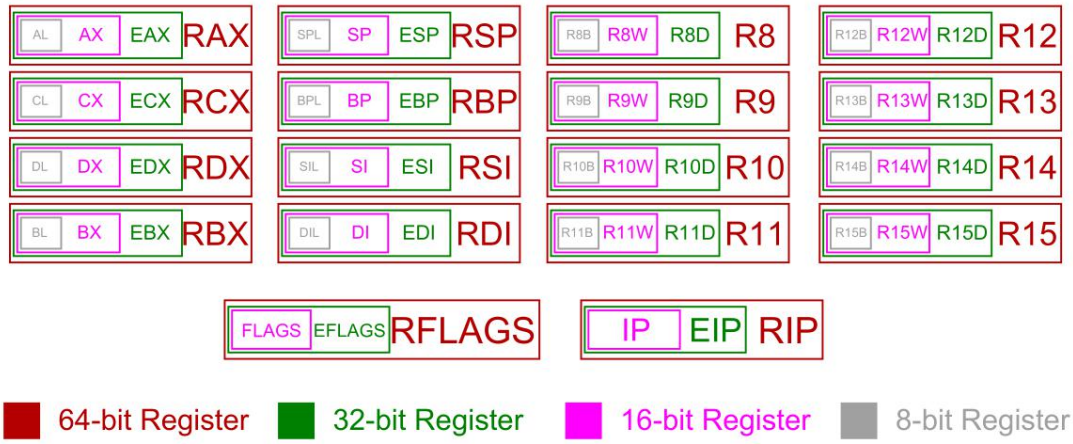
Manuale di programmazione z64

Registri General Purpose



Condizioni di confronto

Condizione	Aritmetica non segnata	Aritmetica segnata
`dest < source`	CF = 1	SF ≠ OF
`dest ≥ source`	CF = 0	SF = OF
`dest > source`	CF = 0 e ZF = 0	ZF = 0 e SF = OF
`dest = source`	ZF = 1	ZF = 1
`dest ≠ source`	ZF = 0	ZF = 0

System V ABI Calling Conventions (64 bit)

Primi sei parametri passati nei registri: RDI, RSI, RDX, RCX, R8, R9
Parametri ulteriori passati su stack

Registri callee-save: RBP, RBX, R12-R15

Instruction Set

Le seguenti convenzioni vengono utilizzate per rappresentare gli operandi delle istruzioni:

- `B` — L'operando è un registro di uso generale, un indirizzo di memoria o un valore immediato. In caso di un indirizzo di memoria, qualsiasi combinazione delle modalità di indirizzamento è lecita. In caso di un immediato, la sua posizione dipende dalla possibile presenza dello spiazzamento e dalla sua dimensione.
- `E` — L'operando è un registro di uso generale, o un indirizzo di memoria. In caso di un indirizzo di memoria, qualsiasi combinazione delle modalità di indirizzamento è lecita.
- `G` — L'operando è un registro di uso generale.
- `K` — L'operanzo è una costante numerica non segnata di valore fino a $2^{32} - 1$
- `M` — L'operando è una locazione di memoria, codificata come uno spiazzamento a partire dal contenuto del registro RIP dopo l'esecuzione della fase di fetch

Classe 0

Tipo	Mnemonico	Operandi	O S Z P C	Descrizione
1	`hlt`	–	- - - - -	Mette la CPU in modalità di basso consumo energetico, finché non viene ricevuta l'interruzione successiva
2	`nop`	–	- - - - -	Nessuna operazione
3	`int`	—	- - - - -	Chiama esplicitamente un gestore di interruzioni

Classe 1

Tipo	Mnemonico	Operandi	O S Z P C	Descrizione
0	mov	B, E	- - - - -	Fa una copia di B in E
1	movsX	E, G	- - - - -	Fa una copia di E in G con estensione del segno
2	movzX	E, G	- - - - -	Fa una copia di E in G con estensione dello zero
3	lea	E, G	- - - - -	Valuta la modalità di indirizzamento, salva il risultato in G
4	push	E	- - - - -	Copia il contenuto di E sulla cima dello stack
5	pop	E	- - - - -	Copia il contenuto della cima dello stack in E
6	pushf	–	- - - - -	Copia sulla cima dello stack il registro FLAGS
7	popf	–	- - - - -	Copia nel registro FLAGS il contenuto della cima dello stack
8	movs	–	- - - - -	Esegue una copia memoria-memoria
9	stos	–	- - - - -	Imposta una regione di memoria ad un dato valore

Istruzione	Tipo di conversione
movsbw %al, %ax	Estendi il segno da byte a word
movsbl %al, %eax	Estendi il segno da byte a longword
movsbq %al, %rax	Estendi il segno da byte a quadword
movswl %ax, %eax	Estendi il segno da word a longword
movswq %ax, %rax	Estendi il segno da word a quadword
movslq %eax, %rax	Estendi il segno da longword a quadword

Classe 2

Tipo	Mnemonico	Operandi	O S Z P C	Descrizione
0	add	B, E	⇕ ⇕ ⇕ ⇕ ⇕	Memorizza in E il risultato di E + B
1	sub	B, E	⇕ ⇕ ⇕ ⇕ ⇕	Memorizza in E il risultato di E - B
2	adc	B, E	⇕ ⇕ ⇕ ⇕ ⇕	Memorizza in D il risultato di E + B + CF
3	sbb	B, E	⇕ ⇕ ⇕ ⇕ ⇕	Memorizza in D il risultato di E - (B + neg(CF))
4	cmp	B, E	⇕ ⇕ ⇕ ⇕ ⇕	Confronta i valori di B ed E calcolando E - B, il risultato viene poi scartato
5	test	B, E	⇕ ⇕ ⇕ ⇕ ⇕	Calcola l'and logico bit a bit di B ed E, il risultato viene poi scartato
6	neg	E	⇕ ⇕ ⇕ ⇕ ⇕	Rimpiazza il valore di E con il suo complemento a 2
7	and	B, E	0 ⇕ ⇕ ⇕ 0	Memorizza in E il risultato dell'and bit a bit tra B ed E
8	or	B, E	0 ⇕ ⇕ ⇕ 0	Memorizza in E il risultato dell'or bit a bit tra B ed E
9	xor	B, E	0 ⇕ ⇕ ⇕ 0	Memorizza in E il risultato dello xor bit a bit tra B edE
10	not	E	0 ⇕ ⇕ ⇕ 0	Rimpiazza il valore di E con il suo complemento a uno
11	bt	K, E	- - - - ⇕	Imposta CF al valore del K-simo bit di E (bit testing)

Classe 3

Tipo	Mnemonico	Operandi	O S Z P C	Descrizione
0	sal	K, G	⇕ ⇕ ⇕ ⇕ ⇕	Moltiplica per 2, K volte
1	sal	G	⇕ ⇕ ⇕ ⇕ ⇕	Moltiplica per 2, RCX volte
0	shl	K, G	⇕ ⇕ ⇕ ⇕ ⇕	Moltiplica per 2, K volte
1	shl	G	⇕ ⇕ ⇕ ⇕ ⇕	Moltiplica per 2, RCX volte
2	sar	K, G	⇕ ⇕ ⇕ ⇕ ⇕	Dividi (con segno) per 2, K volte
3	sar	G	⇕ ⇕ ⇕ ⇕ ⇕	Dividi (con segno) per 2, RCX volte
4	shr	K, G	⇕ ⇕ ⇕ ⇕ ⇕	Dividi (senza segno) per 2, K volte
5	shr	G	⇕ ⇕ ⇕ ⇕ ⇕	Dividi (senza segno) per 2, RCX volte
6	rcl	K, G	⇕ - - - ⇕	Ruota a sinistra, K volte
7	rcl	G	⇕ - - - ⇕	Ruota a sinistra, RCX volte
8	rcr	K, G	⇕ - - - ⇕	Ruota a destra, K volte
9	rcr	G	⇕ - - - ⇕	Ruota a destra, RCX volte
10	rol	K, G	⇕ - - - ⇕	Ruota a sinistra, K volte
11	rol	G	⇕ - - - ⇕	Ruota a sinistra, RCX volte
12	ror	K, G	⇕ - - - ⇕	Ruota a destra, K volte
13	ror	G	⇕ - - - ⇕	Ruota a destra, RCX volte

Classe 4

Tipo	Mnemonico	Operandi	O S Z P C	Descrizione
0	clc	–	- - - - 0	Resetta CF
1	clp†	–	- - - 0 -	Resetta PF
2	clz†	–	- - 0 - -	Resetta ZF
3	cls†	–	- 0 - - -	Resetta SF
4	cli	–	- - - - -	Resetta IF
5	cld	–	- - - - -	Resetta DF
6	clo†	–	0 - - - -	Resetta OF
7	stc	–	- - - - 1	Imposta CF
8	stp†	–	- - - 1 -	Imposta PF
9	stz†	–	- - 1 - -	Imposta ZF
10	sts†	–	- 1 - - -	Imposta SF
11	sti	–	- - - - -	Imposta IF
12	std	–	- - - - -	Imposta DF
13	sto†	–	1 - - - -	Imposta OF

Classe 5

Tipo	Mnemonico	Operandi	O S Z P C	Descrizione
0	jmp	M	- - - - -	Esegue un salto relativo
1	jmp	*G	- - - - -	Esegui un salto assoluto
2	call	M	- - - - -	Esegue una chiamata a subroutine relativa
3	call	*G	- - - - -	Esegue una chiamata a subroutine assoluta
4	ret	–	- - - - -	Ritorna da una subroutine
5	iret	–	⇕ ⇕ ⇕ ⇕ ⇕	Ritorna dal gestore di una interruzione

Classe 6

Tipo	Mnemonico	Operandi	O S Z P C	Descrizione
0	jc	M	- - - - -	Salta a M se CF è impostato
1	jp	M	- - - - -	Salta a M se PF è impostato
2	jz	M	- - - - -	Salta a M se ZF è impostato
3	js	M	- - - - -	Salta a M se SF è impostato
4	jo	M	- - - - -	Salta a M se OF è impostato
5	jnc	M	- - - - -	Salta a M se CF non è impostato
6	jnp	M	- - - - -	Salta a M se PF non è impostato
7	jnz	M	- - - - -	Salta a M se ZF non è impostato
8	jns	M	- - - - -	Salta a M se SF non è impostato
9	jno	M	- - - - -	Salta a M se OF non è impostato

Classe 7

Instruction	Syntax	Semantics
Inbound transfer from parametric I/O port	inX %dx, RAX	Transfer data of size X from the device deployed on the I/O address contained in the %dx register.
Inbound transfer from explicit I/O port	inX $ioport, RAX	Transfer data of size X from the device deployed on the I/O address $ioport.
Outbound transfer to parametric I/O	outX RAX, %dx	Transfer data of size X to the device deployed on the I/O address contained in the %dx register.
Outbound transfer to explicit I/O	outX RAX, $ioport	Transfer data of size X to the device deployed on the I/O address $ioport.
Inbound transfer of a data string	insX	Transfer an arbitrarily large buffer of data from a device.
Outbound transfer of a data string	outsX	Transfer an arbitrarily large buffer of data to a device.