Lab 4 : RISC-V Simulator Report

Gagan Singla and Mayank Gupta

October 3, 2024

1 Basic Maps and Functions

We have used hash maps to create a mapping between the instruction with its correspoding instruction type and also to easily access registers through their aliases.

1.1 Basic mapping developed

1. instructType : Develop a mapping between instruction to its corresponding instruction type.
2. alias : Develop a mapping between alias of register to its standard naming format x0-x31.

1.2 Basic Arrays developed

1. registers : Constructed a long long array of 32 length which will serve as our registers.
2. mem : Constructed a array of vectors of length 0x40000 using bitset where each vector length is 8 which will serve as our memory.

1.3 Utility Functions used

1. twoComplement : To convert the input binary string into its two's complement.
2. convert_deci_to_binary : To convert the input decimal number into a specific length binary string.
3. getPC : To covert the long long interger value of PC to a string of PC.
4. getRegister : To extract the register number from string of register and also from its alias form.
5. unsignedComp : To compare two unsigned numbers (used in the case of bgeu/bltu).
6. We have also used some in-built functions like stoi(string to integer).

2 Input Handling

We are first finding what command are we required to perform and based on that we are performing our task.

2.1 Load Command

1. Upon getting load command, we are first initializing our initialising PC, registers and memory values to 0.
2. We are reading the whole input.s file and storing labels in a map named 'labels' and also check if that same label is not multiple times.

2.2 Run Command

1. We are performing every operation line by line and printing the PC of the line being executed.
2. Depending on the instruction type, we have made functions which will do their job of updating registers, memory ortransfering the PC.

2.3 Regs Command

Prints the value stored in the register in hexadecimal format

2.4 Exit Command

Gracefully exits the simulator

2.5 step Command

Runs a particular operation(depending on the position of program counter) by using instruction specific functions.

2.6 mem Command

Prints memory locations starting from the address in the data section.

2.7 Break Command

We are making a set containing the line number where we have to break the code while using run command. While using run command we check if our present line number is present in the set or not.If it is, we stop there.

2.8 del break Command

Removes that particular line from the set.

3 Executing Different Instructions

3.1 R Type Instruction

We are extracting operation and register number on the operation is gonna be performed and simply executing the operation.

3.2 I Type Instruction

3.2.1 Without Memory

We will extract the operation, register numbers and immediate value from the instruction and then convert immediate value's string into integer type and simply execute the operation.

3.2.2 With Memory

We will extract the memory location, source and output register number from the instruction and then based upon whether we have to load word, byte, halfword or doubleword, we will run the loop and load the data(in a chunk of 1 byte) from the memory into register.

3.3 S Type Instruction

We will extract the memory location, source and output register number from the instruction and then based upon whether we have to store word, byte, halfword or doubleword, we will run the loop and store the value of register into the memory.

3.4 B Type Instruction

We will extract the two registers whose value we have to compare and the immediate value which will decide PC depending on the condition. We will compare both registers value and we have also employed unsignedComp function for bgeu and bltu operation. Depending on the result,we will decide on whether to increase our PC by 4 or by immediate value.

3.5 J type Instruction

We will extract the register value and immediate value from the instruction and then simply update our PC by immediate value and keep the value of next line number in the register.

3.6 U type Instruction

We will extract the register value and immediate value from the instruction and then update the register value depending on the operation using PC and left shifting of the immediate value by 12 bits.

4 Assumptions

1.

5 Testing

1. We checked each operation one by one by covering corner cases and negative values also.
2. We checked both the codes given in the problem statement.