ARMish Processor

Karlo Godfrey Escalona Gregorio

Contents

1 2025/07

1.1 2025/07/12

- Created Github and documentation
- Finished ISA design

1.2 2025/07/13

- Updated ISA design section of documentation

1.3 2025/07/29

- Assembler Research
  - A two-pass assembler sounds simplest

1.4 2025/07/30

- The first-pass portion of the assembler is completed.
- Fixed some issues with the instruction set design
  - Dealing with immediates in the floating point instructions will be done by using fixed point immediates and the converting them to floating point numbers through an instruction.
  - Removed write-back and pre/post indexing bit options for D-type instructions
- TODO: Second-pass portion of the assembler

1.5 2025/08/03

- Updated assembler with formatting option
- Went into more detail for the instruction encoding for add/sub, mul/div, and mac instructions
- TODO: Finish documentation for the remaining instructions and second-pass assembler

1.6 2025/08/04

- Updated documentation detailing new instruction format for floating point types, and instructions that require 4 operands or more.
- Updated D-type instruction details
- TODO: Finish documentation for the remaining instructions and second-pass assembler

2 2025/08

2.1 2025/08/05

- Finished 50% of second-pass assembler. All instructions' most significant 10 bits can be assembled. NOT instruction can be fully assembled
- TODO: Finish second-pass assembler

2.2 2025/08/07

- Finished immediate encoding generation for ARM instructions
- First, a file containing every permutation of instructions was made to characterize the parser. Parsing instructions is done by tokenizing the instructions and placing the tokens into a list. The instructions are then divided based on the length of the list, which is then used to generate the machine code in a pinpoint manner.
- Documented assembler code
- TODO: Finish second-pass assembler

2.3 2025/08/09

- Finish second-pass assembler
- TODO: Instruction Memory, Program Counter, Register File