#Team Member
#Qingyun Zhang 2095754
#Sicong Liu 2089129
#Ying Lu 2093558

The program specifically handles two integers, prints their values, addresses, and results of arithmetic operations, and concludes by safely exiting.

The program is structured into two main sections: the `.data` section for declaring variables and the `.text` section containing the executable instructions.

- Data Section

Variables Declared:
- `input1`: Initially set to 5.
- `input2`: Initially set to 3.

- Text Section

Global Label:
- `_start`: The entry point of the program.

- Operations and System Calls:

1. Print the Value of `input1`:
   - Load the address and value of `input1` into registers.
   - Use a system call to print the integer value.

2. Print a Newline Character:
   - Output a newline to separate outputs clearly.

3. Print the Address of `input1`:
   - Reload the address of `input1` and print it.

4. Print the Value of `input2`:
   - Similar to `input1`, load and print the value and address of `input2`.

5. Increment and Print New Values for `input1` and `input2`:
   - Add 1 to the values of `input1` and `input2`.
   - Store the new values back into their respective addresses.
   - Print these new incremented values.

6. Calculate and Print Sum of `input1 + 1` and `input2 + 1`:
   - Load the incremented values from memory, compute their sum, and print the result.

7. Store the Sum in `input1`'s Address and Print the Address:

- Store the computed sum back at the address of `input1`.
- Print the address where the sum has been stored.

8. Exit the Program:
   - Perform a system call to exit the program cleanly.

This RISC-V assembly program demonstrates fundamental assembly operations including loading and storing data, arithmetic calculations, and performing system calls for printing and exiting. The program efficiently uses system calls to interact with the system for printing integers, characters, and exiting. It also manipulates memory directly to store and retrieve values, showcasing typical low-level programming tasks in an assembly language.

- Screenshot