1.

// Execution begins at address 0

.pos 0

Init: irmovl Stack, %esp // Set up stack pointer

irmovl Stack, %ebp // Set up base pointer

call Main // Execute main program

halt // Terminate program

// The stack starts here and grows to lower addresses

.pos 0x100

Stack:

Main:

pushl %ebp

rrmovl %esp, %ebp

irmovl $5, %ebx //a = 5

pushl %ebx

irmovl $6, %edx //b = 6

pushl %edx

call g //g(a,b) = f(5,6)

rrmovl %ebp, %esp

popl %ebp

ret

f:

pushl %ebp

rrmovl %esp, %ebp

mrmovl 8(%ebp), %ecx //get x

rrmovl %ecx, %eax //result = x

addl %ecx, %ecx // %ecx = 2x

addl %ecx, %eax //result = 2x+x=3\*x

rrmovl %ebp, %esp

popl %ebp

ret

g:

//a is at 8+ebp ,b is at 12+ebp

pushl %ebp

rrmovl %esp, %ebp

mrmovl 8(%ebp), %edx

mrmovl 12(%ebp), %ebx

pushl %edx

call f

rrmovl %eax, %edx //edx =f(a)

pushl %ebx

call f

addl %edx, %eax //eax = f(a)+f(b)

rrmovl %ebp, %esp

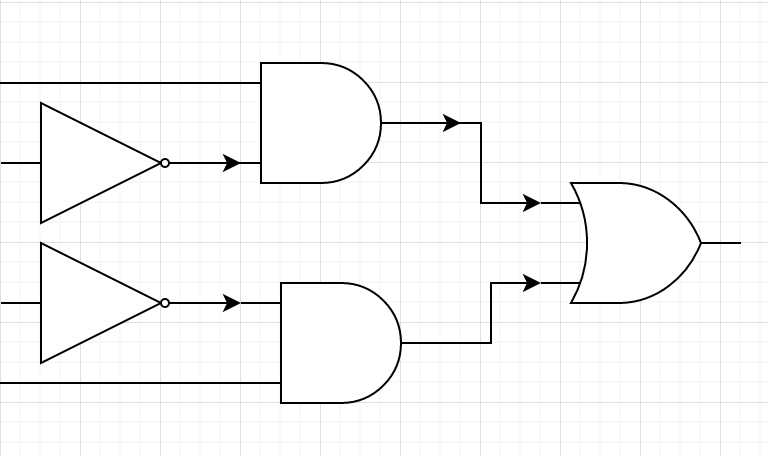
popl %edx

popl %ebx // clean up the stack

popl %ebp

ret

2. Inputs on the left from high to low: A, B, C, D.



Output is on the right.

3.

Fetch icode:ifun 🡨M1[PC] = C:0

rA:rB 🡨M1[PC+1]

valC 🡨M4[PC+2]

valP 🡨PC+6

Decode valB 🡨R[rB]

Execute valE 🡨 valB+valC

Set CC

Memory

Write back R[rB] 🡨 valE

PC update PC 🡨 valP