TRƯỜNG ĐẠI HỌC BÁCH KHOA TP HCM
KHOA KHOA HỌC VÀ KỸ THUẬT MÁY TÍNH

- - - - - - - - - - - -

KIẾN TRÚC MÁY TÍNH

Báo cáo Bài tập lớn

GVHD:	Trần Thanh Bình	
	Võ Tấn Phương	
SV thực hiện:	Nguyễn Xuân Hiến	1652192
	Nguyễn Huỳnh Thoại	1613379
	Lê Hữu Vinh	1614117

Tp. Hồ Chí Minh, Tháng 11/2017

1 Đề bài

- Sắp xếp chuỗi.
- Cho một chuỗi số nguyên 20 phần tử. Sử dụng hợp ngữ assembly MIPS, viết thủ tục sắp xếp chuỗi đó theo tứ tự tăng dần theo giải thuật merge sort. Yêu cầu xuất ra từng bước trong quá trình demo

2 Ý tưởng

- Chia mảng lớn thành những mảng con nhỏ hơn bằng cách chia đôi mảng lớn và chúng ta tiếp tục chia đôi các mảng con cho tới khi mảng con nhỏ nhất chỉ còn 1 phần tử. Sau đó chúng ta sẽ tiếng hành so sánh 2 mảng con có cùng mảng cơ sở (khi chúng ta chia đôi mảng lớn thành 2 mảng con thì mảng lớn đó chúng ta gọi là mảng cơ sở của 2 mảng con đó) khi so sánh chúng sẽ vừa sắp xếp vừa ghép 2 mảng con đó lại thành mảng cơ sở, chúng ta tiếp tục so sánh và ghép các mảng con lại đến khi còn lại mảng duy nhất thì đó là mảng đã được sắp xếp.

3 Code MIPS

```
    .data
A:    .word  2,5,6,8,3,1,9,0,4,7,14,19,13,16,12,15,18,11,10,17
temp:  .word  0:20
space:  .asciiz  " "
endl:   .asciiz  "\n"
    .text

MAIN:

   la  $a0,A
   la  $a1,temp
   ori  $a2,$zero,0                    # Left
   ori     $a3,$zero,19                # Right

   jal  PRINT                          # In ra mang truoc khi sap xep
   jal  MERGESORT                      # Goi ham sap xep

   j  ENDPROGRAM                       # ket thuc chuong trinh

MERGESORT:

   beq $a2,$a3,ENDFUNC                 # if Left = right ket thuc
   add $s0,$a2,$a3                     # mid = left + right
   srl $s0,$s0,1                       # mid=mid / 2;

   addi $sp,$sp,-12
   sw $ra,8($sp)
   sw $a1,4($sp)
   sw $a2,0($sp)
   addi $a1,$a2,0
   addi $a2,$s0,0
   jal  PRINT
   lw $a2,0($sp)
   lw $a1,4($sp)
   lw $ra,8($sp)
   addi $sp,$sp,12

   addi $sp,$sp,-16                      # Cap phat 16 byte cho stack frame
   sw $s0,12($sp)                        # Luu mid vao stack
```

```
   sw $ra,8($sp)                    # Luu thanh ghi $ra vao stack
   sw $a3,4($sp)                    # Luu right vao stack
   sw $a2,0($sp)                    # Luu left vao stack


   addi $a3,$s0,0                   # right = mid
   jal MERGESORT                    # Goi ham sap xep tu  left toi mid

   lw $a3,4($sp)                    # Tra lai right
   lw $s0,12($sp)                   # Tra lai mid

   addi $sp,$sp,-12                 # Cap phat 12 byte cho stack frame
   sw $ra,8($sp)                    # Luu thanh ghi $ra vao stack
   sw $a1,4($sp)                    # Luu mang temp vo stack
   sw $a2,0($sp)                    # Luu left vo stack
   addi $a1,$s0,1                   # temp = mid + 1
   addi $a2,$a3,0                   # Left = right
   jal  PRINT
   lw $a2,0($sp)                    # Tra lai left
   lw $a1,4($sp)                    # Tra lai mang temp
   lw $ra,8($sp)                    # Tra lai thanh ghi $ra
   addi $sp,$sp,12

   addi $a2,$s0,1                    # left = mid + 1
   jal MERGESORT                     # Goi ham sap xep tu mid +1 toi right

   lw $a2,0($sp)                     # Tra lai left
   lw $a3,4($sp)                     # Tra lai right
   lw $ra,8($sp)                     # Tra lai $ra
   lw $s0,12($sp)                    # Tra lai mid
   addi $sp,$sp,16                   # giai phong stack frame

   addi $t3,$a2,0                    # Bien chay i = left

LOOP_ONE:

   slt $at,$a3,$t3                   # if ( mid < i ) $at = 1 else $at = 0
   bne $at,$zero,ENDLOOP_ONE         # if ( $at! = 0) ketthuc
   sll $t4,$t3,2                     # $t4 = i*4
   add $t5,$a0,$t4                   # $t5 = $t4+a[]=a + i*4
   lw $t5,0($t5)
   add $t6,$a1,$t4                   # $t6 = temp + i*4
   sw $t5,0($t6)                     # $t5 = $t6
   addi $t3,$t3,1                    # i++
   j LOOP_ONE

ENDLOOP_ONE:

   addi $t0,$s0,1                    # i2 = mid +1
   addi  $t1,$a2,0                   # i1 = left
   addi $t2,$t0,0                    # $t2 = mid +1
   addi $t3,$a2,0                    # curr = left

LOOPFOR:

   slt $at,$a3,$t3                   # if ( right < i ) $at=1 else $at=0
   bne $at,$zero,ENDFOR              # if ( $at!=0 ) ket thuc
```

```
   sll $t5,$t3,2                        # $t5 = i*4
   add $t5,$t5,$a0                      # $t5 = a + i*4

   bne $t1,$t0,ELSE_ONE                 # if ( left != mid+1 ) nhay den else_one
   sll     $t6,$t2,2                    # $t6 = $t2*4
   add $t6,$t6,$a1                      # $t6 = temp+ $t2*4
   lw $t6,0($t6)
   sw $t6,0($t5)                        # $t5 = $t6
   addi $t2,$t2,1
   j NEAREND

ELSE_ONE:

   slt $at,$a3,$t2                      # if ( right < $t2) $at=1 else $at = 0
   beq $at,$zero,ELSE_TWO               # if ( $at != 0 ) else two
   sll     $t6,$t1,2                    # $t6 = $t1*4
   add $t6,$t6,$a1                      # $t6 = temp+ $t1*4
   lw $t6,0($t6)
   sw $t6,0($t5)                        # a+i*4 = temp+$t1*4
   addi $t1,$t1,1                       # $t1 ++
   j NEAREND

ELSE_TWO:

   sll $t7,$t1,2                        # $t7 = $t1*4
   add $t7,$a1,$t7                      # $t7 = a + $t1*4
   lw $t7,0($t7)
   sll $t8,$t2,2                        # $t8 = $t2 * 4
   add $t8,$a1,$t8                      # $t8=temp + $t2*4
   lw $t8,0($t8)
   slt $at,$t7,$t8                      # if ( $t7 < $t8 ) $at=1 else $at=0
   beq $at,$zero,ELSE_THREE             # if ( $at != 0 ) else three
   sll     $t6,$t1,2
   add $t6,$t6,$a1                      # $t6 = temp+ $t1*4
   lw $t6,0($t6)
   sw $t6,0($t5)                        # $t5=$t6
   addi $t1,$t1,1                       # $t1++
   j NEAREND

ELSE_THREE:

   sll     $t6,$t2,2
   add $t6,$t6,$a1                      # $t6 = temp+ $t2*4
   lw $t6,0($t6)
   sw $t6,0($t5)                        # $t5 =$t6
   addi $t2,$t2,1                       # $t2++
NEAREND:
   addi $t3,$t3,1                       # curr++
   j LOOPFOR

ENDFOR:

   addi $sp,$sp,-12
   sw $ra,8($sp)
   sw $a1,4($sp)
   sw $a2,0($sp)
   addi $a1,$a2,0
   addi $a2,$a3,0
```

```
   jal  PRINT
   lw $a2,0($sp)
   lw $a1,4($sp)
   lw $ra,8($sp)
   addi $sp,$sp,12

ENDFUNC:

   jr $ra



PRINT:
    addi $sp,$sp,-4                    # Cap phat 4 byte cho stack
    sw  $a0,0($sp)                     # Luu mang A vao stack

    addi $s7,$a0,0                     # $s7 = A[]
    add $t8,$a1,$0                     # i = left
LOOP:
    slt $at,$a2,$t8                    # if ( $a2 < $t8) $at=1 else $at=0
    bne  $at,$zero,ENDPRINT            # if ( $at != 0) ket thuc in
    sll  $t9,$t8,2                     # $t9 = i*4
    add  $a0,$s7,$t9                   # $a0=A[]+i*4
    lw   $a0,0($a0)
    li   $v0,1
    syscall
    la   $a0,space                     # In dau cach
    li   $v0,4
    syscall
    addi $t8,$t8,1                     # i++
    j LOOP
ENDPRINT:

    la   $a0,endl                      # Xuong dong
    li   $v0,4
    syscall
    lw  $a0,0($sp)                     # Tra lai giai tri cua mang A[]
    addi $sp,$sp,4                     # Giai phong stack
    jr $ra


ENDPROGRAM:
li $v0,10
syscall
```

4 Kết quả

```
2 5 6 8 3 1 9 0 4 7                    # Chia đôi mảng A,left
2 5 6 8 3
2 5 6
2 5
2                                      # Không thể chia tiếp
5
2 5                                    # Gộp 2 với 5
6                                      # Không thể chia 6
2 5 6                                  # Gộp và sắp xếp 2,5 với 6
8 3                                    # Chia đôi 8,3
8
3
3 8                                    # Gộp và sắp xếp 8 và 3
2 3 5 6 8                              # Gộp và sắp xếp 2,5,6 với 3,8
1 9 0 4 7                              # Chia đôi mảng bên phải của mảng bên trái
1 9 0
1 9
1                                      # Chỉ còn 1 phần tử không thể chia tiếp
9
1 9                                    # Gộp và sắp xếp 1 và 9
0
0 1 9                                  # Gộp và sắp xếp 1,9 với 0
4 7                                    # Chia đôi 4 và 7
4
7
4 7                                    # Gộp 4 và 7
0 1 4 7 9                              # Gộp 0,1,9 và 4,7
0 1 2 3 4 5 6 7 8 9                    # Gộp 2 3 5 6 8 và 0 1 4 7 9
14 19 13 16 12 15 18 11 10 17          # Chia đôi mảng bên phải
14 19 13 16 12
14 19 13
14 19
14                                     # Chỉ còn 1 phần tử là 14 và 19
19
14 19                                  # Gộp và sắp xếp 14 với 19
13
13 14 19                               # Gộp 14,19 với 13
16 12                                  # Chia đôi 16 , 12
16
12
12 16                                  # Gộp 16 và 12
12 13 14 16 19                         # Gôp 13,14,19 với 12,16
15 18 11 10 17                         # Chia đôi mảng bên phải của mảng bên phải
15 18 11
15 18
15                                     # Còn 1 phần tử
18
15 18                                  # Gộp 15 với 18
11
11 15 18                               # Gộp 15,18 với 11
10 17                                  # Chia đôi 10,17
10
17
10 17                                  # Gộp 10,17
10 11 15 17 18                         # Gộp 15,18,11 với 10,17
10 11 12 13 14 15 16 17 18 19          # Gộp 12 13 14 16 19 và 10 11 15 17 18
```

```
0 1 2 3 4 5 6 7 8 9 10 11 12 13 14 15 16 17 18 19  # Gộp hai mảng bên trái và bên phải
```

5 Thống kê số lệnh và loại lệnh

- *Tổng số lệnh:* 6780
- *R-Type:* 2374 Chiếm 35 %
- *I-Type:* 3889 chiếm 57 %
- *J-Type:* 517 chiếm 7 %

6 Tính toán thời gian

- $Clockcycles = InstructionCount * CPI = 6780 * 1 = 6780$
- $CPUExecutionTime = \frac{Clockcycles}{Clockrate} = \frac{6780}{4*10^9} = 1.695\mu s$

7 Code C++

```
#include <iostream>
using namespace std;
void In(int a[],int d, int c)
{
    for (int i = d; i <= c; i++)
    {
         cout << a[i] << " ";
    }
    cout << endl;
}
void mergesort(int A[], int temp[], int left, int right)
{
    if (left == right) return; // List of one element
    int mid = (left + right) / 2;
    In(A,left, mid);
    mergesort(A, temp, left, mid);
    In(A,mid + 1, right);
    mergesort(A, temp, mid + 1, right);
    for (int i = left; i <= right; i++) // Copy subarray to temp
         temp[i] = A[i];
    // Do the merge operation back to A
    int i1 = left; int i2 = mid + 1;
    for (int curr = left; curr <= right; curr++) {
        if (i1 == mid + 1) // Left sublist exhausted
            A[curr] = temp[i2++];
        else if (i2 > right) // Right sublist exhausted
            A[curr] = temp[i1++];
        else if (temp[i1] < temp[i2])
            A[curr] = temp[i1++];
        else A[curr] = temp[i2++];
    }
    In(A,left, right);
}

int main()
{
    int a[] = { 1, 2, 5, 7, 4, 3, 9, 3, 4, 7, 8, 9 };
    int b[12];
```

```
    mergesort(a, b, 0, 11);
    return 0;
}
```