Readme : RISC-V Assembler

Praneeth Chamarthy && Gona Sanjana

October 3, 2024

1 Project Structure

```
project_root/
          Assembler/
          src/
                main.cpp
                simulator.cpp
                instruction.cpp
                memory.cpp
                register_file.cpp
          include/
                simulator.h
                instruction.h
                memory.h
                utils.h
                register_file.cpp
          tests/
                unit/
                      arithmetic.s
                      logical.s
                      shift.s
                integration/
                      bubblesort.s
                      fibonacci.s
                      function_calls.s
                edge_cases/
                    division.s
                    overflow.s
                    shifts.s
                Error Hanadling/
                    invalid_instructions.s
                    invalid_jumps.s
                    memory_access.s
          docs/
                report.pdf
          scripts/
                load_input.sh
          input/
                input.s
                input.hex
          Makefile
          README.md
```

2 File Descriptions

2.1 Source Files (src/)

- **main.cpp**: Main program file to run the simulator.
- **memory.cpp**: Implementation of memory management functions like read and write functions
- **register_file.cpp**: Implementation of register file functions used for printing the registers.
- **instruction.cpp**: Implementation of instruction decoding and execution.

2.2 Header Files (include/)

- **simulator.h**: Main Header file for the simulator class.
- **instruction.h**: Declaration used for instruction handling.
- **memory.h**: Declaration used for memory management.
- **register_file.h**: Declaration used for register printing,reading and writitng.

2.3 Assembler (Assembler/)

- **Assembler**: This sub directory is used to encode the given input.s file when we run load input.s command in the terminal console of my Simulator.

2.4 Test Files (tests/)

- `Unit tests`: Test individual instruction formats (e.g., register number lookup, R-type instruction parsing).
- `Integration tests:` Test complete simulator with data section lookup, label parsing and jumps.
- `Edge case tests`: Verify simulator behavior with overflow conditions and division by 0.
- `Error Handling tests`: Verifies the simulator behaviour when there is an error in the input file

3 Usage Instructions

1. **Compilation:**

```
    make

```

   This will compile the simulator and create the executable in the `bin/` directory and the created objected will be present in `obj` directory

2. **Running the Simulator:**

```
make run

```

   This command will execute the created executable file and displays user interactive simulator console

3. **Cleaning the Project:**

```
make clean

```

   This removes all generated files, including object files and the executable.

4 Input and Output

- Input: Place your RISC-V assembly code in `input/input.s`.
- Output: The simulator outputs the results of execution to the console, including register states and memory contents.

5 Testing

The `tests/` directory contains various test files:

- Unit tests validate individual components of the simulator.
- Integration tests check the simulator's performance on complete programs.
- Edge case tests ensure the simulator handles extreme scenarios correctly.