chuyển số dạng hex sang dec thì chúng ta chỉ cần xét từng bytes của string chứ hex. Nếu bytes đó có giá trị từ 48->57 thì ta trừ cho 48, còn nếu nằm trong 65->70 thì ta trừ cho 55. Sau đó dịch trái 4 bit ở kết quả rồi cộng trực tiếp kết quả đã tính lên vào cho đến khi đến cuối string ‘\0’.
Như vậy ta đã có được số hệ 10 từ hệ 16

2. Phần thống kê lệnh

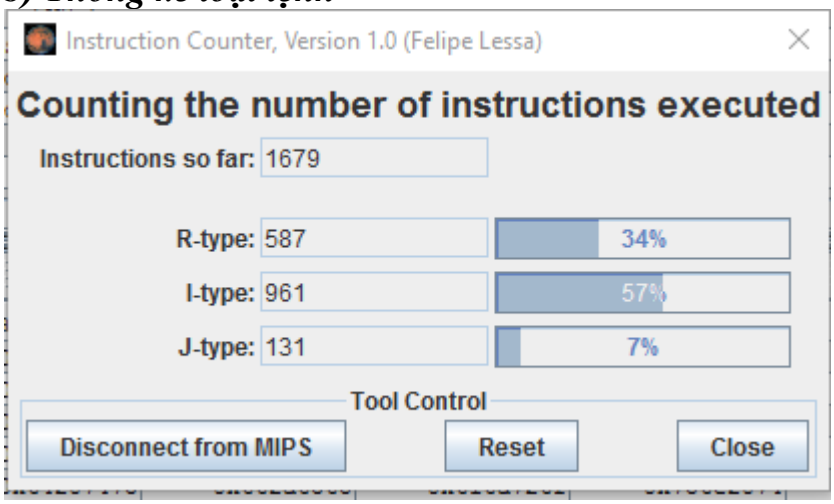
Ứng với input:

```
Please enter the mode!:
Mode=0 corresponds to computation in decimal, Mode=1 corresponds to computation in hexadecimal
1
Please enter the first hex-format number (multiplicand and dividend) with full 8 bits: 55555555
Please enter the second hex-format number (multiplier and divisor) with full 8 bits: 55555555
Result of multiplication in hexadecimal: 0x1C71C71C38E38E39
Quotient of division in hexadecimal: 0x00000001
Reminder of division in hexadecimal: 0x00000000
-- program is finished running --
```

a) Thống kê số lệnh



b) Thống kê loại lệnh



3. Phần tính toán

$$Execution\ Time = \frac{TotalInstructions \times CPI}{Clock\ Rate} = \frac{1679 \times 1}{2 \times 10^9} = 839{,}5\ ns$$

4. Phần Test

Test 1:

```
Please enter the mode!:
Mode=0 corresponds to computation in decimal, Mode=1 corresponds to computation in hexadecimal
0
Please enter the first dec-format number (multiplicand and dividend):2147483647
Please enter the second dec-format number (multiplier and divisor):2147483647

Quotient of division in hexadecimal:1
Reminder of division in hexadecimal: 0
 Result of multiplication in decimal:4611686014132420609
-- program is finished running (dropped off bottom) --
```

Test 2:

```
Please enter the mode!:
Mode=0 corresponds to computation in decimal, Mode=1 corresponds to computation in hexadecimal
0
Please enter the first dec-format number (multiplicand and dividend):69
Please enter the second dec-format number (multiplier and divisor):0

 Invalid division! Divisor can't equal zero

 Result of multiplication in decimal:0
-- program is finished running (dropped off bottom) --
```

Test 3:

```
Please enter the mode!:
Mode=0 corresponds to computation in decimal, Mode=1 corresponds to computation in hexadecimal
0
Please enter the first dec-format number (multiplicand and dividend):894515
Please enter the second dec-format number (multiplier and divisor):-1229

Quotient of division in hexadecimal:-727
Reminder of division in hexadecimal: 1032
 Result of multiplication in decimal:-1099358935
-- program is finished running (dropped off bottom) --
```

Test 4:

```
Please enter the mode!:
Mode=0 corresponds to computation in decimal, Mode=1 corresponds to computation in hexadecimal
0
Please enter the first dec-format number (multiplicand and dividend):2147483647
Please enter the second dec-format number (multiplier and divisor):2147483647

Quotient of division in hexadecimal:1
Reminder of division in hexadecimal: 0
 Result of multiplication in decimal:4611686014132420609
-- program is finished running (dropped off bottom) --
```

Test 5:

```
Please enter the mode!:
Mode=0 corresponds to computation in decimal, Mode=1 corresponds to computation in hexadecimal
0
Please enter the first dec-format number (multiplicand and dividend):2147483647
Please enter the second dec-format number (multiplier and divisor):-2147483647

Quotient of division in hexadecimal:-1
Reminder of division in hexadecimal: 0
 Result of multiplication in decimal:-4611686014132420609
-- program is finished running (dropped off bottom) --
```

Test 6:

```
Please enter the mode!:
Mode=0 corresponds to computation in decimal, Mode=1 corresponds to computation in hexadecimal
0
Please enter the first dec-format number (multiplicand and dividend):13456
Please enter the second dec-format number (multiplier and divisor):25

Quotient of division in hexadecimal:538
Reminder of division in hexadecimal: 6
 Result of multiplication in decimal:336400
-- program is finished running (dropped off bottom) --
```

Test 7:

```
Please enter the mode!:
Mode=0 corresponds to computation in decimal, Mode=1 corresponds to computation in hexadecimal
0
Please enter the first dec-format number (multiplicand and dividend):8576
Please enter the second dec-format number (multiplier and divisor):-13

Quotient of division in hexadecimal:-659
Reminder of division in hexadecimal: 9
 Result of multiplication in decimal:-111488
-- program is finished running (dropped off bottom) --
```

Test 8:

```
Please enter the mode!:
Mode=0 corresponds to computation in decimal, Mode=1 corresponds to computation in hexadecimal
1
Please enter the first hex-format number (multiplicand and dividend) with full 8 bits: 7FFFFFFF
Please enter the second hex-format number (multiplier and divisor) with full 8 bits: FFFFFFFF
Result of multiplication in hexadecimal: 0xFFFFFFFF80000001
Quotient of division in hexadecimal: 0x80000001
Reminder of division in hexadecimal: 0x00000000
-- program is finished running --
```

Test 9:

```
Please enter the mode!:
Mode=0 corresponds to computation in decimal, Mode=1 corresponds to computation in hexadecimal
1
Please enter the first hex-format number (multiplicand and dividend) with full 8 bits: 8FFFFFFF
Please enter the second hex-format number (multiplier and divisor) with full 8 bits: 8FFFFFFF
Result of multiplication in hexadecimal: 0x31000000E0000001
Quotient of division in hexadecimal: 0x00000001
Reminder of division in hexadecimal: 0x00000000
-- program is finished running --
```

Test 10:

```
Please enter the mode!:
Mode=0 corresponds to computation in decimal, Mode=1 corresponds to computation in hexadecimal
1
Please enter the first hex-format number (multiplicand and dividend) with full 8 bits: 00000000
Please enter the second hex-format number (multiplier and divisor) with full 8 bits: 00000000
Result of multiplication in hexadecimal: 0x0000000000000000
Invalid division! Divisor can't equal zero
-- program is finished running --
```