Assembler and Emulator Program Documentation

Author: G.Hema Vardhan Reddy (2301cs19)

Compilation and Execution Commands:

- Compilation: g++ assembler.cpp -o asm for the assembler, and `g++ emu.cpp -o emu for the emulator.
- Execution:
  - Assembler: ./asm test1.asm
  - Emulator: ./emu test5.o

Program Details

1. Assembler

The assembler performs both passes of the assembly process in a single routine, effectively managing common assembly requirements and ensuring robust error detection.

Key Features:

- Whitespace Management: Allows flexibility with spaces between operands and other whitespace, though labels require a space following the colon.
- Comment Handling: Supports both inline and standalone comments.
- Data Formats: Handles data in decimal, octal, and hexadecimal 2's complement.

Error Detection:

- Invalid Mnemonics: Detects when a mnemonic is not recognized.
- Label Validation:

  - Detects labels that are programming keywords.

  - Flags labels that don’t follow required naming conventions (underscores allowed initially, but not numerics or special symbols).

  - Identifies duplicate labels, missing labels, and unused labels.

- Operand Checks: Highlights missing operands, extra operands, and formatting issues with operands.

- Warnings: Detects potential numeric overflow and unused labels.

Generated Outputs:

- Tracks mnemonics, opcodes, and operand patterns.

- Records labels in use.

- Produces:

  - Log File: Contains any warnings and errors.

  - Listing File: Details byte codes produced for each instruction, alongside the mnemonic.

  - Object File: Machine-readable output, created only if no critical errors are found.

Additional Support: Can handle the pseudo-instruction SET.

---

 2. Emulator

The emulator can load, interpret, and execute the object files produced by the assembler.

Execution Options:

- Trace Mode (-t): Step-by-step execution of instructions.

- Memory Dump (-dump): Displays memory contents.

- All (-all): Executes all commands together.

Error Handling:

- Machine Code Validation: Detects incorrect machine code.
- Memory Management: Checks for segmentation faults and stack overflows.
- Opcode Validation: Flags invalid opcodes.

Output Information:

- Reports total instructions executed, memory dump in hexadecimal format, and other execution details.
- Allows runtime command changes.

Sample Usage and Outputs

Assembler Usage

Example 1: test1.asm

- Command: ./asm test1.asm
- Result: Compilation successful.
  - Log File: test1.log (contains only warnings).
  - Listing File: test1.l`
  - Object File: test1.o`
- Warnings: The label label is defined but not used.

Example 2: test2.asm

- Command:./asm test2.asm
- Result: Compilation failed due to errors.
  - Log File:test2.log (contains errors and warnings).
- Errors:
  - Duplicate label on line 4.
  - Label format issues (starting character or use of underscore) on line 10.
  - Undefined label reference on line 5.

  - Operand errors spanning lines 6 to 12.

- Files Not Created: Listing and object files were not generated due to errors.

Example 3:test3.asm

- Command: ./asm test3.asm
- Result: Compilation successful with no warnings.
  - Log File:test3.log (empty as no warnings were found).
  - Listing File:test3.l
  - Object File:test3.o

Emulator Usage

Example 1: test1.o

- Command:./emu test1.o
  - Trace Command (-t): Executes instructions step-by-step.
- Result:
  - Shows trace of each instruction execution:
    - ldc 00000000`: Registers set to A = 0, B = 0, PC = 1, SP = 0.
    - ldc FFFFFFFB`: Registers update to A = FFFFFFFB, B = 0, PC = 2, SP = 0.
    - The execution trace continues.

Example 3: test3.o

- Command:./emu test3.o
  - Trace Command (-t).
- Result:Emulator flags an invalid opcode error due to the misplaced SET instruction at the start. The program halts as expected.

Example 4:

Input: ./emu test4.o Output: Using -all Last few lines: A = 00000000, B = 00000000, PC = 00000008, SP = 00000FFF adj 00000001 A = 00000000, B = 00000000, PC = 00000009, SP = 00001000 HALT 00000000 Total instructions executed: 47654

Working as expected

Example 5:

Input: ./emu test5.o Output: -t for trace -dump for memory dump -all for all commands Enter with hyphen Emulator input: -t ldc 00000005 A = 00000005, B = 00000000, PC = 00000001, SP = 00000000 Emulator input: -t ldc FFFFFFFB A = FFFFFFFB, B = 00000005, PC = 00000002, SP = 00000000 Emulator input: -dump Base address: 0 No. of values: 12 00000000 00000500 FFFFFB00 00000012 00000000 00000004 00000000 00000000 00000000 00000000 00000008 00000000 00000000 00000000 00000000 Emulator input: -all HALT 00000000 Total instructions executed: 3

Working as expected. Memory dump correct.

Example 6:

Not possible to emulatate test6.asm as the assembly code is incorrect (so no object file).

Example 7:bubble.o

- Command:./emu bubble.o
- Result: Bubble sort simulation completes with final register states and instruction count.
  - Instructions Executed: 1044, indicating successful execution of the sorting routine.

Note:

Only programs without major errors generate the object and listing files, while programs with warnings proceed with output generation.