Project 2 - Postfix Translator

Authors: Buğra Keser and Yusuf Anıl Yazıcı
Date: May 5, 2024
Description:

This program is a interpreter that is written in GNU assembly language which interprets a single line of postfix expression involving decimal quantities and outputs the equivalent RISC-V 32- bit machine language instructions.

Design:

The program consists of the following modules:

1. postfix_translator.s: Entry point of the program and contains the required labels to interpret the given line.

Implementation Details:

1. Checking End of the Input:
   - Terminates the program if end of the input is reached.

2. Pushing to the Stack:
   - The number is pushed to the top of stack.

3. Checking Space:
   - Checks if the character is space.
   - Increments the pointer, that is, evaluates the next character if the character is space.

4. Performing Operations:
   - Determines the operator is which one of these : ’+’, ‘-’, ‘*’, ‘^’, ‘&’, ‘|’.
   - Perform the operation in compliance with the operator according to the definitions.
   - Pops the very top of the two values from the stack and pushes the attained value to the stack.

5. Printing Machine Language Instructions:
   - First, prints the addi instructions for registers x1 and x2.
   - After that, prints the operation codes.

Challenges Encountered:

1. Evaluating Character By Character:
   - Pointer is set to the start of the input and it is incremented one by one after each evaluation.

2. Printing the Output:
   - Required output parts are predefined in the data section and they are changed in accord with the values, functions and operations.

Example Inputs/Outputs:

Input :

```
2 3 + 4 5 + *
```

Output:

```
000000000011 00000 000 00010 0010011
000000000010 00000 000 00001 0010011
0000000 00010 00001 000 00001 0110011
000000000101 00000 000 00010 0010011
000000000100 00000 000 00001 0010011
0000000 00010 00001 000 00001 0110011
000000001001 00000 000 00010 0010011
000000000101 00000 000 00001 0010011
0000001 00010 00001 000 00001 0110011
```

Input:

2 3 1 ^ & 9 -

Output:

```
000000000001 00000 000 00010 0010011
000000000011 00000 000 00001 0010011
0000100 00010 00001 000 00001 0110011
000000000010 00000 000 00010 0010011
000000000010 00000 000 00001 0010011
0000111 00010 00001 000 00001 0110011
000000001001 00000 000 00010 0010011
000000000010 00000 000 00001 0010011
0100000 00010 00001 000 00001 0110011
```

Input:

72 49 - 87 | 3 96 ^ 24 101 & * +

Output:

```
000000110001 00000 000 00010 0010011
000001001000 00000 000 00001 0010011
0100000 00010 00001 000 00001 0110011
000001010111 00000 000 00010 0010011
000000010111 00000 000 00001 0010011
0000110 00010 00001 000 00001 0110011
000001100000 00000 000 00010 0010011
000000000011 00000 000 00001 0010011
0000100 00010 00001 000 00001 0110011
000001100101 00000 000 00010 0010011
000000011000 00000 000 00001 0010011
0000111 00010 00001 000 00001 0110011
000000000000 00000 000 00010 0010011
000001100011 00000 000 00001 0010011
0000001 00010 00001 000 00001 0110011
000000000000 00000 000 00010 0010011
000001010111 00000 000 00001 0010011
0000000 00010 00001 000 00001 0110011
```

How to Use:

1. Compile the program in the terminal when you are in the src folder by the command:
   ```
   $ make
   ```
2. Run the executable by the command:
   ```
   $ ./postfix_translator
   ```
3. Enter a valid input.