第三章 汇编代码

数据格式

C声明	Intel数据类型	汇编代码后缀	大小(字节)
`char`	字节	b	1
`short`	字	w	2
`int`	双字	l	4
`long`	四字	q	8
`char *`	四字	q	8
`float`	单精度	s	4
`double`	双精度	l	8

整数寄存器

64	32	16	8	访问信息
`%rax`	`%eax`	`%ax`	`%al`	返回值
`%rbx`	`%ebx`	`%bx`	`%bi`	被调用者保存
`%rcx`	`%ecx`	`%cx`	`%cl`	第4个参数
`%rdx`	`%edx`	`%dx`	`%dl`	第3个参数
`%rsi`	`%esi`	`%si`	`%sil`	第2个参数
`%rdi`	`%edi`	`%di`	`%dil`	第1个参数
`%rbp`	`%ebp`	`%bp`	`%bpl`	被调用者保存
`%rsp`	`%esp`	`%sp`	`%spl`	栈指针
`%r8`	`%r8d`	`%r8w`	`%r8b`	第5个参数
`%r9`	`%r9d`	`%r9w`	`%r9b`	第6个参数
`%r10`	`%r10d`	`%r10w`	`%r10b`	调用者保存
`%r11`	`%r11d`	`%r11w`	`%r11b`	调用者保存
`%r12`	`%r12d`	`%r12w`	`%r12b`	被调用者保存
`%r13`	`%r13d`	`%r13w`	`%r13b`	被调用者保存
`%r14`	`%r14d`	`%r14w`	`%r14b`	被调用者保存
`%r15`	`%r15d`	`%r15w`	`%r15b`	被调用者保存

操作数格式

类型	格式	操作数值	名称
立即数	$\$Imm$	Imm	立即数寻址
寄存器	r_a	$R[r_a]$	寄存器寻址
存储器	Imm	$M[Imm]$	绝对寻址
存储器	(r_a)	$M[R[r_a]]$	间接寻址
存储器	$Imm(r_b)$	$M[Imm + R[r_b]]$	（基址+偏移量）寻址
存储器	$(r_b,\ r_i)$	$M[R[r_b] + R[r_i]]$	变址寻址
存储器	$Imm(r_b,\ r_i)$	$M[Imm + R[r_b] + R[r_i]]$	变址寻址
存储器	$(,\ r_i,\ s)$	$M[R[r_i] \cdot s]$	比例变址寻址
存储器	$Imm(,\ r_i,\ s)$	$M[Imm + R[r_i] \cdot s]$	比例变址寻址
存储器	$(r_b,\ r_i,\ s)$	$M[R[r_b] + R[i] \cdot s]$	比例变址寻址
存储器	$Imm(r_b,\ r_i,\ s)$	$M[Imm + R[r_b] + R[r_i] \cdot s]$	比例变址寻址

数据传送指令

简单数据传送指令

指令		效果	描述
MOV	$S,\ D$	$D \leftarrow S$	**传送**
`movb`			传送字节
`movw`			传送字
`movl`			传送双字
`movq`			传送四字
`movabsq`	$I,\ R$	$R \leftarrow I$	传送绝对的四字

零扩展数据传送指令

指令	效果	描述
MOVZ S, R	$R \leftarrow$零扩展(S)	**以零扩展进行传送**
`movzbw`		将做了零扩展的字节传送到字
`movzbl`		将做了零扩展的字节传送到双字
`movzwl`		将做了零扩展的字传送到双字
`movzbq`		将做了零扩展的字节传送到四字
`movzwq`		将做了零扩展的字传送到四字

符号扩展数据传送指令

指令	效果	描述
MOVS S, R	$R \leftarrow$符号扩展(S)	**传送符号扩展的字节**
`movsbw`		将做了符号扩展的字节传送到字
`movsbl`		将做了符号扩展的字节传送到双字
`movswl`		将做了符号扩展的字传送到双字
`movsbq`		将做了符号扩展的字节传送到四字
`movswq`		将做了符号扩展的字传送到四字
`movslq`		将做了符号扩展的双字传送到四字
`cltq`	`%rax` ←符号扩展(`%eax`)	把 `%eax` 符号扩展到 `%rax`

入栈和出栈指令

指令	效果	描述
`pushq` S	`R[%rsp]` ← `R[%rsp] - 8` ;	把四字压入栈
	`M[R[%rsp]]` ← S	
`popq` D	D ← `M[R[%rsp]]` ;	将四字弹出栈
	`R[%rsp]` ← `R[%rsp] + 8`	

算数操作

整数算术操作

指令	效果	描述
`leaq S, D`	$D \leftarrow \&S$	加载有效地址
`INC D`	$D \leftarrow D + 1$	加1
`NEC D`	$D \leftarrow D - 1$	减1
`NEG D`	$D \leftarrow -D$	取负
`NOT D`	$D \leftarrow \ D$	取补
`ADD S, D`	$D \leftarrow D + S$	加
`SUB S, D`	$D \leftarrow D - S$	减
`IMUL S, D`	$D \leftarrow D * S$	乘
`XOR S, D`	$D \leftarrow D \hat{} S$	异或
`OR S, D`	$D \leftarrow D \mid 1$	或
`AND S, D`	$D \leftarrow D \& 1$	与
`SAL k, D`	$D \leftarrow D << k$	左移
`SHL k, D`	$D \leftarrow D << k$	左移（等同于SAL）
`SAR K, D`	$D \leftarrow D >>_A k$	算术右移
`SHR k, D`	$D \leftarrow D >>_L k$	逻辑右移

特殊的算数操作

指令	效果	描述
`imulq S`	`R[%rax]` : `R[%rax]` $\leftarrow S \times$ `R[%rax]`	有符号全乘法
`mulq S`	`R[%rax]` : `R[%rax]` $\leftarrow S \times$ `R[%rax]`	无符号全乘法
`clto`	`R[%rax]` : `R[%rax]` $\leftarrow$(符号扩展) `R[%rax]`	转换为八字
`idivq S`	`R[%rax]` $\leftarrow$ `R[%rax]` : `R[%rax]` $\mod S$	有符号除法
`divq S`	`R[%rax]` $\leftarrow$ `R[%rax]` : `R[%rax]` $\div S$	无符号除法

条件码

条件码	效果(`t = a + b`)	描述
`CF`	`(unsigned) t < (unsigned) a`	进位标志。检查无符号溢出
`ZF`	`t == 0`	零标志。零
`SF`	`t < 0`	符号标志。负数
`OF`	`a < 0 == b < 0` `&& (t < 0 != a < 0)`	溢出标志。有符号溢出

设置条件码

指令	基于	描述
CMP $S_1,\ S_2$	$S_2 - S_1$	比较
`cmpb`		比较字节
`cmpw`		比较字
`cmpl`		比较双字
`cmpq`		比较四字
TEST S_1, S_2	$S_1 \& S_2$	测试
`testb`		测试字节
`testw`		测试字
`testl`		测试双字
`testq`		测试四字

访问条件码

指令	同义名	效果	设置条件
`sete D`	`setz`	$D \leftarrow$ `ZF`	相等/零
`setne D`	`setnz`	$D \leftarrow$ `~ZF`	不等/非零
`sets D`		$D \leftarrow$ `SF`	负数
`setns D`		$D \leftarrow$ `~SF`	非负数
`setg D`	`setnle`	$D \leftarrow$ `~(SF ^ OF) & ~ZF`	大于(>)
`setge D`	`setnl`	$D \leftarrow$ `~(SF ^ OF)`	大于等于(≥)
`setl D`	`setnge`	$D \leftarrow$ `SF ^ OF`	小于(<)
`setle D`	`setng`	$D \leftarrow$ `(SF ^ OF) | ZF`	小于等于(≤)
`seta D`	`setnbe`	$D \leftarrow$ `~CF & ~ZF`	超过(>)
`setae D`	`setnb`	$D \leftarrow$ `~CF`	超过或相等(≥)
`setb D`	`setnae`	$D \leftarrow$ `CF`	低于(<)
`setbe D`	`setna`	$D \leftarrow$ `CF | ZF`	低于或相等(≤)

条件跳转指令

指令	同义名	跳转条件	描述
`jmp Label`		`1`	直接跳转
`jmp *Operand`		`1`	间接跳转
`je Label`	`jz`	`ZF`	相等/零
`jne Label`	`jnz`	`~ZF`	不相等/非零
`js Label`		`SF`	负数
`jns Label`		`~SF`	非负数
`jg Label`	`jnle`	`~(SF ^ OF) & ~ZF`	大于(>)
`jge Label`	`jnl`	`~(SF ^ OF)`	大于等于(≥)
`jl Label`	`jnge`	`SF ^ OF`	小于(<)
`jle Label`	`jng`	`(SF ^ OF) | ZF`	小于等于(≤)
`ja Label`	`jnbe`	`~CF & ~ZF`	超过(>)
`jae Label`	`jnb`	`~CF`	超过或相等(≥)
`jb Label`	`jnae`	`CF`	低于(<)
`jbe Label`	`jna`	`CF | ZF`	低于或相等(≤)

条件传送指令

指令	同义名	跳转条件	描述
`cmove S, R`	`comvz`	`ZF`	相等/零
`cmovne S, R`	`comvnz`	`~ZF`	不相等/非零
`cmovs S, R`		`SF`	负数
`cmovns S, R`		`~SF`	非负数
`cmovg S, R`	`cmovnle`	`~(SF ^ OF) & ~ZF`	大于(>)
`cmovgn S, R`	`cmovnl`	`~(SF ^ OF)`	大于等于(≥)
`cmovl S, R`	`cmovnge`	`SF ^ OF`	小于(<)
`cmovle S, R`	`cmovng`	`(SF ^ OF) | ZF`	小于等于(≤)
`cmova S, R`	`cmovnbe`	`~CF & ~ZF`	超过(>)
`cmovae S, R`	`cmovnb`	`~CF`	超过或相等(≥)
`cmovb S, R`	`cmovnae`	`CF`	低于(<)
`cmovbe S, R`	`cmovna`	`CF | ZF`	低于或相等(≤)

转移控制

指令	描述
`call Label`	过程调用
`call *Operand`	过程调用
`ret`	从过程调用中返回

传送函数参数的寄存器

操作数大小	1	2	3	4	5	6
64	`%rdi`	`%rsi`	`%rdx`	`%rcx`	`%r8`	`%r9`
32	`%edi`	`%esi`	`%edx`	`%ecx`	`%r8d`	`%r9d`
16	`%di`	`%si`	`%dx`	`%cx`	`%r8w`	`%r9w`
8	`%dil`	`%sil`	`%dl`	`%cl`	`%r8b`	`%r9b`

指针运算

表达式	类型	值	汇编代码
`E`	`int *`	x_E	`movq %rdx, %rax`
`E[0]`	`int`	$M[x_E]$	`movl (%rdx), %rax`
`E[i]`	`int`	$M[x_E + 4i]$	`movl (%rdx, %rcx, 4), %rax`
`&E[2]`	`int *`	$x_E + 8$	`leaq 8(%rdx), %rax`
`E + i - 1`	`int *`	$x_E + 4i - 4$	`leaq -4(%rdx, %rcx, 4), %rax`
`*(E + i - 3)`	`int`	$M[x_E + 4i - 12]$	`movl -12(%rdx, %rcx, 4), %eax`
`&E[i] - E`	`long`	i	`movq %rcx, %rax`

媒体寄存器

256	128	描述
`%ymm0`	`%xmm0`	`1st` FParg.返回值
`%ymm1`	`%xmm1`	`2nd` FP参数
`%ymm2`	`%xmm2`	`3rd` FP参数
`%ymm3`	`%xmm3`	`4th` FP参数
`%ymm4`	`%xmm4`	`5th` FP参数
`%ymm5`	`%xmm5`	`6th` FP参数
`%ymm6`	`%xmm6`	`7th` FP参数
`%ymm7`	`%xmm7`	`8th` FP参数
`%ymm8`	`%xmm8`	调用者保存
`%ymm9`	`%xmm9`	调用者保存
`%ymm10`	`%xmm10`	调用者保存
`%ymm11`	`%xmm11`	调用者保存
`%ymm12`	`%xmm12`	调用者保存
`%ymm13`	`%xmm13`	调用者保存
`%ymm14`	`%xmm14`	调用者保存
`%ymm15`	`%xmm15`	调用者保存

浮点传送指令

指令	源	目的	描述
`vomvss`	M_{32}	X	传送单精度数
`vmovss`	X	M_{32}	传送单精度数
`vmovsd`	M_{64}	X	传送双精度数
`vmovsd`	X	M_{64}	传送双精度数
`vmovaps`	X	X	传送对齐的封装好的单精度数
`vmovapd`	X	X	传送对其的封装好的双精度数

浮点转换指令

双操作数浮点转换指令

X：XMM寄存器

R_{32}：32位通用寄存器　R_{64}：64位通用寄存器

M_{32}：32位内存范围　M_{64}：64位内存范围

指令	源	目的	描述
`vcvttss2si`	X/M_{32}	R_{32}	用截断的方法把单精度数转换成整数
`vcvttsd2si`	X/M_{64}	R_{32}	用截断的方法把双精度数转换成整数
`vcvttss2siq`	X/M_{32}	R_{64}	用截断的方法把单精度数转换成四字整数
`vxvttsd2siq`	X/M_{64}	R_{64}	用截断的方法把双精度数转换成四字整数

三操作数浮点转换指令

指令	源1	源2	目的	描述
`vcvtsi2ss`	M_{32}/R_{32}	X	X	
`vcvtsi2sd`	M_{32}/R_{32}	X	X	
`vcvtsi2ssq`	M_{64}/R_{64}	X	X	
`vcvtsi2sdq`	M_{64}/R_{64}	X	X	

浮点运算操作

标量浮点算数运算

单精度	双精度	效果	描述
`vaddss`	`vaddsd`	$\leftarrow S_2 + S_1$	浮点数加
`vsubss`	`vsubsd`	$D \leftarrow S_2 - S_1$	浮点数减
`vmulss`	`vmulsd`	$D \leftarrow S_2 \times S_1$	浮点数乘
`vdivss`	`vdivsd`	$D \leftarrow S_2 / S_1$	浮点数除
`vmaxss`	`vmaxsd`	$D \leftarrow \max(S_2,\ S_1)$	浮点数最大值
`vminss`	`vminsd`	$D \leftarrow \min(S_2,\ S_1)$	浮点数最小值
`sqrtss`	`sqrtsd`	$D \leftarrow \sqrt{S_1}$	浮点数平方根

位级操作

单精度	双精度	效果	描述
`vxorps`	`vorpd`	$D \leftarrow S_2 \hat{} S_1$	位级异或
`vandps`	`andpd`	$D \leftarrow S_2 \& S_1$	位级与

比较操作

指令	基于	描述
`ucomiss` $S_1,\ S_2$	$S_2 - S_1$	比较单精度值
`ucomisd` $S_1,\ S_2$	$S_2 - S_1$	比较双精度值

顺序 $S_2 : S_1$	CF	ZF	PF
无序的	1	1	1
$S_2 < S_1$	1	0	0
$S_2 = S_1$	0	1	0
$S_2 > S_1$	0	0	0