Project 3 Simple ARM-like Simulator

How to Run and How It Works

To run the simulator, first go to the project folder in your terminal and run make. Make will compile the program and create the executable I named as “sim”. By default, the simulator reads instructions from PP3_input.txt. If you want to use a different file, you can run ./sim <inputfile>. For example, running ./sim PP3_input.txt will execute the sample input file.

The simulator works by first reading the assembly input file and converting each line into an instruction object. The CPU object file then executes these instructions one by one. Each instruction updates the state of the registers, memory, and the NZCV flags. After executing each line, the program prints the instruction itself along with the updated CPU state, so you can follow exactly what happened.

Instruction Logic and Functionality

The simulator supports numerous instructions found in ARM assembly. Arithmetic instructions like ADD and SUB perform addition and subtraction between registers or between a register and an immediate value. If the instruction has the s suffix, it updates the NZCV flags, for example, N is set if the result is negative, Z is set if the result is zero, C is the carry-out from addition or borrow in subtraction, and V is set if overflow occurs.

Bitwise operations like AND, ORR, and EOR perform logical operations between values. The s suffix updates the N and Z flags, but the C and V flags do not change. Shift operations (LSL and LSR) shift a register value left or right. The s suffix also updates the carry flag so that the last bit shifted out.

Move instructions include MOV, which copies a value or register into another register, and MVN, which copies the bitwise NOT of a value. Memory operations include LDR and STR, which load and store values between registers and memory. Memory in the simulator is limited to five 32-bit words starting at address 0x100.

The compare instruction CMP subtracts one value from another without storing the result, but it updates the NZCV flags. Conditional instructions, like ADDEQ or BEQ, only execute if certain conditions are met based on the NZCV flags. For example, BEQ branches to a label only if the Z flag is set.

Test Cases

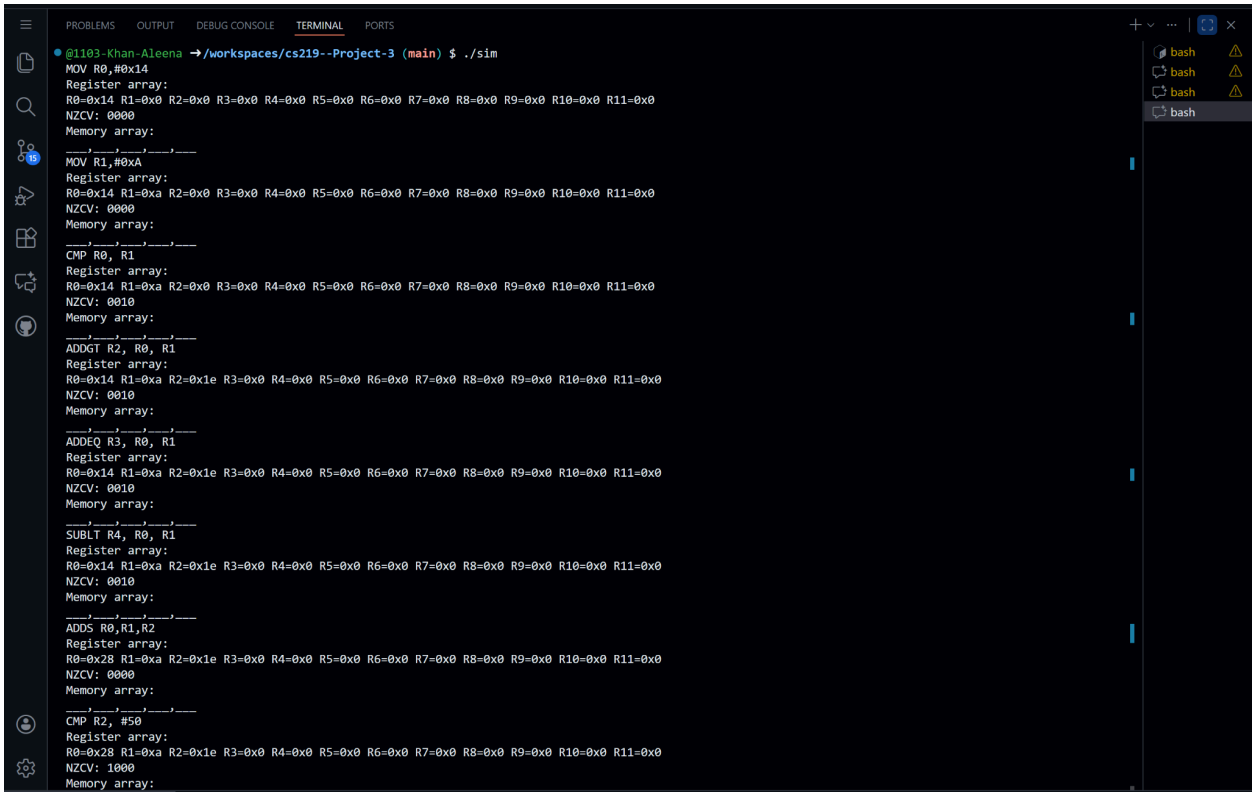
To make sure the simulator works, I tested several cases.

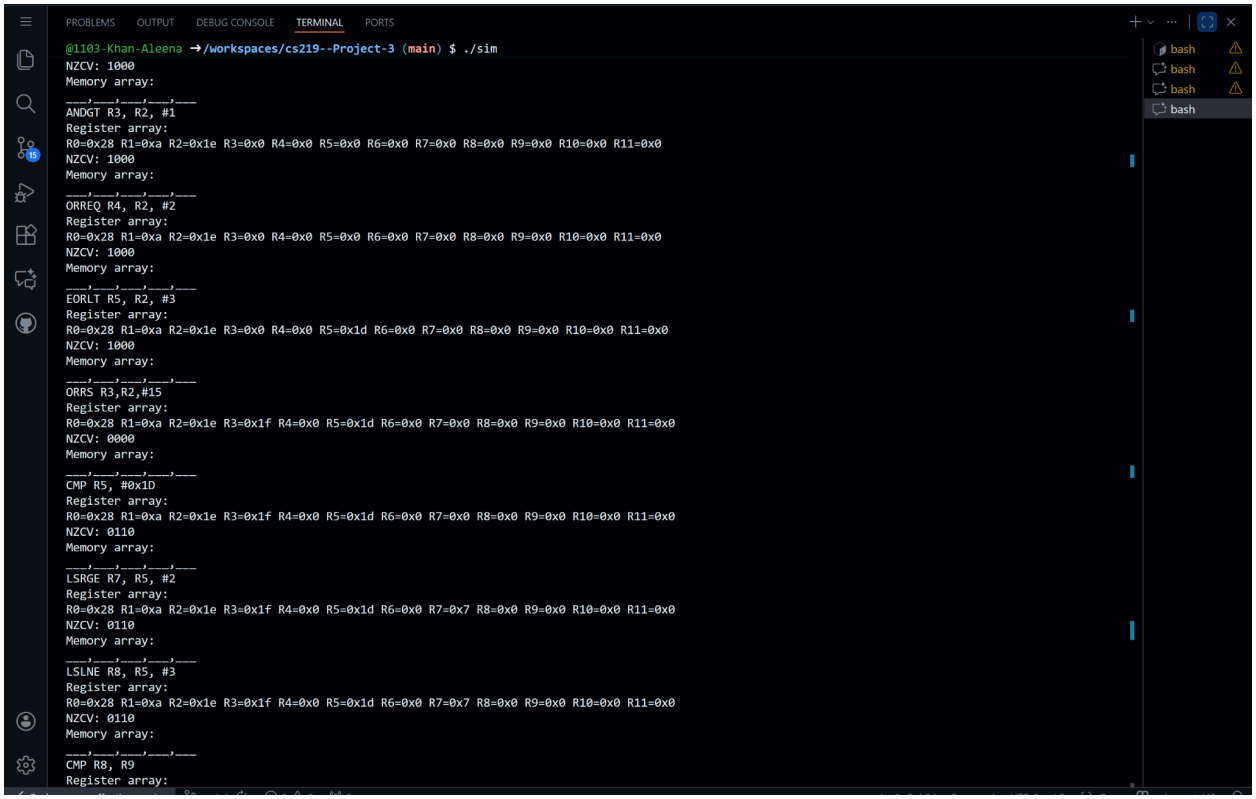
1. **MOV and ADD:** Moving 5 into R1 and 7 into R2, then adding them into R3 results in R3 = 12. NZCV flags remain 0 unless the S variant is used.
2. **CMP and BEQ:** Setting R1 to 5, then comparing R1 with 5 sets the Z flag. A BEQ branch to a label will be taken, so R0 ends up being 1.
3. **Memory operations:** Storing 42 in memory at address 0x100 and then loading it into R3 correctly sets mem[0] = 42 and R3 = 42.
4. **Shifts:** Shifting 1 left by 1 with LSL gives R2 = 2, and the carry flag is updated depending on the bit shifted out.

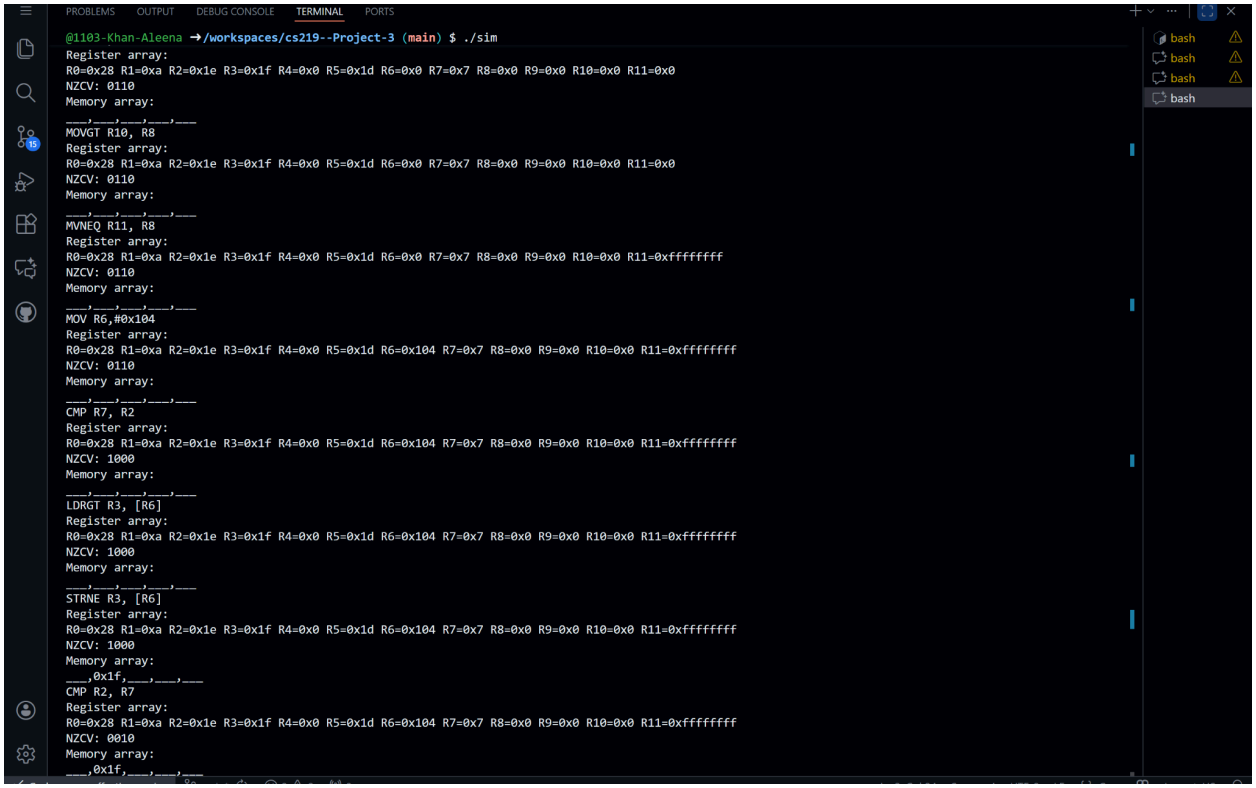
Invalid Cases

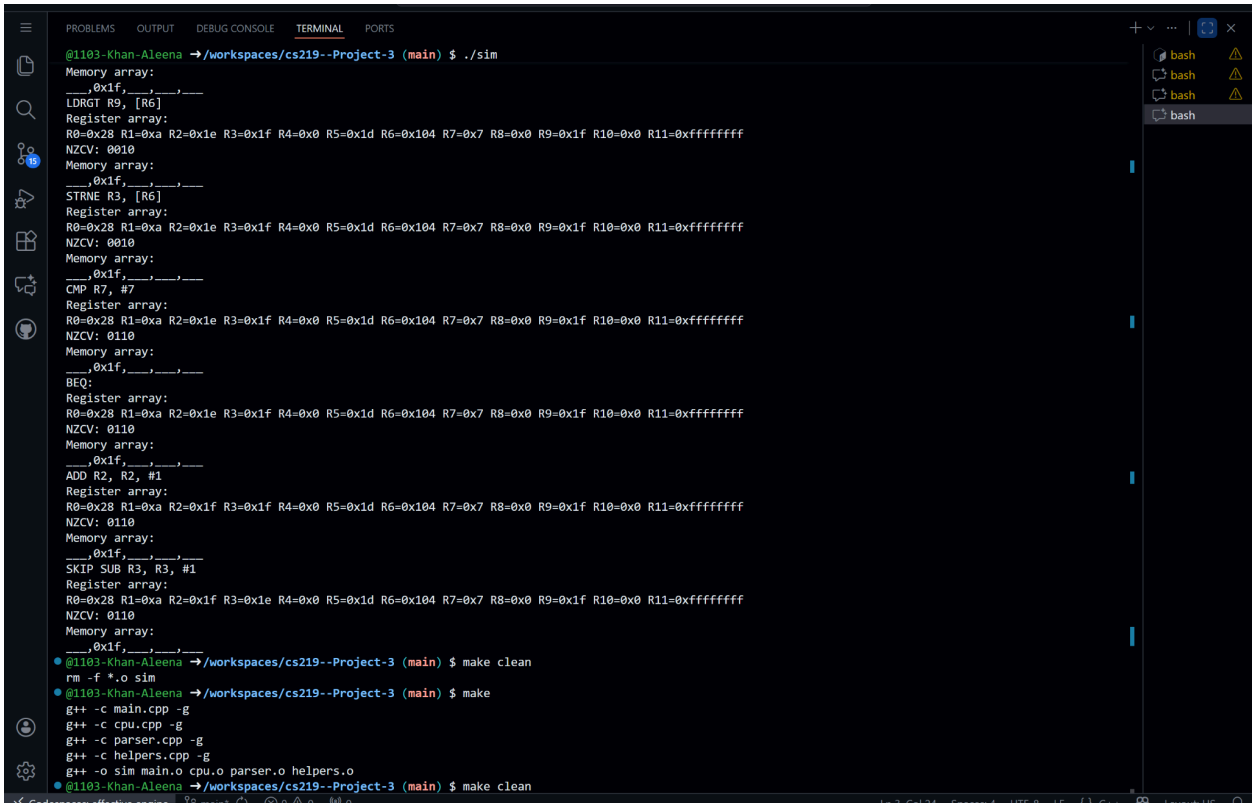
The simulator also handles invalid cases. For example, when trying to access memory that isn't properly aligned, like address 0x101, will not change memory or registers, but the simulator still prints the CPU state. Unknown opcodes are ignored, and invalid register names like R12 are ignored if used as a destination.

SCREENSHOTS OF OUTPUT IN ORDER:


PROBLEMS
OUTPUT
DEBUG CONSOLE
TERMINAL
PORTS
@1103-Khan-Aleena ➜/workspaces/cs219--Project-3 (main) $ ./sim
MOV R0,#0x14
Register array:
R0=0x14 R1=0x0 R2=0x0 R3=0x0 R4=0x0 R5=0x0 R6=0x0 R7=0x0 R8=0x0 R9=0x0 R10=0x0 R11=0x0
NZCV: 0000
Memory array:
___,___,___,___,___
MOV R1,#0xA
Register array:
R0=0x14 R1=0xa R2=0x0 R3=0x0 R4=0x0 R5=0x0 R6=0x0 R7=0x0 R8=0x0 R9=0x0 R10=0x0 R11=0x0
NZCV: 0000
Memory array:
___,___,___,___,___
CMP R0, R1
Register array:
R0=0x14 R1=0xa R2=0x0 R3=0x0 R4=0x0 R5=0x0 R6=0x0 R7=0x0 R8=0x0 R9=0x0 R10=0x0 R11=0x0
NZCV: 0010
Memory array:
___,___,___,___,___
ADDGT R2, R0, R1
Register array:
R0=0x14 R1=0xa R2=0x1e R3=0x0 R4=0x0 R5=0x0 R6=0x0 R7=0x0 R8=0x0 R9=0x0 R10=0x0 R11=0x0
NZCV: 0010
Memory array:
___,___,___,___,___
ADDEQ R3, R0, R1
Register array:
R0=0x14 R1=0xa R2=0x1e R3=0x0 R4=0x0 R5=0x0 R6=0x0 R7=0x0 R8=0x0 R9=0x0 R10=0x0 R11=0x0
NZCV: 0010
Memory array:
___,___,___,___,___
SUBLT R4, R0, R1
Register array:
R0=0x14 R1=0xa R2=0x1e R3=0x0 R4=0x0 R5=0x0 R6=0x0 R7=0x0 R8=0x0 R9=0x0 R10=0x0 R11=0x0
NZCV: 0010
Memory array:
___,___,___,___,___
ADDS R0,R1,R2
Register array:
R0=0x28 R1=0xa R2=0x1e R3=0x0 R4=0x0 R5=0x0 R6=0x0 R7=0x0 R8=0x0 R9=0x0 R10=0x0 R11=0x0
NZCV: 0000
Memory array:
___,___,___,___,___
CMP R2, #50
Register array:
R0=0x28 R1=0xa R2=0x1e R3=0x0 R4=0x0 R5=0x0 R6=0x0 R7=0x0 R8=0x0 R9=0x0 R10=0x0 R11=0x0
NZCV: 1000
Memory array:
bash
bash
bash
bash


PROBLEMS
OUTPUT
DEBUG CONSOLE
TERMINAL
PORTS
@1103-Khan-Aleena ➜/workspaces/cs219--Project-3 (main) $ ./sim
NZCV: 1000
Memory array:
___,___,___,___,___
ANDGT R3, R2, #1
Register array:
R0=0x28 R1=0xa R2=0x1e R3=0x0 R4=0x0 R5=0x0 R6=0x0 R7=0x0 R8=0x0 R9=0x0 R10=0x0 R11=0x0
NZCV: 1000
Memory array:
___,___,___,___,___
ORREQ R4, R2, #2
Register array:
R0=0x28 R1=0xa R2=0x1e R3=0x0 R4=0x0 R5=0x0 R6=0x0 R7=0x0 R8=0x0 R9=0x0 R10=0x0 R11=0x0
NZCV: 1000
Memory array:
___,___,___,___,___
EORLT R5, R2, #3
Register array:
R0=0x28 R1=0xa R2=0x1e R3=0x0 R4=0x0 R5=0x1d R6=0x0 R7=0x0 R8=0x0 R9=0x0 R10=0x0 R11=0x0
NZCV: 1000
Memory array:
___,___,___,___,___
ORRS R3,R2,#15
Register array:
R0=0x28 R1=0xa R2=0x1e R3=0x1f R4=0x0 R5=0x1d R6=0x0 R7=0x0 R8=0x0 R9=0x0 R10=0x0 R11=0x0
NZCV: 0000
Memory array:
___,___,___,___,___
CMP R5, #0x1D
Register array:
R0=0x28 R1=0xa R2=0x1e R3=0x1f R4=0x0 R5=0x1d R6=0x0 R7=0x0 R8=0x0 R9=0x0 R10=0x0 R11=0x0
NZCV: 0110
Memory array:
___,___,___,___,___
LSRGE R7, R5, #2
Register array:
R0=0x28 R1=0xa R2=0x1e R3=0x1f R4=0x0 R5=0x1d R6=0x0 R7=0x7 R8=0x0 R9=0x0 R10=0x0 R11=0x0
NZCV: 0110
Memory array:
___,___,___,___,___
LSLNE R8, R5, #3
Register array:
R0=0x28 R1=0xa R2=0x1e R3=0x1f R4=0x0 R5=0x1d R6=0x0 R7=0x7 R8=0x0 R9=0x0 R10=0x0 R11=0x0
NZCV: 0110
Memory array:
___,___,___,___,___
CMP R8, R9
Register array:
bash
bash
bash
bash


PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL PORTS
@1103-Khan-Aleena ➜/workspaces/cs219--Project-3 (main) $ ./sim
Register array:
R0=0x28 R1=0xa R2=0x1e R3=0x1f R4=0x0 R5=0x1d R6=0x0 R7=0x7 R8=0x0 R9=0x0 R10=0x0 R11=0x0
NZCV: 0110
Memory array:
___,___,___,___,___
MOVGT R10, R8
Register array:
R0=0x28 R1=0xa R2=0x1e R3=0x1f R4=0x0 R5=0x1d R6=0x0 R7=0x7 R8=0x0 R9=0x0 R10=0x0 R11=0x0
NZCV: 0110
Memory array:
___,___,___,___,___
MVNEQ R11, R8
Register array:
R0=0x28 R1=0xa R2=0x1e R3=0x1f R4=0x0 R5=0x1d R6=0x0 R7=0x7 R8=0x0 R9=0x0 R10=0x0 R11=0xffffffff
NZCV: 0110
Memory array:
___,___,___,___,___
MOV R6,#0x104
Register array:
R0=0x28 R1=0xa R2=0x1e R3=0x1f R4=0x0 R5=0x1d R6=0x104 R7=0x7 R8=0x0 R9=0x0 R10=0x0 R11=0xffffffff
NZCV: 0110
Memory array:
___,___,___,___,___
CMP R7, R2
Register array:
R0=0x28 R1=0xa R2=0x1e R3=0x1f R4=0x0 R5=0x1d R6=0x104 R7=0x7 R8=0x0 R9=0x0 R10=0x0 R11=0xffffffff
NZCV: 1000
Memory array:
___,___,___,___,___
LDRGT R3, [R6]
Register array:
R0=0x28 R1=0xa R2=0x1e R3=0x1f R4=0x0 R5=0x1d R6=0x104 R7=0x7 R8=0x0 R9=0x0 R10=0x0 R11=0xffffffff
NZCV: 1000
Memory array:
___,___,___,___,___
STRNE R3, [R6]
Register array:
R0=0x28 R1=0xa R2=0x1e R3=0x1f R4=0x0 R5=0x1d R6=0x104 R7=0x7 R8=0x0 R9=0x0 R10=0x0 R11=0xffffffff
NZCV: 1000
Memory array:
___,0x1f,___,___,___
CMP R2, R7
Register array:
R0=0x28 R1=0xa R2=0x1e R3=0x1f R4=0x0 R5=0x1d R6=0x104 R7=0x7 R8=0x0 R9=0x0 R10=0x0 R11=0xffffffff
NZCV: 0010
Memory array:
___,0x1f,___,___,___
bash
bash
bash
bash


PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL PORTS
@1103-Khan-Aleena ➜/workspaces/cs219--Project-3 (main) $ ./sim
Memory array:
___,0x1f,___,___,___
LDRGT R9, [R6]
Register array:
R0=0x28 R1=0xa R2=0x1e R3=0x1f R4=0x0 R5=0x1d R6=0x104 R7=0x7 R8=0x0 R9=0x1f R10=0x0 R11=0xffffffff
NZCV: 0010
Memory array:
___,0x1f,___,___,___
STRNE R3, [R6]
Register array:
R0=0x28 R1=0xa R2=0x1e R3=0x1f R4=0x0 R5=0x1d R6=0x104 R7=0x7 R8=0x0 R9=0x1f R10=0x0 R11=0xffffffff
NZCV: 0010
Memory array:
___,0x1f,___,___,___
CMP R7, #7
Register array:
R0=0x28 R1=0xa R2=0x1e R3=0x1f R4=0x0 R5=0x1d R6=0x104 R7=0x7 R8=0x0 R9=0x1f R10=0x0 R11=0xffffffff
NZCV: 0110
Memory array:
___,0x1f,___,___,___
BEQ:
Register array:
R0=0x28 R1=0xa R2=0x1e R3=0x1f R4=0x0 R5=0x1d R6=0x104 R7=0x7 R8=0x0 R9=0x1f R10=0x0 R11=0xffffffff
NZCV: 0110
Memory array:
___,0x1f,___,___,___
ADD R2, R2, #1
Register array:
R0=0x28 R1=0xa R2=0x1f R3=0x1f R4=0x0 R5=0x1d R6=0x104 R7=0x7 R8=0x0 R9=0x1f R10=0x0 R11=0xffffffff
NZCV: 0110
Memory array:
___,0x1f,___,___,___
SKIP SUB R3, R3, #1
Register array:
R0=0x28 R1=0xa R2=0x1f R3=0x1e R4=0x0 R5=0x1d R6=0x104 R7=0x7 R8=0x0 R9=0x1f R10=0x0 R11=0xffffffff
NZCV: 0110
Memory array:
___,0x1f,___,___,___
@1103-Khan-Aleena ➜/workspaces/cs219--Project-3 (main) $ make clean
rm -f *.o sim
@1103-Khan-Aleena ➜/workspaces/cs219--Project-3 (main) $ make
g++ -c main.cpp -g
g++ -c cpu.cpp -g
g++ -c parser.cpp -g
g++ -c helpers.cpp -g
g++ -o sim main.o cpu.o parser.o helpers.o
@1103-Khan-Aleena ➜/workspaces/cs219--Project-3 (main) $ make clean
bash
bash
bash
bash