

ARCHITETTURA DEGLI ELABORATORI
APPELLO DEL 18/07/2017

ESERCIZIO I [8 PUNTI]

I registri R1 e R3 contengono due indirizzi di parole nella memoria dati. Si scriva un frammento di codice assembly che scambi le due parole.

Soluzione

lw R2, R1(0)	$R1 \leftarrow \text{valore all'indirizzo in } R2$
lw R4, R3(0)	$R3 \leftarrow \text{valore all'indirizzo in } R4$
sw R2, R3(0)	<i>store word per scambio</i>
sw R4, R1(0)	

ESERCIZIO II [8 PUNTI]

Si assuma che il contenuto dei primi 16 registri del Register File sia $R_i = 2i$, per $i = 0, 1, 2, \dots, 15$. Nel segmento testo sono memorizzate (in sequenza) queste due istruzioni:

0 0 8 4 4 8 2 *A*
0 1 2 *E* 1 8 2 0

L'indirizzo della prima delle due è:

0 0 0 0 1 *C* 2 0

- (1) Si scrivano le due istruzioni in linguaggio assembly;
- (2) Si indichino quali registri cambiano il loro valore dopo l'esecuzione delle due istruzioni e si riporti, per ciascuno, il nuovo valore in codifica esadecimale.

Soluzione (1) In codice binario la prima istruzione è:

0000 0000 1000 0100 0100 1000 0010 1010

da cui:

- *opcode* = 000000;
- *sham* = 00000
- *function* = 101010

Quindi, ispezionando la tabella delle info utili si conclude che è una *slt*. Inoltre,

- $rs = 00100 = 4$;
- $rt = 00100 = 4$;
- $rd = 01001 = 9$

La sua codifica assembly quindi : *slt R9, R4, R4*.

In codice binario la seconda istruzione è:

0000 0001 0010 1110 0100 1000 0010 0000

da cui:

- $opcode = 000000$;
- $sham = 00000$
- $function = 100000$

Quindi, ispezionando la tabella delle info utili si conclude che è una *add*. Inoltre,

- $rs = 01001 = 9$;
- $rt = 01110 = 14$;
- $rd = 00011 = 3$

La sua codifica assembly quindi : *add R3, R9, R14*.

Soluzione (2) I registri che cambiano valore sono:

- *R9* che assume valore 0 0 0 0 0 0 0 0;
- *R3* che assume valore 0 0 0 0 0 0 1 *C*;
- *PC* che assume valore 0 0 0 0 1 *C* 2 8

INFO UTILI

istruzione	codice	function	ALUOp
and	000000	100100	00
or	000000	100101	00
add	000000	100000	00
sub	000000	100010	00
slt	000000	101010	00
sw	101011	xxxxxx	01
lw	100011	xxxxxx	01
beq	000100	xxxxxx	10
nop	000000	000000	00