DHA Suffa University
Department of Computer Science
Computer Organization & Assembly Language
Spring 2021
Lab # 04 (Branches)

Objective:

To deal with branches and jumps in MIPS.

A **branch** is an instruction in a computer program that can cause a computer to begin executing a different instruction sequence and thus deviate from its default behavior of executing instructions in order. Branch may also refer to the act of switching execution to a different instruction sequence as a result of executing a branch instruction. Branch instructions are used to implement control flow in program loops and conditionals.

A **jump** instruction, like "jmp", just switches the CPU to executing a different piece of code

Branches

- Comparison for conditional branches is built into instruction

```
b       target          #  unconditional branch to program label target
beq     $t0,$t1,target  #  branch to target if  $t0 = $t1
blt     $t0,$t1,target  #  branch to target if  $t0 < $t1
ble     $t0,$t1,target  #  branch to target if  $t0 <= $t1
bgt     $t0,$t1,target  #  branch to target if  $t0 > $t1
bge     $t0,$t1,target  #  branch to target if  $t0 >= $t1
bne     $t0,$t1,target  #  branch to target if  $t0 <> $t1
```

Example 1:

```
# Take two numbers from the user and tell which is greater
.data
var1 : .asciiz "Enter the first value\n"
var2 : .asciiz "Enter the second value\n"
.globl main
.text
main:
        li $v0 , 4
        la $a0 , var1
        syscall

        li $v0 , 5
        syscall
        move $t0 , $v0

        li $v0 , 4
        la $a0 , var2
        syscall

        li $v0 , 5
        syscall
        move $t1 , $v0

        bgt  $t0 , $t1 , ifGreater
        move $t2 , $t1
        b    printNow

ifGreater:
        move $t2 , $t0
printNow:
        move $a0 , $t2
        li $v0, 1
        syscall

        li $v0 , 10
        syscall
```

Example 2:

```
.data
max: .word 7
value1: .word 1
.text
lw $t0, max
lw $t1, value1
Looping:
beq $t1, $t0, exit
move $a0, $t1
li $v0,1
syscall
add $t1, $t1, 1
j Looping
exit:
li $v0, 10
syscall
```

Example 3

```
.data
string1: .asciiz "This is "
string2: .asciiz "\n"
num1: .word 1
num2: .word 6
.text

lw $t0, num1
lw $t1, num2

Loop:
        beq $t0, $t1, Exit

        la $a0, string1
        li $v0, 4
        syscall

        move $a0, $t0
        li $v0, 1
        syscall

        add $t0, $t0, 1

        la $a0, string2
        li $v0, 4
```

```
        syscall

        b Loop

Exit:
        li $v0, 10
        syscall
```

LAB Task 04

(1) Write the MIPS code for the following C code:

```
int main() {
  int a=1;
  int b=1;
  int number =1;
  printf("Enter the table number\n");
  scanf("%d", &a);

  while(b != 11)
  {
     number = a*b;

  printf("%d * %d = %d\n", a,b,number);
  b++;
  }

  return 0;}
```

(2) Write the MIPS code for the following C code:

```
int main()
{
   int num, count, sum = 0;
   printf("Enter a positive integer: ");
   scanf("%d", &num);
   for(count = 1; count <= num; count++)
   {
      sum =sum+count;
   }
   printf("Sum = %d", sum);
   return 0;
}
```

(3) Write the MIPS code for the following C code:

```
int main()
{
   int i=0;

for(i=15; i>0; i-=2)
            {
            printf("%d",i);
            }

   return 0;
}
```