1.1

Stage	**Generic**	**Specific**
	rrmovl rA, rB	rrmovl %esp,%ebp
Fetch	icode : ifun ← M1[PC] rA:rB←M1[PC+1] valP←PC+2	icode : ifun ← M1[0x044] = 2 : 0 rA:rB←M1[0x045] = 4 : 5 valP←0x044+2 = 0x046
Decode	valA ← R[rA]	valA ← R[%esp] = 128
Execute	valE ← 0 + valA	valE ← 128
Memory		
Write Back	R[rB] ← valE	R[%ebp] ← 128
Update PC	PC ← valP	PC ← 0x046

1.2

Stage	**Generic**	**Specific**
	OPl rA, rB	xorl %eax,%eax
Fetch	icode : ifun ← M1[PC] rA:rB←M1[PC+1] valP←PC+2	icode : ifun ← M1[0x052] = 6 : 3 rA:rB ← M1[0x053] = 0 : 0 valP ← PC+2 = 0x054
Decode	valA ← R[rA] valB ← R[rB]	valA ← R[%eax] = 1 valB ← R[%eax] = 1
Execute	valE←valB OP valA Set CC	valE ← 1 ^ 1 = 0 ZF ← 1, SF ← 0, OF ← 0
Memory		
Write Back	R[rB] ← valE	R[%eax] ← 0
Update PC	PC ← valP	PC ← 0x054

1.3

Stage	**Generic**	**Specific**
	irmovl V, rB	irmovl $4,%ebx
Fetch	icode : ifun ← M1[PC] rA:rB ← M1[PC+1] valC ← M4[PC+2] valP ← PC+6	icode : ifun ← M1[0x063] = 3 : 0 rA:rB ← M1[0x06c] = f : 3 valC ← M4[0x06d] =4 valP ← 0x063+6 = 0x069
Decode		
Execute	valE←0+valC	valE←0+4 = 4
Memory		
Write Back	R[rB] ← valE	R[%ebx] ←4
Update PC	PC ← valP	PC ← 0x069

1.4

Stage	**Generic**	**Specific**
	OPl rA, rB	addl %ebx,%ecx
Fetch	icode : ifun ← M1[PC] rA:rB ← M1[PC+1] valP ← PC+2	icode : ifun ← M1[0x069] = 6 : 0 rA:rB ← M1[0x06a] = 3 : 1 valP ← 0x069+2 = 0x06b
Decode	valA ← R[rA] valB ← R[rB]	valA ← R[%ebx] =4 valB ← R[%ecx] =256
Execute	valE←valB OP valA Set CC	valE ← 256 + 4 = 260 ZF ← 0, SF ← 0, OF ← 0
Memory		
Write Back	R[rB] ← valE	R[%ecx] ← 260
Update PC	PC ← valP	PC ← 0x06b

2.

Stage	leave
Fetch	icode: ifun ← M1[PC] valP ← PC+1
Decode	valA ← R[%ebp] valB ← R[%ebp]
Execute	valE ← valB+4
Memory	valM ← M4[valA]
Write Back	R[%esp] ← valE R[%ebp] ← valM
Update PC	PC ← valP