CE119-Lab04

1. Thực hành với thủ tục

```
        .data
prompt: .asciiz "Enter one number: "

        .text
main:   jal getInt
        move $s0, $v0
        j exit

getInt: li $v0, 4
        la $a0, prompt
        syscall

        li $v0, 5
        syscall
        jr $ra

exit:
```

Hình 1: nhập từ cửa sổ I/O một số nguyên vào thanh ghi $s0

```
. . .
        move    $a0, $s0
        move    $a1, $s1
        move    $a2, $s2
        move    $a3, $s3
        jal     proc_example

        move    $a0, $v0
        li      $v0, 1
        syscall
. . .
proc_example:
        addi    $sp, $sp, -4
        sw      $s0, 0($sp)
        add     $t0, $a0, $a1
        add     $t1, $a2, $a3
        sub     $s0, $t0, $t1
        move    $v0, $s0
        lw      $s0, 0($sp)
        addi    $sp, $sp, 4
        jr      $ra
```

Hình 2: Phép toán (a + b) - (c + d)

- Chạy từng bước và theo dõi sự thay đổi của thanh ghi PC, $ra, $sp, $fp trong 2 hình trên.
- Chạy toàn chương trình một lần để xem kết quả
- Với code trong Hình 1, nếu bỏ dòng code "j exit", việc gì sẽ xảy ra?
- Viết lại code trong Hình 1, thêm vào thủ tục tên showInt để in ra cửa sổ I/O giá trị của số int nhập vào cộng thêm 1.
- Viết lại code trong Hình 2, lúc này chương trình chính cần tính giá trị của cả hai biểu thức: (a + b) – (c + d) và (a – b) + (c – d), hàm proc_example có hai giá trị trả về và trong thân hàm sử dụng hai biến cục bộ $s0 và $s1 ($s1 lưu kết quả của (a – b) + (c – d))
- Viết lại code trong Hình 2, lúc này chương trình chính cần tính giá trị của cả hai biểu thức: (a + b) – (c + d), (e - f) hàm proc_example có 6 input và hai giá trị trả về và trong thân hàm sử dụng hai biến cục bộ $s0 và $s1 ($s1 lưu kết quả của e - f)

2. Thực hành với đệ quy

Khi thân của một thủ tục gọi một thủ tục khác, giá trị của thanh ghi $ra ngay lập tức bị ghi đè, xóa mất giá trị cũ, dẫn tới sau khi thủ tục này thực hiện xong sẽ không có địa chỉ để quay về. Vì vậy, nếu trong trường hợp lồng thủ tục hay đệ quy, thanh ghi $ra cũng phải được lưu lại cùng các thanh ghi $s.

Viết code MIPS, chạy kiểm thử trên MARS và giải thích ý nghĩa rõ ràng cho chương trình tính giai thừa được viết bằng code C như Hình 3.

```
main ()
{
   printf ("The factorial of 10 is %d\n", fact (10));
}

int fact (int n)
{
   if (n < 1)
      return (1);
   else
      return (n * fact (n - 1));
}
```

Hình 3: Tính giai thừa

3. Bài tập

1. Tiếp tục nội dung 2. Vẽ lại hình ảnh của các stack trong trường hợp tính 5! và 10!
2. Viết chương trình nhập vào n và xuất ra chuỗi Fibonacci tương ứng

--------*Hết*--------