King Fahd University of Petroleum and Minerals
Computer Engineering Department
College of Computing and Mathematics

COE 301: Computer Organization
Assignment 1, Term 251
Bit Shuffling using MIPS Assembly

Submission Guidelines:

- The due date is Saturday Oct. 4, 2025 Midnight.
- The assignment can be solved individually or collaboratively in groups of two students. Both team members should be declared in Gradescope when submitting the assignment.
- The submitted code should include the name(s) and KFUPM ID(s) of the student(s) who did the assignment.
- The submitted file should be renamed as “**assign1.asm**”
- Submit your assignment through Gradescope. Email submissions are not accepted.
- The autograder in Gradescope is designed to give you feedback and help you in debugging. Passing the autograder tests does not guarantee a full mark in the assignment as the tests are not comprehensive and does not test all cases. It is your responsibility to make sure that the submitted code is correct and can handle all cases.
- Late submission is not acceptable.
- Using any AI tool for coding or debugging is not allowed.

Objective:

The goal of this assignment is to use MIPS assembly language to implement some bit shuffling and manipulation.

Implementation

1. Prompt the user to enter 32-bit value in hexadecimal format. The input should be read as a string of hexadecimal digits and spaces only. You can assume that all entered characters are either valid hexadecimal digit (uppercase or lowercase characters) or spaces. You can also assume that the maximum length of the input is 10 characters.
2. Convert the entered value to from hexadecimal to binary format. Ignore all spaces and consider the least significant 32-bit (8 hexadecimal digits) only if more hex digits are entered. (see the examples below)
3. Perform the following bit shuffling\manipulations
   - Reveres order of bits

Input	b_0	b_1	b_2	b_3	b_4	b_5	b_6	b_7	b_8	b_9	…	b_{22}	b_{23}	b_{24}	b_{25}	b_{26}	b_{27}	b_{28}	b_{29}	b_{30}	b_{31}
Output	b_{31}	b_{30}	b_{29}	b_{28}	b_{27}	b_{26}	b_{25}	b_{24}	b_{23}	b_{22}	…	b_9	b_8	b_7	b_6	b_5	b_4	b_3	b_2	b_1	b_0

   - Reverse order of bytes.

Input	Byte$_3$	Byte$_2$	Byte$_1$	Byte$_0$
output	Byte$_0$	Byte$_1$	Byte$_2$	Byte$_3$

4. Convert the results of shuffling to hexadecimal and print them to the screen
5. Add comments to explain your code

Grading:

Your submission will be evaluated based on the following grade distribution:

- Read the input correctly and verify it is a valid hex value (20 pts)
- Convert the input to binary (10 pts)
- Perform the 1st bit shuffling (reverse order of bits) (30 pts)
- Perform the 2nd bit shuffling (reverse order of bits) (20 pts)
- Convert the results to hexadecimal and print the results (10 pts)
- Code organization and comments (10 pts)

Sample runs:

```
Enter a 32-bit value in hexadecimal format: AbCd 1234
The reverse order of bits is: 2C48B3D5
The reverse order of bytes is: 3412CDAB
```

```
Enter a 32-bit value in hexadecimal format: 1234
The reverse order of bits is: 2C480000
The reverse order of bytes is: 34120000
```

```
Enter a 32-bit value in hexadecimal format: aB00001234
The reverse order of bits is: 2C480000
The reverse order of bytes is: 34120000
```