

Computer Architecture Lab Mid term

Full name: Nguyễn Hồng Phúc

Student ID: 20225659

Bài 1:

- Ý tưởng bài toán:

1. Nhập số nguyên N.
2. Sau đó chúng ta chia N cho 10 để lấy các chữ số lưu vào mảng \$s1 và \$s2.
3. Với \$s1 mang giá trị địa chỉ đầu của mảng. Còn \$s2 mang giá trị cuối của mảng. Cho \$s1 và \$s2 chạy từ 2 đầu của mảng để tính tổng từ 2 đầu.
4. Vòng lặp chạy đến khi \$s1 lớn hơn hoặc bằng \$s2 thì dừng lại.
5. So sánh hai tổng nếu bằng nhau thì thông báo “là số may mắn”.
6. Nếu không thì thông báo “Không phải là số may mắn”

- Ý nghĩa của các chương trình con:

1. Hàm xuly dùng để lưu các giá trị phần tử vào mảng.
2. Hàm tính tổng dùng để tính tổng từ 2 đầu của mảng. Sử dụng các lệnh lw, addi và sub để di chuyển địa chỉ thanh ghi.

-Kết quả chương trình:

The screenshot shows the Mars MIPS simulator interface. The **Text Segment** window displays the following assembly code:

Bkpt	Address	Code	Basic	Source
	0x00400000	0x24020005	addiu \$2,\$0,5	6: li \$v0,5
	0x00400004	0x0000000c	syscall	7: syscall
	0x00400008	0x00028021	addu \$16,\$0,\$2	8: move \$s0,\$v0 # s0 la N
	0x0040000c	0x3c011001	lui \$1,4097	9: la \$s1,mang #s1 la mang luu cac gia tri chu so
	0x00400010	0x34310000	ori \$17,\$1,0	
	0x00400014	0x00119021	addu \$18,\$0,\$17	10: move \$s2,\$s1 #s2 dung de luu mang nguoc lai
	0x00400018	0x240b0000	addiu \$11,\$0,0	11: li \$t3,0 #t3 luu tong 1
	0x0040001c	0x240c0000	addiu \$12,\$0,0	12: li \$t4,0 #t4 luu tong 2
	0x00400020	0x08100009	j 0x00400024	13: j xuly
	0x00400024	0x2408000a	addiu \$9,\$0,10	15: li \$t0,10
	0x00400028	0x0208001a	div \$16,\$8	16: div \$s0,\$t0
	0x0040002c	0x00004810	mfhi \$9	17: mfhi \$t1 # luu phan du

The **Data Segment** window shows memory addresses and values for various segments. The **Labels** window lists labels and their addresses. The **Run I/O** window shows the program's output:

```
1234600
la so may man
-- program is finished running --
```

Bài 2:

- Ý tưởng bài toán:

1. Đầu tiên nhập giá trị số phần tử của mảng
2. Nhập các phần tử vào thanh ghi. Lưu địa chỉ thanh ghi đầu tiên ở \$t1
3. Thiết lập các thanh ghi \$t3 lưu tổng các số dương còn \$t4 lưu tổng các số chẵn.
4. Thực hiện load phần tử trong mảng. Xem nó có phải số dương hay không. Sau đó xem nó có phải số chẵn hay không. Tính tổng các phần tử vào các thanh ghi \$t3, \$t4.
5. Tiếp tục vòng lặp cho hết phần tử cuối cùng thì thông báo kết quả đã tính được.

- Ý nghĩa các chương trình con:

1. Vòng lặp loop_read_element dùng để nhập các phần tử của mảng
2. Vòng lặp loop_calc_sum dùng để tính các tổng. Trong đó đầu tiên chúng ta xét nó có phải số dương hay không, nếu không thì nhảy sang nhãn not_positive để tính xem nó có chẵn hay không.
3. Nếu nó đã là số dương thì tính tổng vào \$t3 rồi lại xét tiếp liệu phần tử đó có chẵn hay không. Nếu có thì tính tổng lưu vào \$t4. Nếu không nhảy đến phần tử tiếp theo bằng nhãn next_iteration.

- Kết quả chương trình:

The screenshot displays a MIPS simulator interface with several panels:

- Text Segment:** Shows assembly code with addresses, hex codes, and comments. Key instructions include `addiu $2,$0,4`, `li $v0, 4`, `la $a0, size_prompt`, `syscall`, `move $t0, $v0`, `la $t1, array`, `li $t2, 0`, and `la $a0, elem_prompt`.
- Labels:** Lists labels such as `main`, `loop_read_element`, `loop_calc_sum`, `not_positive`, `even`, `next_iteration`, `array`, `size_prompt`, `elem_prompt`, and `pos_sum_msg` with their corresponding addresses.
- Registers:** A table showing the state of registers \$zero through \$31. Notable values include \$t0 = 268501016, \$t3 = 14, and \$t4 = 6.
- Data Segment:** A memory dump showing values at various addresses, including 1850015776, 544367988, 1835363429, 980708965, 1968373792, 1718558829, 1936683040, 1986622569, 1970151525, 1919246957, 2112115, 544044371, 1696622191, 544105846, 1651340654, 980644453, and 655392.
- Mars Messages / Run I/O:** Shows the sequence of user inputs and program outputs:

```
Enter element: 1
Enter element: 2
Enter element: 3
Enter element: -4
Enter element: -5
Enter element: 8
Sum of positive numbers: 14
Sum of even numbers: 6
```

Bài 3:

- Ý tưởng bài toán:

1. Khai báo vùng nhớ \$s1 và \$s2 để lưu trữ hai chuỗi
2. Trong hàm main:

- Nhập 2 chuỗi từ bàn phím
- Gọi hàm compare_string để so sánh hai chuỗi
- ở trong hàm compare_string lần lượt load dữ liệu các kí tự của 2 mảng có cùng thứ tự sau đây chuyển về chữ Hoa bằng lệnh andi. Kiểm tra xem 2 chuỗi có kết thúc cùng 1 lúc không (2 chuỗi có độ dài bằng nhau không) bằng nhãn check_null. Nếu không có độ dài bằng nhau lập tức thông báo “khác nhau”
- Thông báo 2 chuỗi giống nhau nếu chạy đến hết chuỗi.

- Ý nghĩa các chương trình con:

1. Hàm compare_string sử dụng 2 con trỏ để duyệt từng kí tự của 2 chuỗi. Kiểm tra kí tự tương ứng của hai chuỗi. Nếu không giống nhau thì trả về 0. Nếu kết thúc cả hai chuỗi cùng 1 lúc thì trả về 1.

- Kết quả chạy chương trình:

The screenshot displays a MIPS simulator interface with three main panels:

- Text Segment:** Shows assembly code with addresses, codes, basic instructions, and source comments. Key instructions include `addiu $2,$0,0x00000009`, `li $v0, 4`, `syscall`, and `li $a0, prompt1`.
- Data Segment:** Displays memory addresses and their corresponding values in hexadecimal and ASCII. The ASCII values show the strings "abcDEF" and "ABCDEF".
- Registers:** A table listing registers \$zero through \$31, their numbers, and their current values. Most registers are 0x00000000, while \$v0 is 0x00000004.

At the bottom, the **Mars Messages** panel shows the program's output:

```
Enter the first string: abcDEF
Enter the second string: ABCDEF
The strings are equal.

-- program is finished running --
```