

Y86 Instruction Set Reference

Instruction	Byte offset from PC									Instruction	Byte offset from PC										
	0	1	2	3	4	5	6	7	8		9	0	1	2	3	4	5	6	7	8	9
halt	0	0									OPq rA, rB	6	fn	rA	rB						
nop	1	0									jXX Dest	7	fn	Dest							
cmovXX rA, rB	2	fn	rA	rB							call Dest	8	0	Dest							
irmovq V, rB	3	0	f	rB					V		ret	9	0								
rmmovq rA, D(rB)	4	0	rA	rB					D		pushq rA	a	0	rA	f						
mrmovq D(rB), rA	5	0	rA	rB					D		popq rA	b	0	rA	f						

cmovXX:			
rrmovq	20	cmovne	24
cmovle	21	cmovge	25
cmovl	22	cmovg	26
cmove	23		

OPq:	
addq	60
subq	61
andq	62
xorq	63

jXX:			
jmp	70	jne	74
jle	71	jge	75
jl	72	fg	76
je	73		

Registers:		Args:	
%rax ⁺	0	%rsp	4
%rcx ⁺	1	%rbp*	5
%rdx ⁺	2	%rsi ⁺	6
%rbx*	3	%rdi ⁺	7

⁺caller-save *callee-save

Stage	HALT	NOP	CMOV	IRMOVQ
Fch	icode ← M ₁ [PC] valP ← PC + 1	icode ← M ₁ [PC] valP ← PC + 1	icode:ifun ← M ₁ [PC] rA:rB ← M ₁ [PC+1] valP ← PC + 2	icode:ifun ← M ₁ [PC] rA:rB ← M ₁ [PC+1] valC ← M ₈ [PC+2] valP ← PC + 10
Dec			valA ← R[rA]	
Exe	cpu.stat = HLT		valE ← valA Cnd ← Cond(CC,ifun)	valE ← valC
Mem				
WB			Cnd ? R[rB] ← valE	R[rB] ← valE
PC	PC ← 0	PC ← valP	PC ← valP	PC ← valP
Stage	RMMOVQ	MRMOVQ	OPq	JUMP
Fch	icode:ifun ← M ₁ [PC] rA:rB ← M ₁ [PC+1] valC ← M ₈ [PC+2] valP ← PC + 10	icode:ifun ← M ₁ [PC] rA:rB ← M ₁ [PC+1] valC ← M ₈ [PC+2] valP ← PC + 10	icode:ifun ← M ₁ [PC] rA:rB ← M ₁ [PC+1] valP ← PC + 2	icode:ifun ← M ₁ [PC] valC ← M ₈ [PC+1] valP ← PC + 9
Dec	valA ← R[rA] valB ← R[rB]	valB ← R[rB]	valA ← R[rA] valB ← R[rB]	
Exe	valE ← valB + valC	valE ← valB + valC	valE ← valB OP valA	Cnd ← Cond(CC,ifun)
Mem	M ₈ [valE] ← valA	valM ← M ₈ [valE]		
WB		R[rA] ← valM	R[rB] ← valE	
PC	PC ← valP	PC ← valP	PC ← valP	PC ← Cnd?valC:valP
Stage	CALL	RET	PUSHQ	POPQ
Fch	icode:ifun ← M ₁ [PC] valC ← M ₈ [PC+1] valP ← PC + 9	icode:ifun ← M ₁ [PC] valP ← PC + 1	icode:ifun ← M ₁ [PC] rA:rB ← M ₁ [PC+1] valP ← PC + 2	icode:ifun ← M ₁ [PC] rA:rB ← M ₁ [PC+1] valP ← PC + 2
Dec	valB ← R[RSP]	valA ← R[RSP] valB ← R[RSP]	valA ← R[rA] valB ← R[RSP]	valA ← R[RSP] valB ← R[RSP]
Exe	valE ← valB - 8	valE ← valB + 8	valE ← valB - 8	valE ← valB + 8
Mem	M ₈ [valE] ← valP	valM ← M ₈ [valA]	M ₈ [valE] ← valA	valM ← M ₈ [valA]
WB	R[RSP] ← valE	R[RSP] ← valE	R[RSP] ← valE	R[RSP] ← valE R[rA] ← valM
PC	PC ← valC	PC ← valM	PC ← valP	PC ← valP