

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

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

OPq:	
addq	60
subq	61
andq	62
xorq	63

jXX:	
jmp	70
jle	71
j1	72
je	73
jne	74
jge	75
jpg	76

Trap IDs:	
charout	0
charin	1
decout	2
decin	3
strout	4
flush	5

Registers:		
%rax ⁺	0	%rbp [*] 5
%rcx ⁺	1	%rsi ⁺ 6
%rdx ⁺	2	%rdi ⁺ 7
%rbx [*]	3	%r8-%r11 ⁺
%rsp	4	%r12-%r14 [*]
⁺ caller-save [*] callee-save		

Args:	
%rdi	
%rsi	
%rdx	
%rcx	
%r8	
%r9	

Stage	HALT	NOP	cmovXX	IRMOVQ
Fch	icode:ifun ← M ₁ [PC]	icode:ifun ← M ₁ [PC]	icode:ifun ← M ₁ [PC] rA:rB ← M ₁ [PC+1]	icode:ifun ← M ₁ [PC] rA:rB ← M ₁ [PC+1] valC ← M ₈ [PC+2] valP ← PC + 10
Dec	valP ← PC + 1	valP ← PC + 1	valP ← PC + 2 valA ← R[rA]	valP ← PC + 10
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	jXX
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 Set CC	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
PC	PC ← valC	PC ← valM	PC ← valP	PC ← valP