COMSATS UNIVERSITY ISLAMABAD

Name:	Muhammad Attiq
Reg. No:	FA23 – BCE – 060
Section:	BCE – 5A
Subject:	Computer Organization & Architecture
Instructor:	Dr. Irfanullah
Lab Report:	2

Lab # 02 Strings and Conditional Execution

Objectives

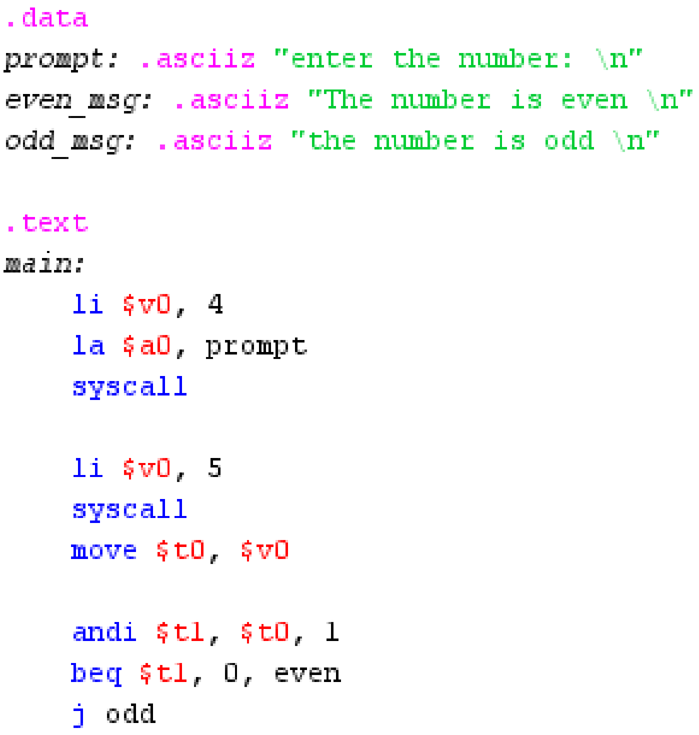
- Using syscall for character string output
- Conditional Execution using 'branch' and 'jump' instruction for implementing
    - Loops
    - If – else statements

In-Lab

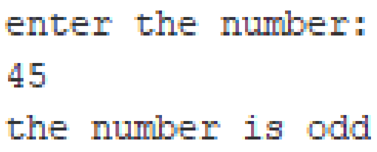
Tasks

- Write a program that gets a number from user and displays "Even", or "Odd. (If-else)

Code

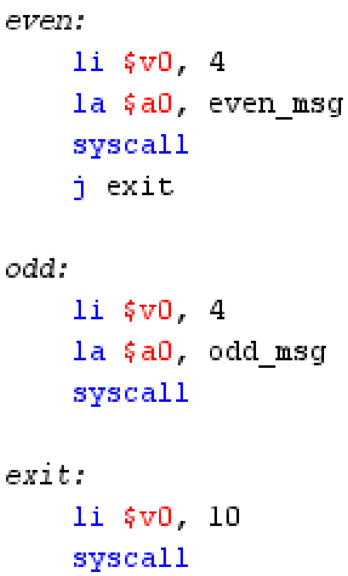


```
.data
prompt: .asciiz "enter the number: \n"
even_msg: .asciiz "The number is even \n"
odd_msg: .asciiz "the number is odd \n"

.text
main:
    li $v0, 4
    la $a0, prompt
    syscall

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

    andi $t1, $t0, 1
    beq $t1, 0, even
    j odd
```



```
enter the number:
45
the number is odd
```

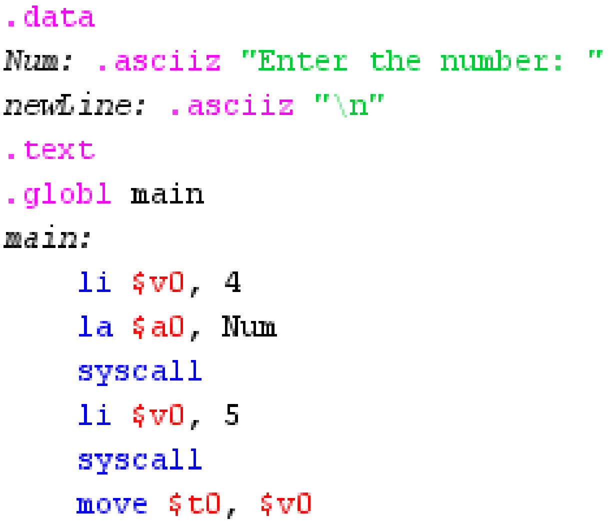
Output



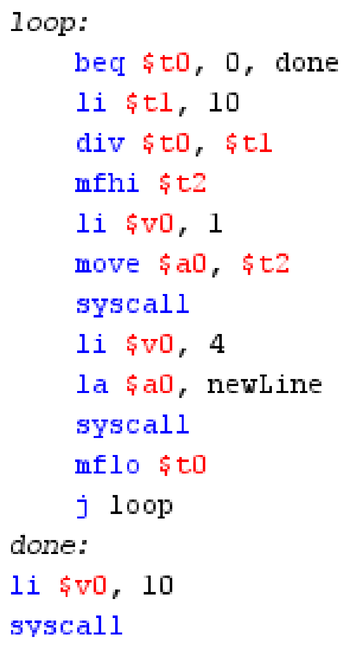
```
even:
    li $v0, 4
    la $a0, even_msg
    syscall
    j exit

odd:
    li $v0, 4
    la $a0, odd_msg
    syscall

exit:
    li $v0, 10
    syscall
```

- Write a program that input a number from user displays each digit separately. For example if input is 3986, output is 6, 8, 9 ,3. (Loop)

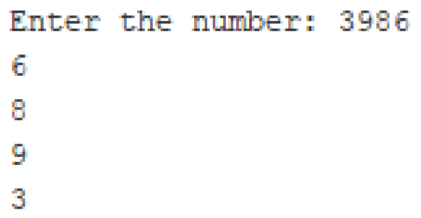
Code



```
.data
Num: .asciiz "Enter the number: "
newLine: .asciiz "\n"
.text
.globl main
main:
    li $v0, 4
    la $a0, Num
    syscall
    li $v0, 5
    syscall
    move $t0, $v0
```



```
loop:
    beq $t0, 0, done
    li $t1, 10
    div $t0, $t1
    mfhi $t2
    li $v0, 1
    move $a0, $t2
    syscall
    li $v0, 4
    la $a0, newLine
    syscall
    mflo $t0
    j loop
done:
li $v0, 10
syscall
```

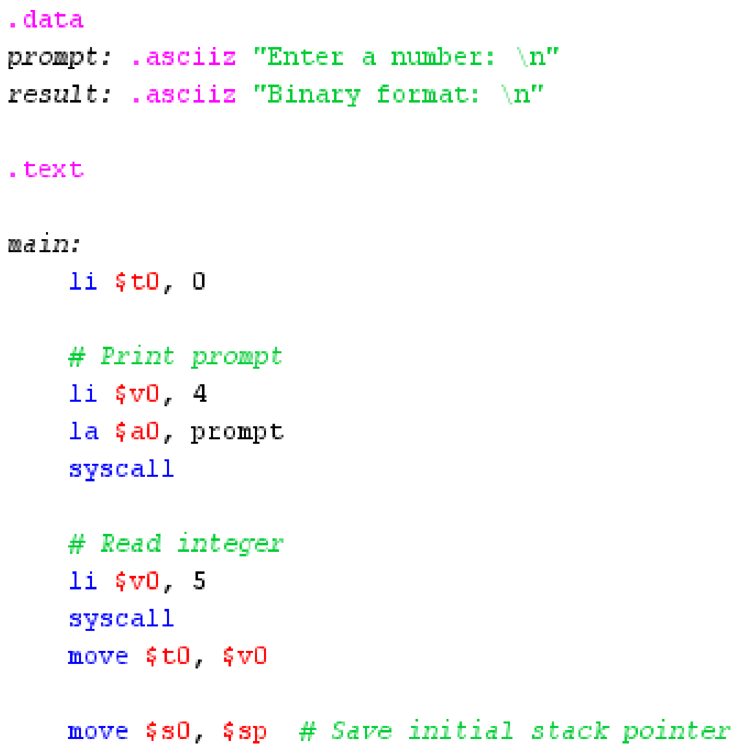


```
Enter the number: 3986
6
8
9
3
```

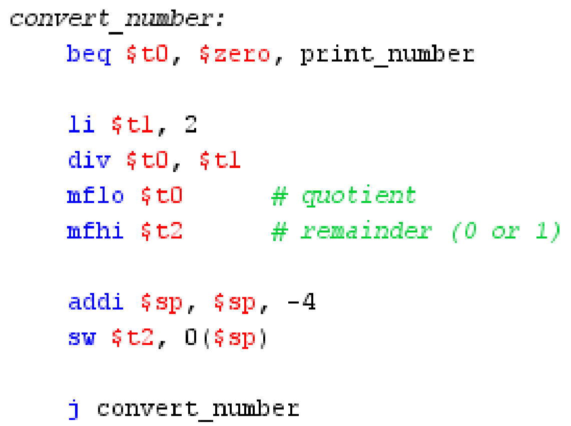
Output

- Write a program that converts a decimal number to binary. (Loop)

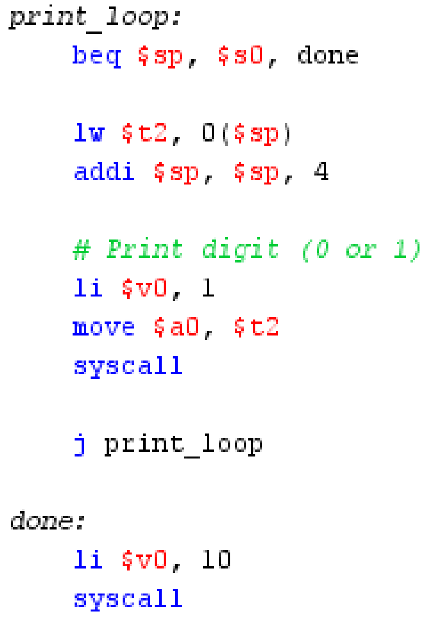
Code



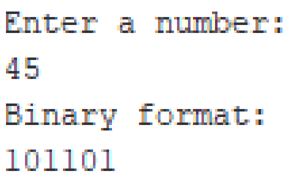
```
.data
prompt: .asciiz "Enter a number: \n"
result: .asciiz "Binary format: \n"

.text

main:
    li $t0, 0

    # Print prompt
    li $v0, 4
    la $a0, prompt
    syscall

    # Read integer
    li $v0, 5
    syscall
    move $t0, $v0

    move $s0, $sp  # Save initial stack pointer
```



```
convert_number:
    beq $t0, $zero, print_number

    li $t1, 2
    div $t0, $t1
    mflo $t0      # quotient
    mfhi $t2      # remainder (0 or 1)

    addi $sp, $sp, -4
    sw $t2, 0($sp)

    j convert_number
```



```
print_loop:
    beq $sp, $s0, done

    lw $t2, 0($sp)
    addi $sp, $sp, 4

    # Print digit (0 or 1)
    li $v0, 1
    move $a0, $t2
    syscall

    j print_loop

done:
    li $v0, 10
    syscall
```



```
Enter a number:
45
Binary format:
101101
```

Output

Critical Analysis

Task 1: Even/Odd

In this task I practiced using basic conditions. The program checks if a number is even or odd. It was an easy task but it helped me understand how to use if-else properly. I also learned that zero should be counted as even.

Task 2: Display digits separately

This task was about breaking a number into digits. I used division by 10 in a loop to get each digit. The problem I faced was that digits come out in reverse order, so I had to think of a way to fix that. Doing this made me more confident with loops.

Task 3: Decimal to Binary

Here I learned how to change a decimal number into binary. I used repeated division by 2 to get each bit. The main difficulty was printing the binary digits in the correct order. So, I used stack for this purpose. This task also showed me how numbers are stored in binary form inside the computer.

Conclusion

These three tasks taught me the basics step by step. The first task helped me with conditions, the second with loops, and the third with number conversion. Doing them myself cleared many small doubts I had about assembly programming.

Lab Assessment		
Pre Lab	**/5**	**/25**
Performance	**/5**	
Results	**/5**	
Viva	**/5**	
Critical Analysis	**/5**	
Instructor Signature and Comments		