

```

if ( test == 0 ) {
    istruzione_1
}
else {
    istruzione_2
}
istruzione_3

```

PONENDO: test → \$t0

Per mantenere lo schema dell'istruzione IF-ELSE mostrato in figura la traduzione in Assembly MIPS della condizione logica su uguaglianza è:

- A. L'istruzione bne \$t0,\$zero, ELSE, dove ELSE è l'etichetta di istruzione_2
- B. L'istruzione j ELSE, dove ELSE è l'etichetta di istruzione_2
- C. L'istruzione bne \$t0,\$zero, END_IF, dove END_IF è l'etichetta di istruzione_3
- D. L'istruzione j END_IF dove END_IF è l'etichetta di istruzione_3

```

if ( val != Max ) {
    istruzione_1
}
else {
    istruzione_2
}
istruzione_3

```

PONENDO: val → \$t0
Max → \$t1

Per mantenere lo schema dell'istruzione IF-ELSE mostrato in figura la traduzione in Assembly MIPS della condizione logica su disuguaglianza è:

- A. L'istruzione j ELSE, dove ELSE è l'etichetta di istruzione_2
- B. L'istruzione beq \$t0,\$t1, ELSE, dove ELSE è l'etichetta di istruzione_2
- C. L'istruzione j END_IF dove END_IF è l'etichetta di istruzione_3
- D. L'istruzione beq \$t0,\$t1, END_IF, dove END_IF è l'etichetta di istruzione_3

```

for (n = 0; n != Max; n = n+1) {
    istruzione_1
}
istruzione_2

```

PONENDO: n → \$t0
Max → \$t1

Per ottenere lo schema dell'istruzione FOR mostrato in figura la traduzione in Assembly MIPS della condizione logica su disuguaglianza è:

- A. L'istruzione beq \$t0,\$t1, END_FOR, dove END_FOR è l'etichetta della istruzione_2
- B. L'istruzione j END_FOR, dove END_FOR è l'etichetta della istruzione_2
- C. L'istruzione beq \$t0,\$t1, CICLO, dove CICLO è l'etichetta che porta alla ripetizione del ciclo FOR
- D. L'istruzione j CICLO dove CICLO è l'etichetta che porta alla ripetizione del ciclo FOR

```

if ( a < b ) {
    min = a ;
}
else {
    min= b ;
}
a = 0 ;

```

Assegnando alle variabili i registri:
a b min
\$t1 \$t2 \$s0

1	<pre> slt \$t5, \$t1, \$t2 beq \$t5, \$zero, ELSE addi \$s0, \$t1, 0 j END_IF ELSE: addi \$s0, \$t2, 0 END_IF: add \$t1, \$zero, \$zero </pre>	2	<pre> slt \$t5, \$t1, \$t2 bne \$t5, \$zero, ELSE addi \$s0, \$t1, 0 j END_IF ELSE: addi \$s0, \$t2, 0 END_IF: add \$t1, \$zero, \$zero </pre>
3	<pre> bne \$t1, \$t2, ELSE addi \$s0, \$t1, 0 j END_IF ELSE: addi \$s0, \$t2, 0 END_IF: add \$t1, \$zero, \$zero </pre>	4	<pre> slt \$t5, \$t1, \$t2 beq \$t5, \$zero, ELSE addi \$s0, \$t1, 0 ELSE: addi \$s0, \$t2, 0 END_IF: add \$t1, \$zero, \$zero </pre>

```

val = 65539 ;

ponendo: val ↔ $s0
rappresentazione binaria di 65539:
00000000000000001000000000000001

```

Le istruzioni in figura possono essere tradotte in Assembly MIPS:

1 addi \$s0, \$zero, 60000

addi \$s0, \$s0, 5539

2 ori \$s0, \$zero, 1
sll \$s0, \$s0, 16
ori \$s0, \$s0, 3

3 addi \$s0, \$zero, 65539

4 ori \$s0, \$zero, 1
srl \$s0, \$s0, 16
ori \$s0, \$s0, 3

```

char primo = 'S', secondo = 'B', temp, min ;
if (secondo < primo ) {
    temp = primo ;
    primo = secondo ;
    secondo = temp ; }
min = primo ;

```

1 assegnando: primo secondo temp min
 \$ s1 \$ s2 \$ t0 \$ s5
 ↓ ↓ ↓ ↓
 slt \$t3, \$s2, \$s1 END_IF
 addi \$t0, \$s1, 0
 addi \$s1, \$s2, 0
 addi \$s2, \$t0, 0
END_IF add \$s5, \$zero, \$s1

2 assegnando: primo secondo temp min
 \$ s1 \$ s2 \$ t0 \$ s5
 ↓ ↓ ↓ ↓
 slt \$t3, \$s2, \$s1
 beq \$t3, \$zero, END_IF
 addi \$t0, \$s1, 0
 addi \$s1, \$s2, 0
 addi \$s2, \$t0, 0
END_IF add \$s5, \$zero, \$s1

3 assegnando: primo secondo temp min
 \$ s1 \$ s2 \$ t0 \$ s5
 ↓ ↓ ↓ ↓
 bne \$s2, \$s1, END_IF
 addi \$t0, \$s1, 0
 addi \$s1, \$s2, 0
 addi \$s2, \$t0, 0
END_IF add \$s5, \$zero, \$s1

4 assegnando: primo secondo temp min
 \$ s1 \$ s2 \$ t0 \$ s5
 ↓ ↓ ↓ ↓
 slt \$s2, \$s1 END_IF
 addi \$t0, \$s1, 0
 addi \$s1, \$s2, 0
 addi \$s2, \$t0, 0
END_IF add \$s5, \$zero, \$s1

1. La traduzione in Assembly MIPS dell'assegnamento val = A[k] con A[] Array di INTERI è data:

A [] val k
\$s3 \$t0 \$t7

sll \$t1, \$t7, 2
add \$t6, \$t1, \$s3
lw \$t0, 0 (\$t6)

A [] val k
\$s3 \$t0 \$t7

add \$t6, \$t7, \$s3
lw \$t0, 0 (\$t6)

A [] val k
\$s3 \$t0 \$t7

lw \$t0, \$t7 (\$s3)

A [] val k
\$s3 \$t0 \$t7

sll \$t1, \$t7, 2
lw \$t0, 0 (\$t1)

1

2

3

4

2. La traduzione in Assembly MIPS dell'assegnamento A[k] = val con A[] Array di INTERI è data:

1

2

3

4

A [] val k
\$s3 \$t0 \$t7

add \$t6, \$t7, \$s3
sw \$t0, 0 (\$t6)

A [] val k
\$s3 \$t0 \$t7

sll \$t1, \$t7, 2
add \$t6, \$t1, \$s3
sw \$t0, 0 (\$t6)

A [] val k
\$s3 \$t0 \$t7

sll \$t1, \$t7, 2
sw \$t0, 0 (\$t1)

A [] val k
\$s3 \$t0 \$t7

sw \$t0, \$t7 (\$s3)

3. La traduzione in Assembly MIPS dell'assegnamento val = testo[k] con testo[] Array di caratteri ASCII è data:

1

2

3

4

testo [] val k
\$t1 \$s5 \$t7

add \$t6, \$t7, \$t1
lbu \$s5, 0 (\$t6)

testo [] val k
\$t1 \$s5 \$t7

add \$t6, \$t7, \$t1
sw \$s5, 0 (\$t6)

testo [] val k
\$t1 \$s5 \$t7

add \$t6, \$t7, \$t1
lbu \$s5, \$t7 (\$t1)

testo [] val k
\$t1 \$s5 \$t7

add \$t6, \$t7, \$t1
lbu \$s5, \$t1 (\$t7)

La traduzione in Assembly MIPS dell'assegnamento testo[k] = val con testo[] Array di caratteri ASCII è data:

1

2

3

4

testo [] val k
\$t1 \$s5 \$t7

add \$t6, \$t7, \$t1
lbu \$s5, 0 (\$t6)

testo [] val k
\$t1 \$s5 \$t7

add \$t6, \$t7, \$t1
sb \$s5, 0 (\$t6)

testo [] val k
\$t1 \$s5 \$t7

add \$t6, \$t7, \$t1
sb \$s5, \$t7 (\$t1)

testo [] val k
\$t1 \$s5 \$t7

add \$t6, \$t7, \$t1
sb \$s5, \$t1 (\$t7)

ARRAY di INTERI
if (0 < A[k])

Le istruzioni in figura possono essere tradotte in Assembly MIPS:

1

2

3

4

Base A [] k temp
\$s0 \$t1 \$t3

sll \$t4, \$t1, 2
add \$t5, \$t4, \$s0
lw \$s6, 0 (\$t5)
slt \$t0, \$zero, \$s6
beq \$t0, \$zero, END_IF

Base A [] k temp
\$s0 \$t1 \$t3

lw \$s6, \$t1 (\$s0)
slt \$t0, \$zero, \$s6
beq \$t0, \$zero, END_IF

Base A [] k temp
\$s0 \$t1 \$t3

add \$t5, \$t3, \$s0
lw \$s6, 0 (\$t5)
slt \$t0, \$zero, \$s6
beq \$t0, \$zero, END_IF

Base A [] k temp
\$s0 \$t1 \$t3

sll \$t4, \$t1, 2
add \$t5, \$t4, \$s0
sw \$s6, 0 (\$t5)
slt \$t0, \$zero, \$s6
beq \$t0, \$zero, END_IF

ARRAY di INTERI

```
temp = A[k] ;
A[k+1] = temp ;
```

Le istruzioni in figura possono essere tradotte in Assembly MIPS:

1

Base A[]	k	temp
\$s0	\$t1	\$t3
add \$t5, \$t3, \$s0		
lw \$t3, 0(\$t5)		
sw \$t3, 4(\$t5)		

2

Base A[]	k	temp
\$s0	\$t1	\$t3
sll \$t4, \$t1, 2		
add \$t5, \$t4, \$s0		
lw \$t3, 0(\$t5)		
sw \$t3, 4(\$t5)		

3

Base A[]	k	temp
\$s0	\$t1	\$t3
sll \$t4, \$t1, 2		
add \$t5, \$t4, \$s0		
sw \$t3, 0(\$t5)		
lw \$t3, 4(\$t5)		

4

Base A[]	k	temp
\$s0	\$t1	\$t3
sll \$t4, \$t1, 2		
add \$t5, \$t4, \$s0		
lw \$t3, 0(\$t5)		
sw \$t3, 0(\$t5)		

ARRAY di CARATTERI ASCII

```
while ( testo[k] != 0 )
```

Le istruzioni in figura possono essere tradotte in Assembly MIPS:

1

Base testo[]	k
\$s3	\$t7
WHILE lbu \$t1, \$t7(\$s3)	
beq \$t1, \$zero, END WHILE	

2

Base testo[]	k
\$s3	\$t7
WHILE add \$t2, \$t7, \$s3	
lbu \$t1, 0(\$t2)	
beq \$t1, \$zero, END WHILE	

3

Base testo[]	k
\$s3	\$t7
WHILE add \$t2, \$t7, \$s3	
sb \$t1, 0(\$t2)	
beq \$t1, \$zero, END WHILE	

4

Base testo[]	k
\$s3	\$t7
WHILE add \$t2, \$t7, \$s3	
lbu \$t2, 0(\$t7)	
beq \$t1, \$zero, END WHILE	

ARRAY di CARATTERI ASCII

```
if ( testo[k] == '\032' )
```

1

Base testo[]	k
\$s3	\$t7
ori \$t6, \$zero, 32	
lbu \$t1, \$t7(\$s3)	
bne \$t1, \$t6, END_IF	

2

Base testo[]	k
\$s3	\$t7
ori \$t6, \$zero, 32	
add \$t2, \$t7, \$s3	
sb \$t1, 0(\$t2)	
bne \$t1, \$t6, END_IF	

3

Base testo[]	k
\$s3	\$t7
ori \$t6, \$zero, 32	
add \$t2, \$t7, \$s3	
lbu \$t1, 0(\$t2)	
bne \$t1, \$t6, END_IF	

4

Base testo[]	k
\$s3	\$t7
ori \$t6, \$zero, 32	
add \$t2, \$t7, \$s3	
lbu \$t1, 0(\$t2)	
bne \$t1, \$t6, END_IF	

ARRAY di CARATTERI ASCII

```
x[k] = y[k] ;
```

1

2

3

4

Base x[]	Base y[]	k
\$s1	\$s2	\$t3
add \$t6, \$t3, \$s2		
lbu \$t0, 0(\$t6)		
add \$t5, \$t3, \$s1		
sb \$t0, 0(\$t5)		

Base x[]	Base y[]	k
\$s1	\$s2	\$t3
add \$t6, \$t3, \$s2		
lw \$t0, 0(\$t6)		
add \$t5, \$t3, \$s1		
sw \$t0, 0(\$t5)		

Base x[]	Base y[]	k
\$s1	\$s2	\$t3
lbu \$t0, \$t3(\$t6)		
sb \$t0, \$t3(\$t5)		

Base x[]	Base y[]	k
\$s1	\$s2	\$t3
add \$t6, \$t3, \$s2		
lhu \$t0, 0(\$t6)		
add \$t5, \$t3, \$s1		
sh \$t0, 0(\$t5)		

il formato dei numeri frazionari fissato dallo standard IEEE 754 precisione SEMPLICE è:

1

2

3

4

1 bit	8 bit	23 bit
s	esponente	mantissa
$(-1)^s \times (1 + 0, \text{mantissa}) \times 2^{\text{esponente}}$		

1 bit	8 bit	23 bit
s	esponente polarizzato	mantissa
$(-1)^s \times (1 + 0, \text{mantissa}) \times 2^{(\text{esponente polarizzato} - 127)}$		

8 bit	24 bit
esponente polarizzato	mantissa
$(1 + 0, \text{mantissa}) \times 2^{(\text{esponente polarizzato} - 127)}$	

1 bit	8 bit	23 bit
s	esponente polarizzato	mantissa
$(-1)^s \times (0, \text{mantissa}) \times 2^{(\text{esponente polarizzato} - 127)}$		

Il formato dei numeri frazionari fissato dallo standard IEEE 754 precisione DOPPIA è:

1

2

3

4

1 bit	11 bit	20 bit
s	esponente polarizzato	mantissa
32 bit		

1 bit	11 bit	20 bit
s	esponente polarizzato	mantissa
32 bit		

1 bit	11 bit	20 bit
s	esponente polarizzato	mantissa
32 bit		

12 bit	20 bit
esponente polarizzato	mantissa
32 bit	

x	y	f(x, y)
0 0		1
0 1		0
1 0		1
1 1		0

$$f(x, y) = x \cdot y + \bar{x} \cdot y$$

$$f(x, y) = x \cdot \bar{y} + \bar{x} \cdot \bar{y}$$

$$f(x, y) = \bar{x} \cdot \bar{y} + x \cdot \bar{y}$$

$$f(x, y) = (\bar{x} + y) \cdot (x + \bar{y})$$

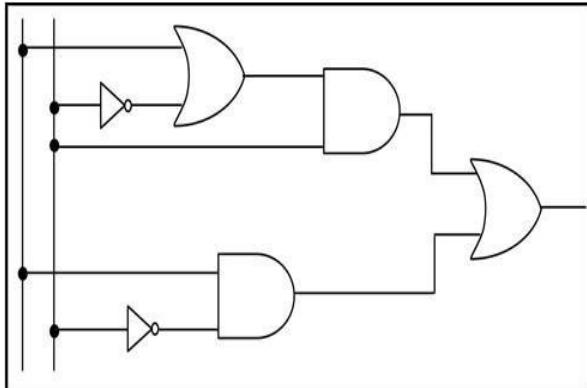
x	y	z	f(x, y, z)
0 0 0			0
0 0 1			1
0 1 0			0
0 1 1			0
1 0 0			0
1 0 1			1
1 1 0			0
1 1 1			1

$$f(x, y) = \bar{x} \cdot \bar{y} \cdot z + x \cdot \bar{y} \cdot z + x \cdot y \cdot z$$

$$f(x, y) = \bar{x} \cdot \bar{y} \cdot \bar{z} + \bar{x} \cdot y \cdot \bar{z} + \bar{x} \cdot y \cdot z + x \cdot \bar{y} \cdot \bar{z} + x \cdot y \cdot \bar{z}$$

$$f(x, y) = x \cdot y \cdot \bar{z} + \bar{x} \cdot y \cdot \bar{z} + \bar{x} \cdot \bar{y} \cdot \bar{z}$$

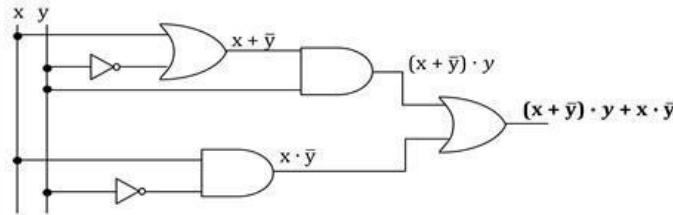
$$f(x, y) = (\bar{x} + \bar{y} + z) \cdot (x + \bar{y} + z) \cdot (x + y + z)$$



1

2

Espressione Booleana associata al terminale output della Rete Combinatoria



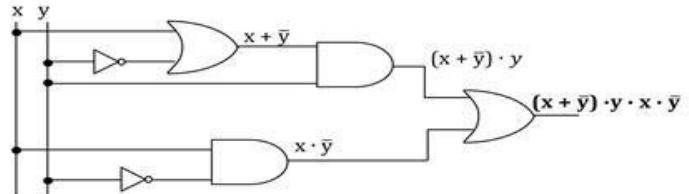
Funzione Booleana $f(x,y)$ associata alla Rete Combinatoria:

xy	$x + \bar{y}$	$(x + \bar{y}) \cdot y$	$x \cdot \bar{y}$	$(x + \bar{y}) \cdot y + x \cdot \bar{y}$
00	1	0	0	0
01	0	0	0	0
10	1	0	1	1
11	1	1	0	1

3

D5 - R3

Espressione Booleana associata al terminale output della Rete Combinatoria



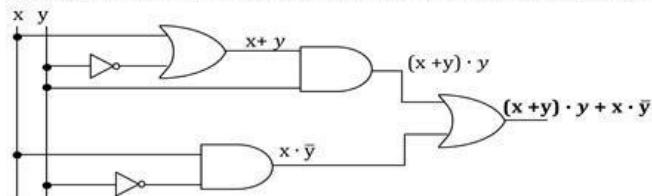
Funzione Booleana $f(x,y)$ associata alla Rete Combinatoria:

xy	$x + \bar{y}$	$(x + \bar{y}) \cdot y$	$x \cdot \bar{y}$	$(x + \bar{y}) \cdot y + x \cdot \bar{y}$
00	1	0	0	0
01	0	0	0	0
10	1	0	1	0
11	1	1	0	0

4

D5 - R2

Espressione Booleana associata al terminale output della Rete Combinatoria

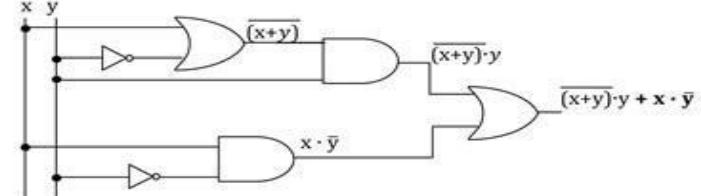


Funzione Booleana $f(x,y)$ associata alla Rete Combinatoria:

xy	$x + y$	$(x + y) \cdot y$	$x \cdot \bar{y}$	$(x + y) \cdot y + x \cdot \bar{y}$
00	0	0	0	0
01	1	1	0	1
10	1	0	1	1
11	1	1	0	1

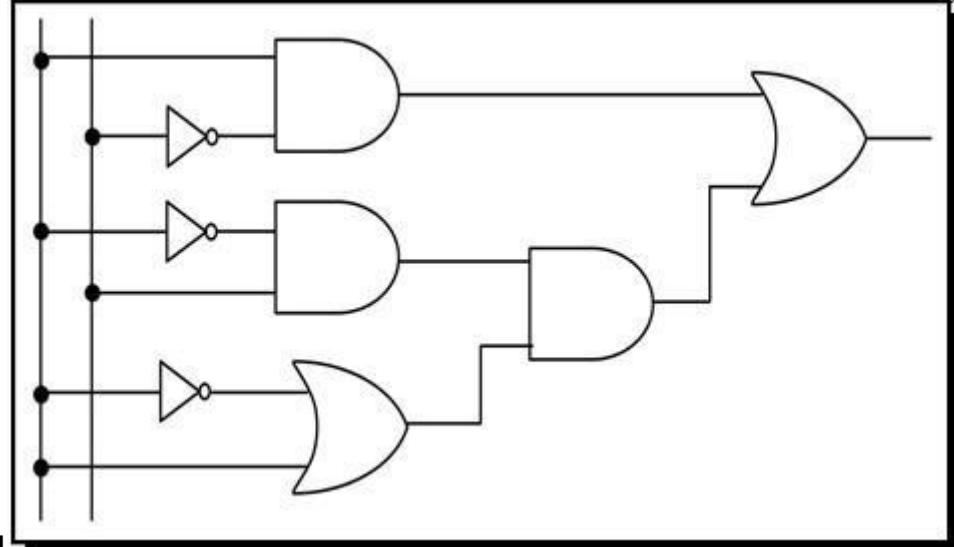
D5 - R4

Espressione Booleana associata al terminale output della Rete Combinatoria



Funzione Booleana $f(x,y)$ associata alla Rete Combinatoria:

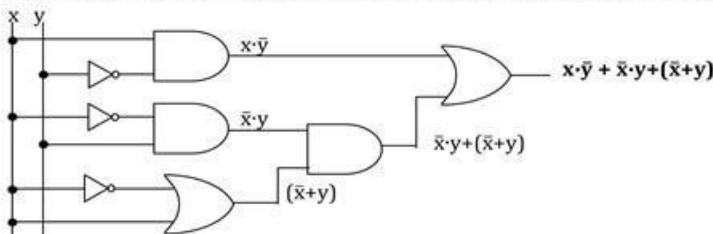
xy	$\bar{x} + \bar{y}$	$(\bar{x} + \bar{y}) \cdot y$	$x \cdot \bar{y}$	$(\bar{x} + \bar{y}) \cdot y + x \cdot \bar{y}$
00	1	0	0	0
01	1	1	1	1
10	1	0	0	0
11	0	0	0	0



L'Analisi trial Rete Combinatoria in figura è ottenuta mediante:

1

Espressione Booleana associata al terminale *output* della Rete Combinatoria

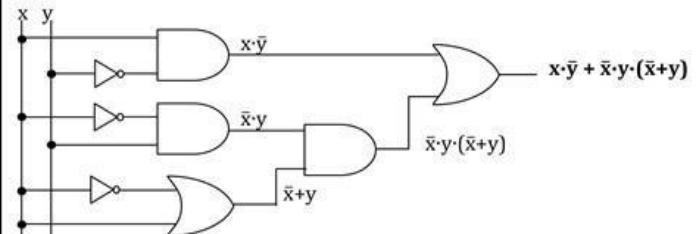


Funzione Booleana $f(x,y)$ associata alla Rete Combinatoria:

xy	$x \cdot \bar{y}$	$\bar{x} \cdot y$	$(\bar{x}+y)$	$\bar{x} \cdot y + (\bar{x}+y)$	$x \cdot \bar{y} + \bar{x} \cdot y + (\bar{x}+y)$
00	0	0	1	1	1
01	0	1	1	1	1
10	1	0	0	0	1
11	0	0	1	1	1

2

Espressione Booleana associata al terminale *output* della Rete Combinatoria

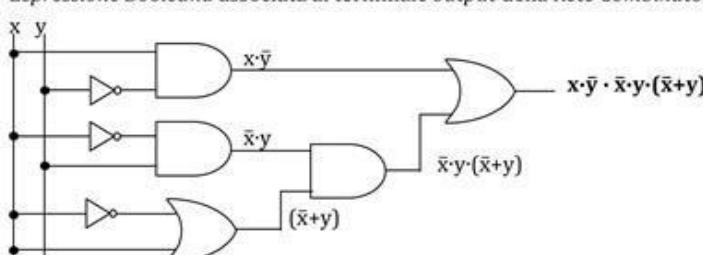


Funzione Booleana $f(x,y)$ associata alla Rete Combinatoria:

xy	$x \cdot \bar{y}$	$\bar{x} \cdot y$	$(\bar{x}+y)$	$\bar{x} \cdot y \cdot (\bar{x}+y)$	$x \cdot \bar{y} + \bar{x} \cdot y \cdot (\bar{x}+y)$
00	0	0	1	0	0
01	0	1	1	1	1
10	1	0	0	0	1
11	0	0	1	0	0

3

Espressione Booleana associata al terminale *output* della Rete Combinatoria

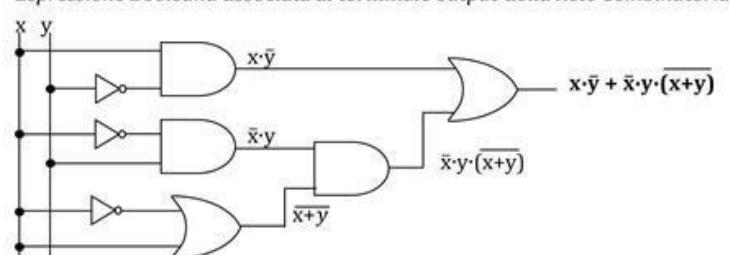


Funzione Booleana $f(x,y)$ associata alla Rete Combinatoria:

xy	$x \cdot \bar{y}$	$\bar{x} \cdot y$	$(\bar{x}+y)$	$\bar{x} \cdot y \cdot (\bar{x}+y)$	$x \cdot \bar{y} \cdot \bar{x} \cdot y \cdot (\bar{x}+y)$
00	0	0	1	0	0
01	0	1	1	1	0
10	1	0	0	0	0
11	0	0	1	0	0

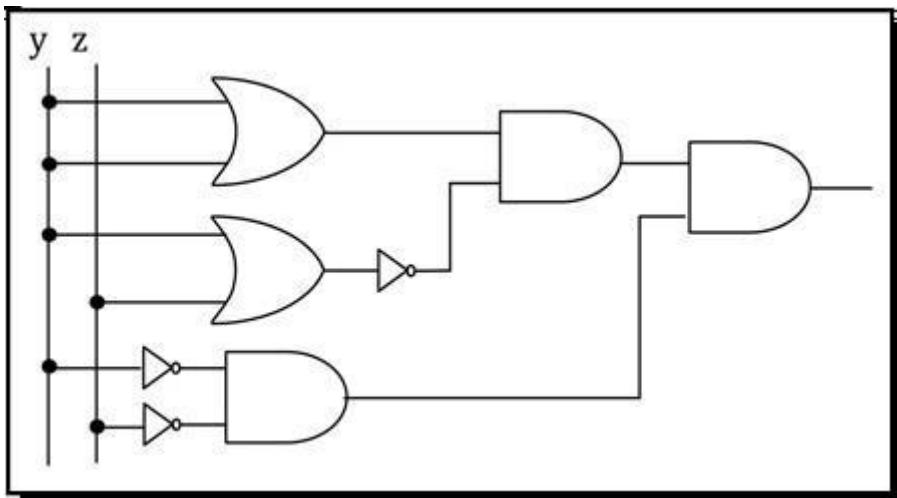
4

Espressione Booleana associata al terminale *output* della Rete Combinatoria



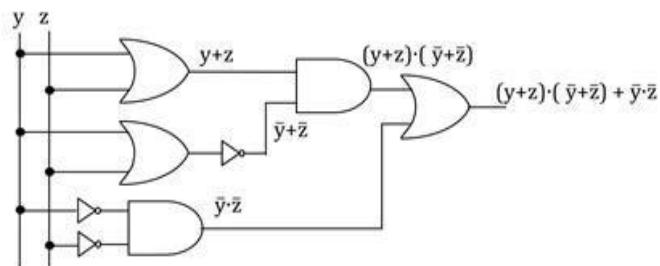
Funzione Booleana $f(x,y)$ associata alla Rete Combinatoria:

xy	$x \cdot \bar{y}$	$\bar{x} \cdot y$	$(\bar{x}+y)$	$\bar{x} \cdot y \cdot (\bar{x}+y)$	$x \cdot \bar{y} + \bar{x} \cdot y \cdot (\bar{x}+y)$
00	0	0	1	0	0
01	0	1	0	0	0
10	1	0	0	0	1
11	0	0	0	0	0



1

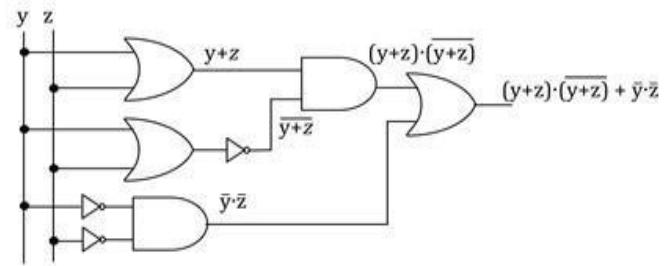
Espressione Booleana associata al terminale output della Rete Combinatoria

Funzione Booleana $f(x,y)$ associata alla Rete Combinatoria:

yz	$y+z$	$\bar{y}+z$	$(y+z) \cdot (\bar{y}+z)$	$\bar{y} \cdot \bar{z}$	$(y+z) \cdot (\bar{y}+z) + \bar{y} \cdot \bar{z}$
00	0	1	0	1	1
01	1	1	1	0	1
10	1	1	1	0	1
11	1	0	0	0	0

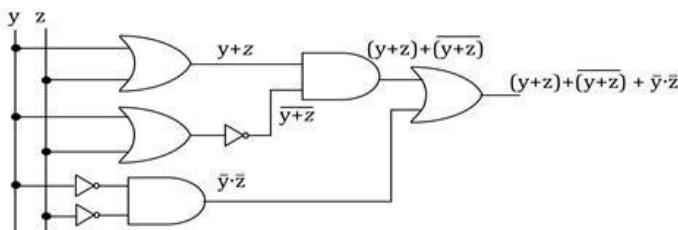
2

Espressione Booleana associata al terminale output della Rete Combinatoria

Funzione Booleana $f(x,y)$ associata alla Rete Combinatoria:

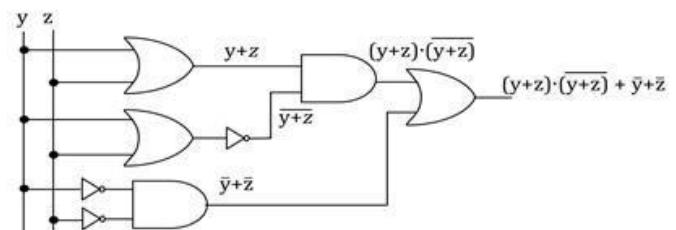
yz	$y+z$	$\bar{y}+z$	$(y+z) \cdot (\bar{y}+z)$	$\bar{y} \cdot \bar{z}$	$(y+z) \cdot (\bar{y}+z) + \bar{y} \cdot \bar{z}$
00	0	1	0	1	1
01	1	0	0	0	0
10	1	0	0	0	0
11	1	0	0	0	0

Espressione Booleana associata al terminale output della Rete Combinatoria

Funzione Booleana $f(x,y)$ associata alla Rete Combinatoria:

yz	$y+z$	$\bar{y}+z$	$(y+z) + (\bar{y}+z)$	$\bar{y} \cdot \bar{z}$	$(y+z) + (\bar{y}+z) + \bar{y} \cdot \bar{z}$
00	0	1	1	1	1
01	1	0	1	0	1
10	1	0	1	0	1
11	1	0	1	0	1

Espressione Booleana associata al terminale output della Rete Combinatoria

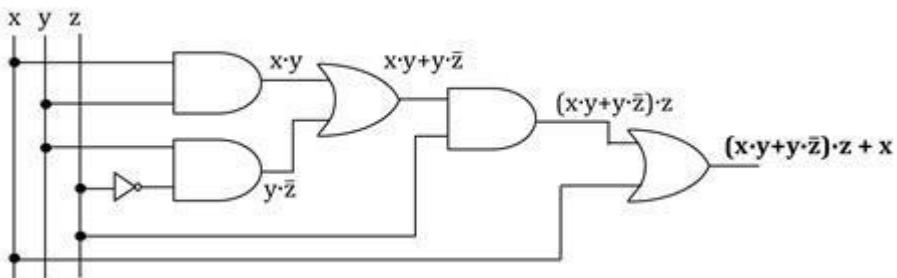
Funzione Booleana $f(x,y)$ associata alla Rete Combinatoria:

yz	$y+z$	$\bar{y}+z$	$(y+z) \cdot (\bar{y}+z)$	$\bar{y} \cdot \bar{z}$	$(y+z) \cdot (\bar{y}+z) + \bar{y} \cdot \bar{z}$
00	0	1	0	0	0
01	1	0	0	1	1
10	1	0	0	1	1
11	1	0	0	1	1

$$(x \cdot y + y \cdot \bar{z}) \cdot z + x$$

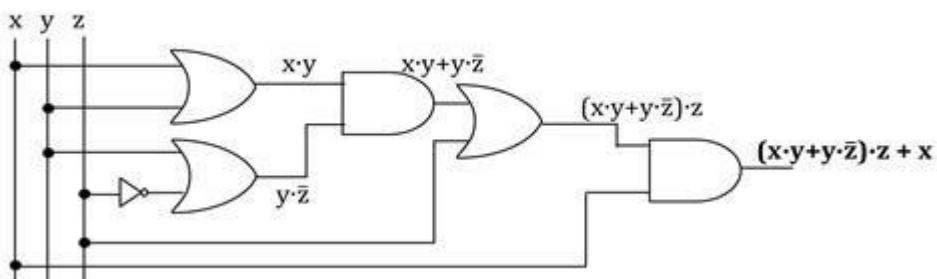
1

Espressione Booleana associata al terminale output della Rete Combinatoria



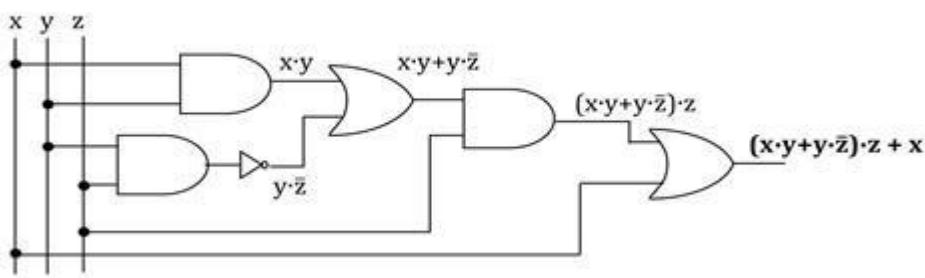
2

Espressione Booleana associata al terminale output della Rete Combinatoria



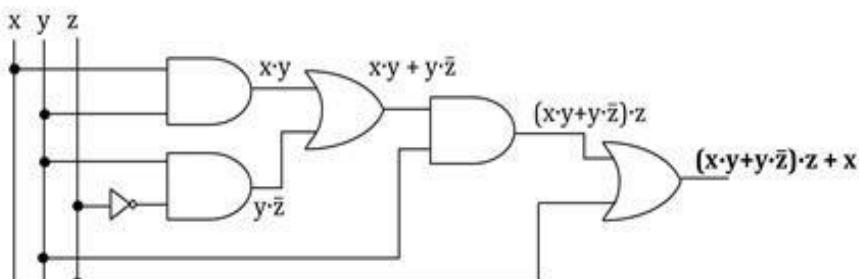
3

Espressione Booleana associata al terminale output della Rete Combinatoria



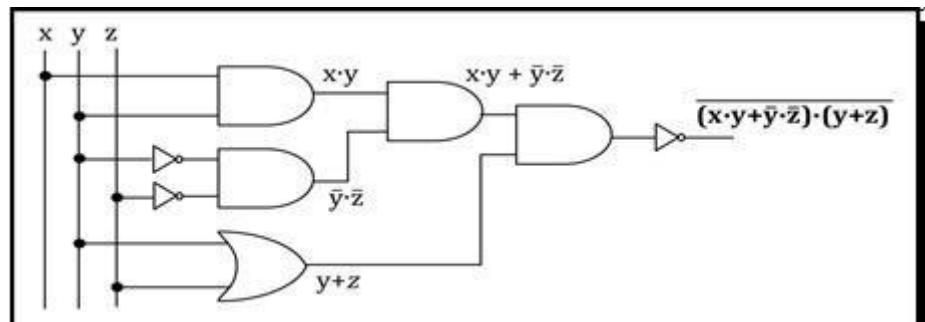
4

Espressione Booleana associata al terminale output della Rete Combinatoria

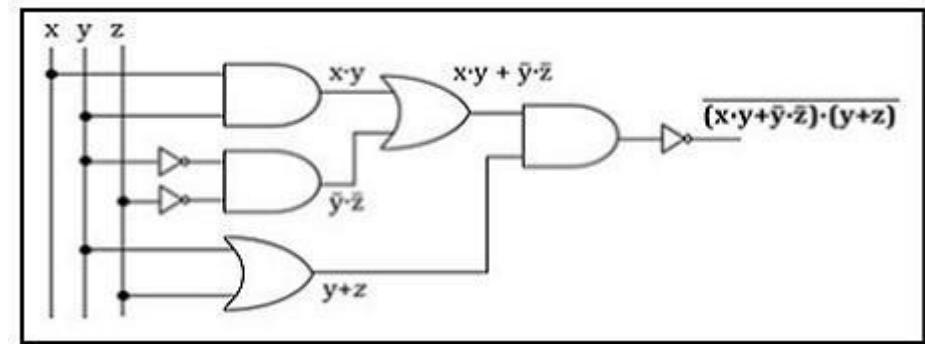


$$(x \cdot y + \bar{y} \cdot \bar{z}) \cdot (y + z)$$

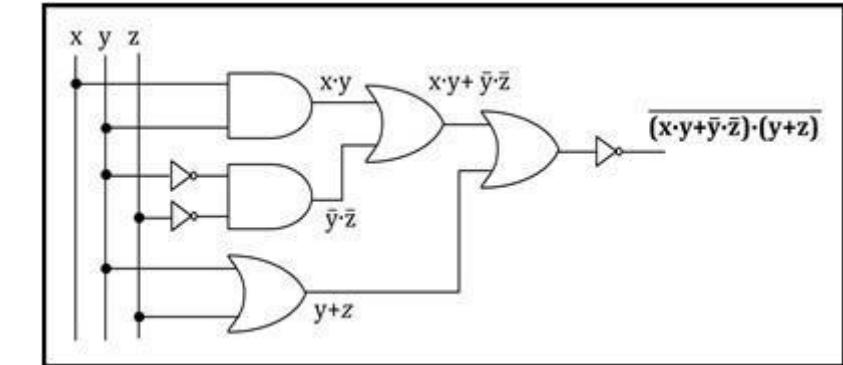
1



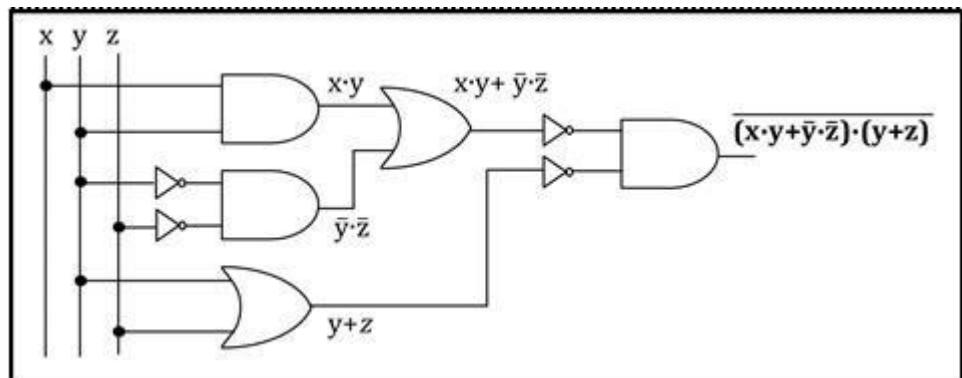
2



3

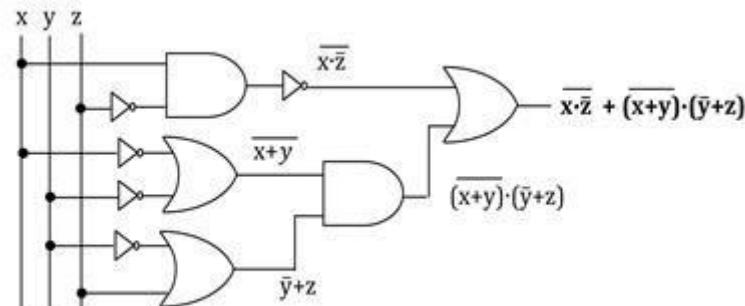


4

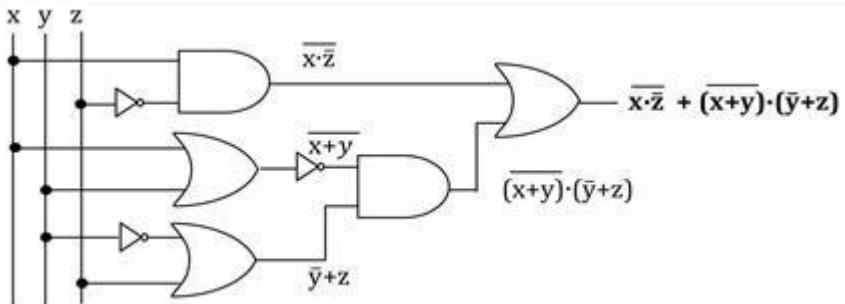


$$\overline{x \cdot \bar{z}} + (\overline{x+y}) \cdot (\bar{y}+z)$$

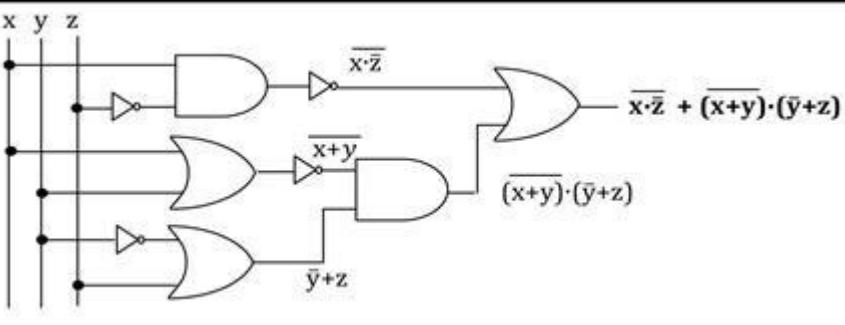
1



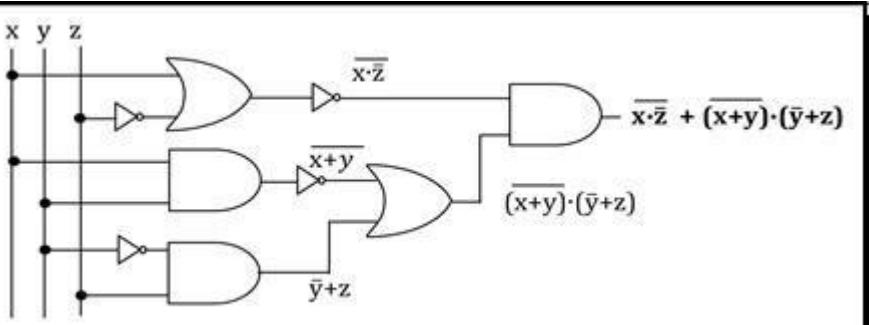
2



3



4



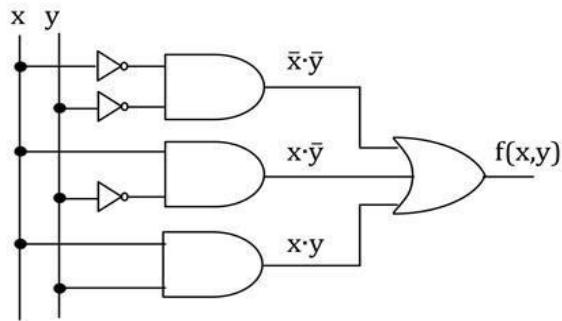
x	y	$f(x,y)$
0	0	1
0	1	0
1	0	1
1	1	1

2

1 Espressione Somma di Prodotti associata

$$f(x,y,z) = \bar{x} \cdot \bar{y} + x \cdot \bar{y} + x \cdot y$$

Rete Combinatoria AND to OR equivalente

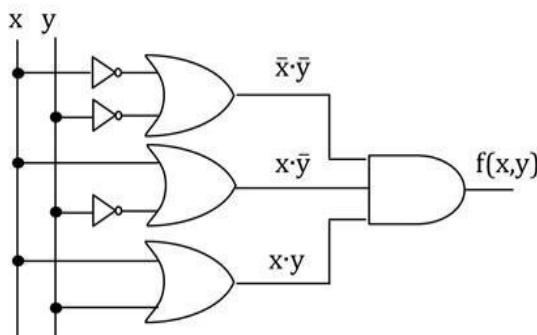


3

Espressione Somma di Prodotti associata

$$f(x,y,z) = \bar{x} \cdot \bar{y} + x \cdot \bar{y} + x \cdot y$$

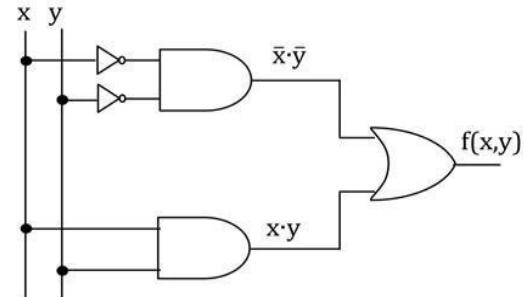
Rete Combinatoria AND to OR equivalente



Espressione Somma di Prodotti associata

$$f(x,y,z) = \bar{x} \cdot \bar{y} + x \cdot y$$

Rete Combinatoria AND to OR equivalente

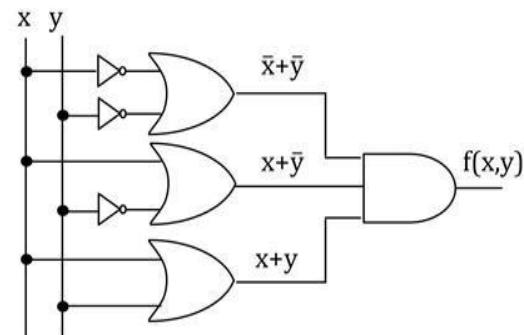


4

Espressione Somma di Prodotti associata

$$f(x,y,z) = (\bar{x} + \bar{y}) \cdot (x + \bar{y}) \cdot (x + y)$$

Rete Combinatoria AND to OR equivalente

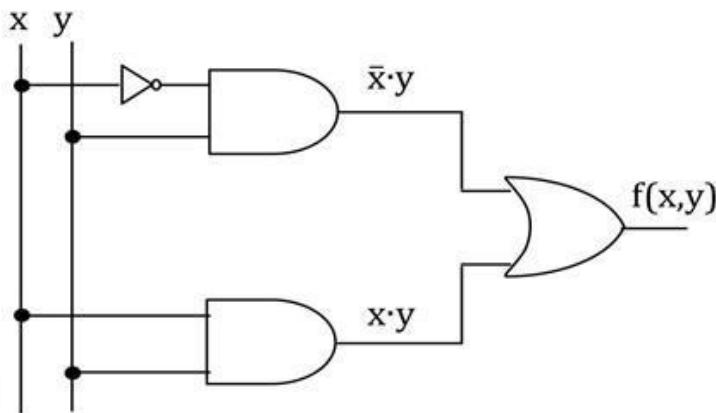


x	y	f(x,y)
00		0
01		1
10		1
11		0

1

Espressione Somma di Prodotti associata

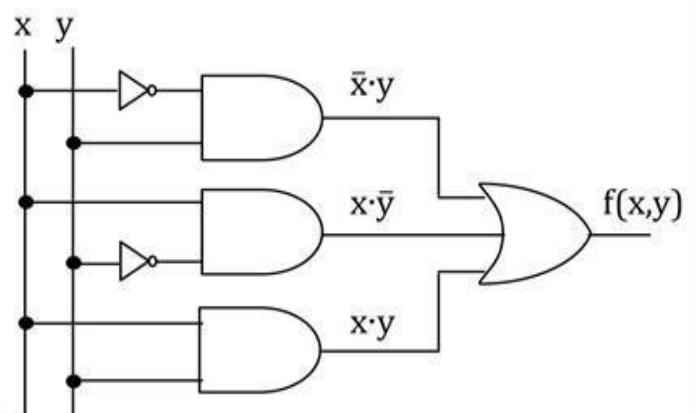
$$f(x,y,z) = \bar{x} \cdot y + x \cdot y$$

Rete Combinatoria AND to OR equivalente

2

Espressione Somma di Prodotti associata

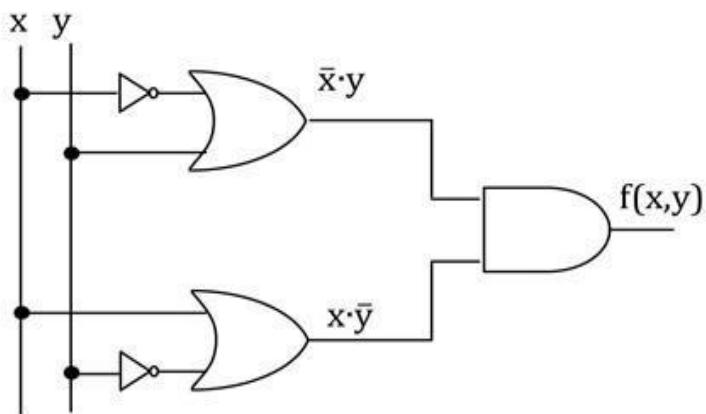
$$f(x,y,z) = \bar{x} \cdot y + x \cdot \bar{y} + x \cdot y$$

Rete Combinatoria AND to OR equivalente

3

Espressione Somma di Prodotti associata

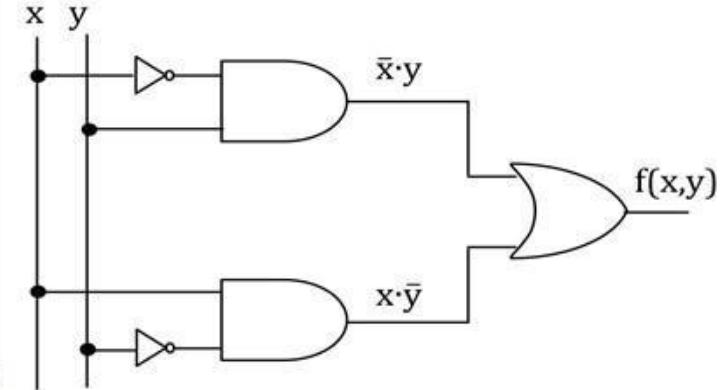
$$f(x,y,z) = \bar{x} \cdot \bar{y} + x \cdot \bar{y}$$

Rete Combinatoria AND to OR equivalente

4

Espressione Somma di Prodotti associata

$$f(x,y,z) = \bar{x} \cdot y + x \cdot \bar{y}$$

Rete Combinatoria AND to OR equivalente

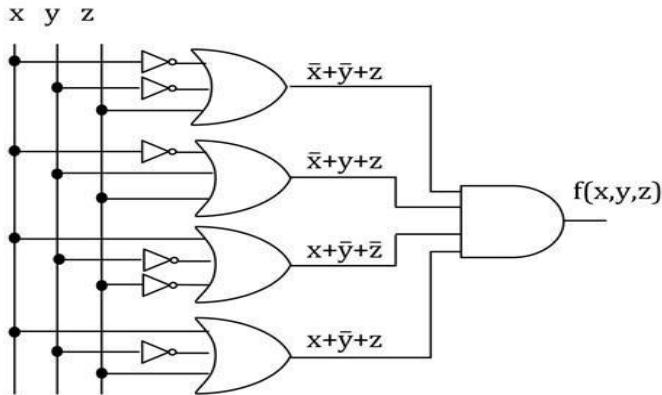
x y z	f(x,y,z)
000	0
001	1
010	0
011	1
100	1
101	1
110	0
111	0

1

2

Espressione Somma di Prodotti associata

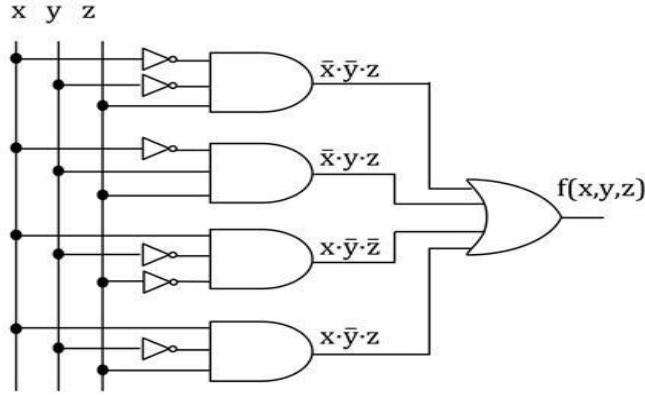
$$f(x,y,z) = (\bar{x} + \bar{y} + z) \cdot (\bar{x} + y + z) \cdot (x + \bar{y} + \bar{z}) \cdot (x + \bar{y} + z)$$

Rete Combinatoria AND to OR equivalente

3

Espressione Somma di Prodotti associata

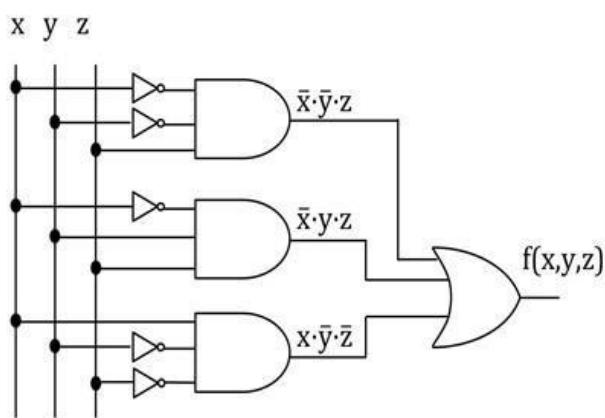
$$f(x,y,z) = \bar{x} \cdot \bar{y} \cdot z + \bar{x} \cdot y \cdot z + x \cdot \bar{y} \cdot \bar{z} + x \cdot \bar{y} \cdot z$$

Rete Combinatoria AND to OR equivalente

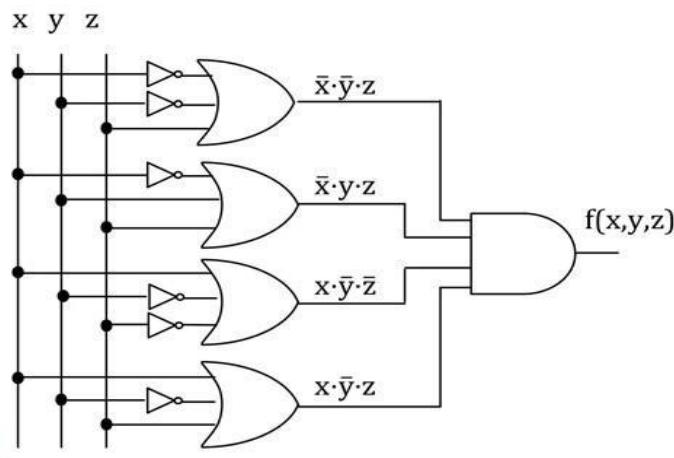
4

Espressione Somma di Prodotti associata

$$f(x,y,z) = \bar{x} \cdot \bar{y} \cdot z + \bar{x} \cdot y \cdot z + x \cdot \bar{y} \cdot \bar{z}$$

Rete Combinatoria AND to OR equivalente*Espressione Somma di Prodotti associata*

$$f(x,y,z) = \bar{x} \cdot \bar{y} \cdot z + \bar{x} \cdot y \cdot z + x \cdot \bar{y} \cdot \bar{z} + x \cdot \bar{y} \cdot z$$

Rete Combinatoria AND to OR equivalente

x y z	f(x,y,z)
000	0
001	0
010	1
011	0
100	0
101	0
110	1
111	0

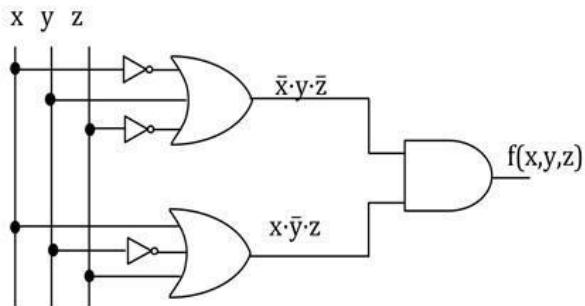
1

2

Espressione Somma di Prodotti associata

$$f(x,y,z) = \bar{x} \cdot y \cdot \bar{z} + x \cdot \bar{y} \cdot z$$

Rete Combinatoria AND to OR equivalente

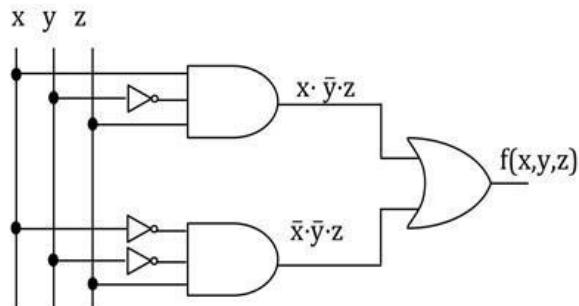


3

Espressione Somma di Prodotti associata

$$f(x,y,z) = x \cdot \bar{y} \cdot z + \bar{x} \cdot \bar{y} \cdot z$$

Rete Combinatoria AND to OR equivalente

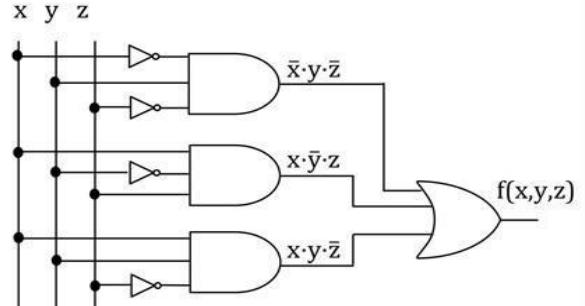


4

Espressione Somma di Prodotti associata

$$f(x,y,z) = \bar{x} \cdot y \cdot \bar{z} + x \cdot \bar{y} \cdot z + x \cdot y \cdot \bar{z}$$

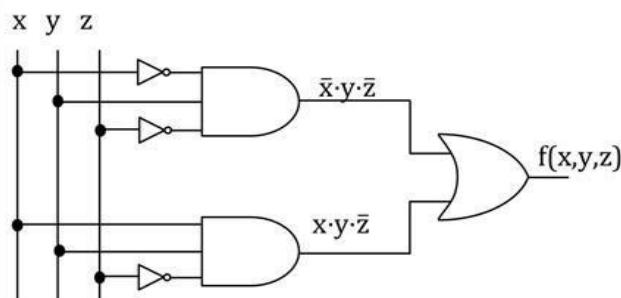
Rete Combinatoria AND to OR equivalente



Espressione Somma di Prodotti associata

$$f(x,y,z) = \bar{x} \cdot y \cdot \bar{z} + x \cdot y \cdot \bar{z}$$

Rete Combinatoria AND to OR equivalente



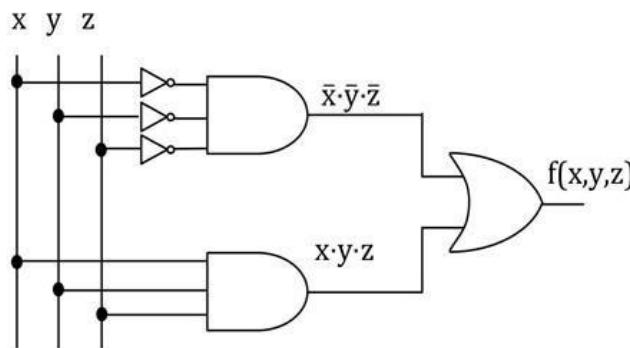
x	y	z	f(x,y,z)
000			1
001			0
010			0
011			0
100			0
101			0
110			1
111			1

1

2

Espressione Somma di Prodotti associata

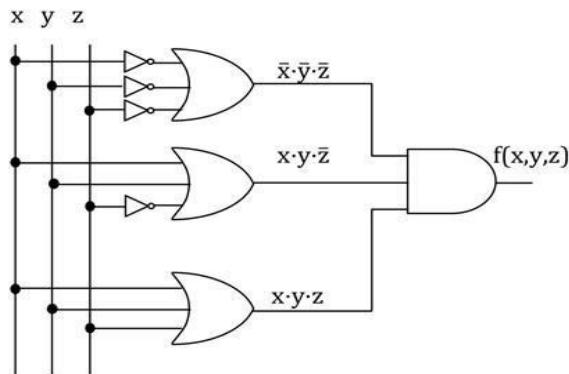
$$f(x,y,z) = \bar{x} \cdot \bar{y} \cdot \bar{z} + x \cdot y \cdot z$$

Rete Combinatoria AND to OR equivalente

3

Espressione Somma di Prodotti associata

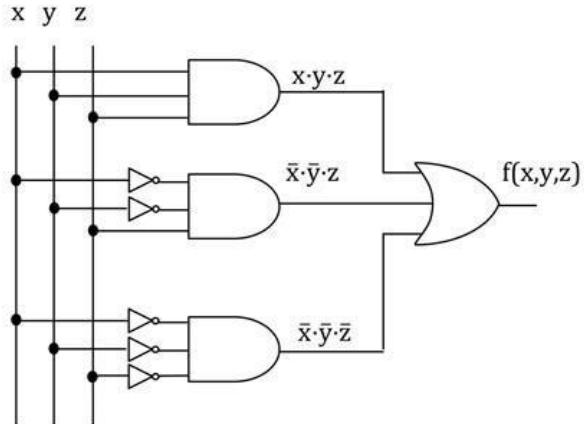
$$f(x,y,z) = \bar{x} \cdot \bar{y} \cdot \bar{z} + x \cdot y \cdot \bar{z} + x \cdot y \cdot z$$

Rete Combinatoria AND to OR equivalente

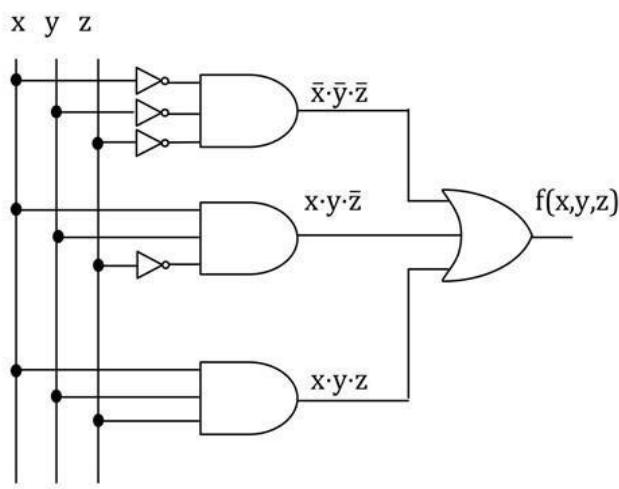
4

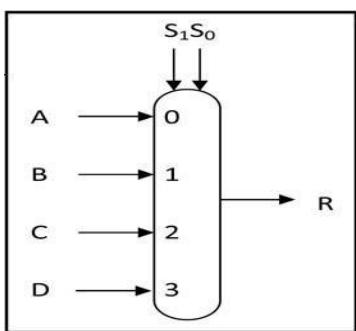
Espressione Somma di Prodotti associata

$$f(x,y,z) = x \cdot y \cdot z + \bar{x} \cdot \bar{y} \cdot z + \bar{x} \cdot \bar{y} \cdot \bar{z}$$

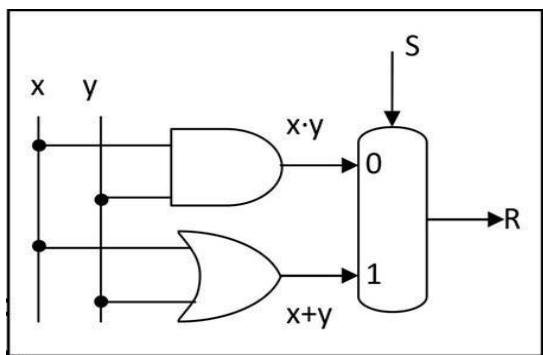
Rete Combinatoria AND to OR equivalente*Espressione Somma di Prodotti associata*

$$f(x,y,z) = \bar{x} \cdot \bar{y} \cdot \bar{z} + x \cdot y \cdot \bar{z} + x \cdot y \cdot z$$

Rete Combinatoria AND to OR equivalente

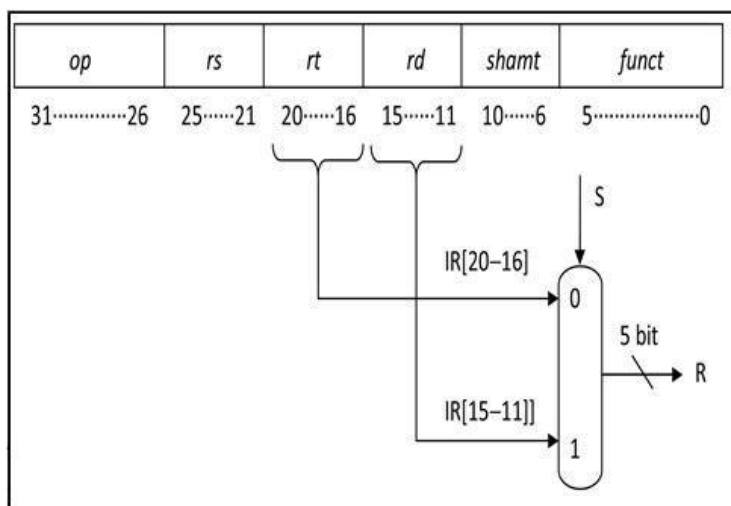


- A. Un Decodificatore 2 a 2² = 4
 B. Un Decodificatore 4 a 2⁴ = 16
 C. Un Multiplexer 2⁴=16 a 1
 D. Un Multiplexer 2² = 4 a 1

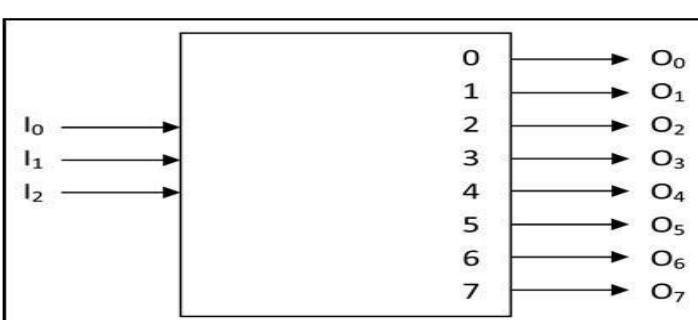


in figura realizza la funzione:

- A. Se S=0 allora R=x·y, se S=1 allora R=x+y
 B. Se x=0 allora R=x·y, se x=1 allora R=x+y
 C. Se y=0 allora R=x·y, se y=1 allora R=x+y
 D. Se S=0 allora R=x+y, se S=1 allora R= x-y



- A. Se x=0 allora R=IR[20,16], se x=1 allora R=IR[15,11]
 B. Se y=0 allora R=IR[20,16], se y=1 allora R=IR[15,11]
 C. Se S=0 allora R=IR[20,16], se S=1 allora R=IR[15,11]
 D. Se S=0 allora R=IR[15,11], se S=1 allora R=IR[20,16]



Il simbolo grafico in figura rappresenta:

- A. Un Decodificatore 3 a 23 = 8
- B. Un Multiplexer 23 = 8 a 1
- C. Un Multiplexer 28 a 1
- D. Un Decodificatore 8 a 28

Addizione sulla singola posizione					
INPUT			OUTPUT		
Operando a	Operando b	Riporto input CarryIn	Riporto output CarryOut	Risultato Result	
0	0	0	0	0	
0	0	1	0	1	
0	1	0	0	1	
0	1	1	1	0	
1	0	0	0	1	
1	0	1	1	0	
1	1	0	1	0	
1	1	1	1	1	

L'Espressione Booleana in forma Somma di Prodotti della funzione Result che fornisce il bit del Risultato rappresentata nella Tavola di verità dell'Addizione riportata in figura è data da:

$$\text{Result} = \bar{a} \cdot \bar{b} \cdot \overline{\text{CarryIn}} + \bar{a} \cdot b \cdot \overline{\text{CarryIn}} + a \cdot \bar{b} \cdot \overline{\text{CarryIn}} + a \cdot b \cdot \text{CarryIn}$$

A.

$$\text{Result} = \bar{a} \cdot \bar{b} \cdot \text{CarryIn} + \bar{a} \cdot b \cdot \overline{\text{CarryIn}} + a \cdot \bar{b} \cdot \overline{\text{CarryIn}} + a \cdot b \cdot \text{CarryIn}$$

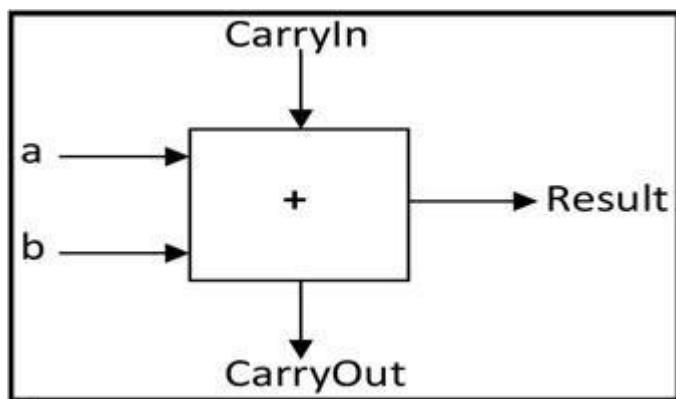
B.

$$\text{Result} = \bar{a} \cdot \bar{b} \cdot \text{CarryIn} + \bar{a} \cdot b \cdot \text{CarryIn} + a \cdot \bar{b} \cdot \overline{\text{CarryIn}} + a \cdot b \cdot \text{CarryIn}$$

C.

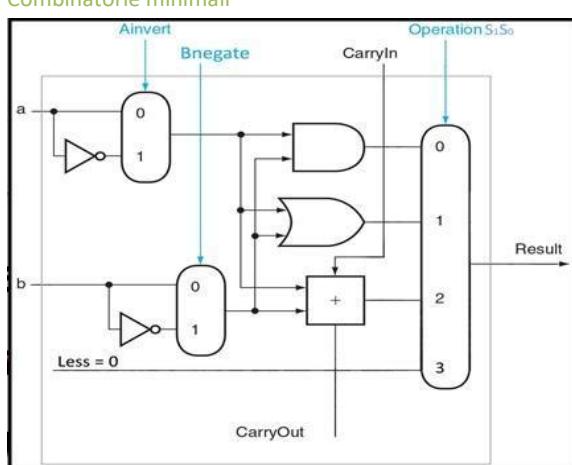
$$\text{Result} = \bar{a} \cdot \bar{b} \cdot \text{CarryIn} + \bar{a} \cdot b \cdot \overline{\text{CarryIn}} + a \cdot \bar{b} \cdot \text{CarryIn} + a \cdot b \cdot \overline{\text{CarryIn}}$$

D.



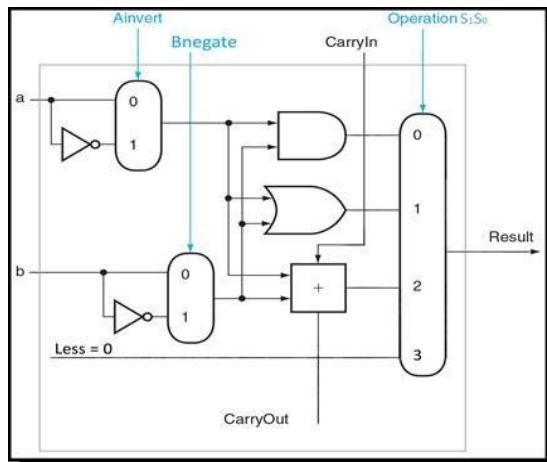
Il simbolo grafico riportato in figura rappresenta:

- A. Un multiplexer 2 a 1
- B. L'ALU a un bit relativa alla singola posizione della sequenza
- C. L'ALU a 32 bit
- D. Il Sommatore completo che calcola l'Addizione dei bit relativi alla singola posizione della sequenza binaria, costruito utilizzando le Reti Combinatorie minimali



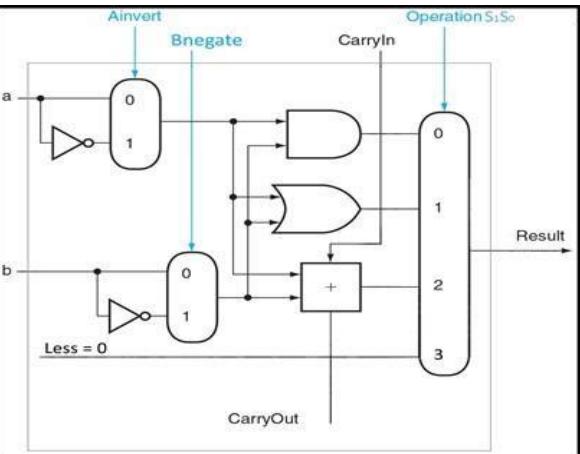
La Rete Combinatoria in figura rappresenta:

- A. L'ALU ad un bit relativa alla singola posizione della sequenza binaria
- B. Il Sommatore completo che calcola l'Addizione dei bit relativi alla singola posizione della sequenza binaria
- C. L'ALU a 32 bit
- D. Un multiplexer 2 a 1



Nell'ALU a un bit in figura, per selezionare il risultato delle istruzioni AND e OR ai due bit del segnale di controllo OperationS1S0 sono assegnati i valori:

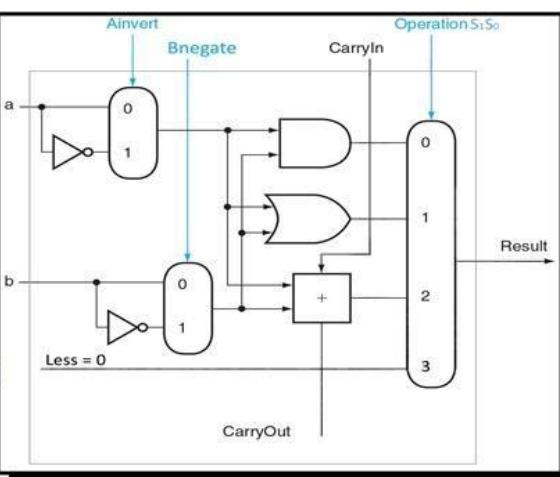
- A. OperationS1S0=00 per AND, e OperationS1S0=01 per OR
- B. OperationS1S0=01 per AND, e OperationS1S0=00 per OR
- C. OperationS1S0=10 per AND, e OperationS1S0=11 per OR
- D. OperationS1S0=01 per AND, e OperationS1S0=10 per OR



1. <immagine>

Nell'ALU a un bit in figura, per selezionare il risultato dell'istruzione ADD ai due bit del segnale di controllo OperationS1S0 sono assegnati i valori:

- A. OperationS1S0=11
- B. OperationS1S0=00
- C. OperationS1S0=01
- D. OperationS1S0=10



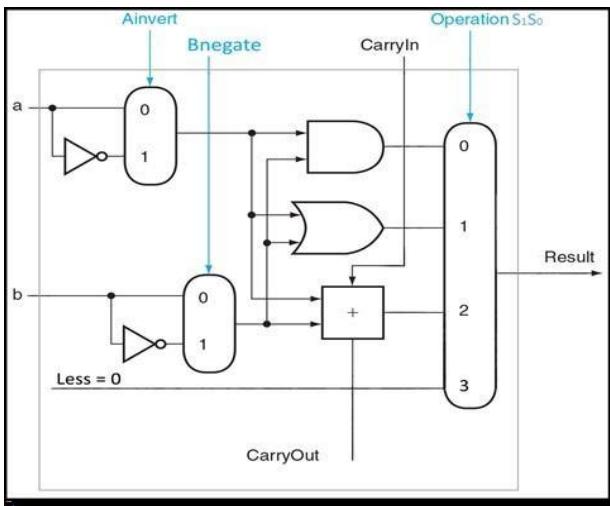
Il segnale output denotato con CarryOut nella Rete Combinatoria in figura è inviato:

- A. All'ALU ad un bit relativa alla posizione successiva come segnale di controllo del multiplexer 2 a 1 per la scelta della forma dell'operando da utilizzare

B. All'ALU ad un bit relativa alla posizione successiva come segnale di controllo del multiplexer 4 a 1 per la scelta del risultato

C. All'ALU ad un bit relativa alla posizione successiva come riporto input CarryIn0

D. Come output dell'ALU a 32 bit



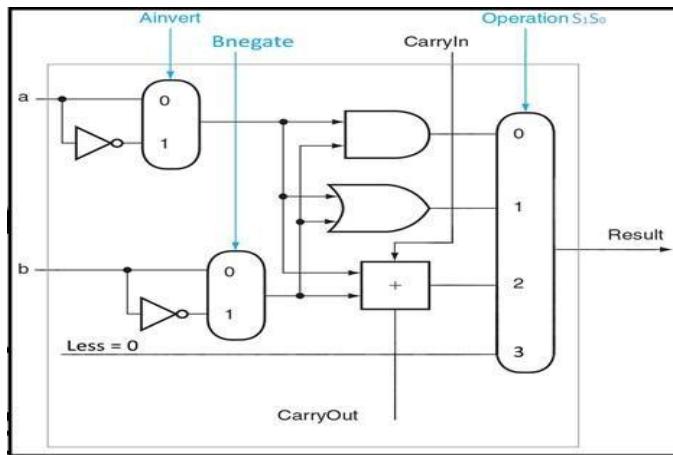
Nel circuito dell'ALU ad un bit riportato in figura, per l'esecuzione dell'istruzione SUB l'opposto del secondo operando si ottiene:

A. Ponendo a 1 il segnale di controllo Bnegate per selezionare il valore dell'operando complementato, e ponendo a 1 il segnale di controllo CarryIn0 dell'ALU ad un bit relativa alla posizione meno significativa per sommare il valore 1 alla sequenza complementata

B. Ponendo a 1 segnale di controllo Bnegate per selezionare il valore dell'operando complementato, e ponendo a 1 il segnale di controllo Ainvert per sommare il valore 1 alla sequenza complementata

C. Ponendo a 1 segnale di controllo Bnegate per selezionare il valore dell'operando complementato, e ponendo a 0 il riporto input CarryIn0 dell'ALU ad un bit relativa alla posizione meno significativa

D. Ponendo a 1 segnale di controllo Bnegate per selezionare il valore dell'operando complementato, e ponendo il segnale di controllo OperationS1S0=01 per sommare il valore 1 alla sequenza complementata



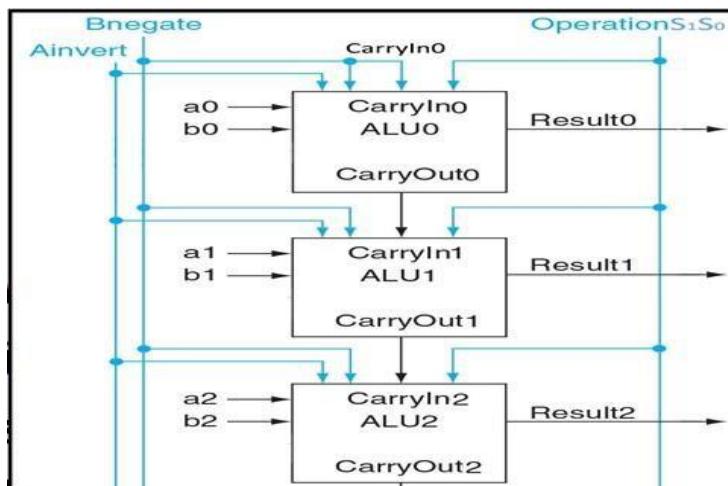
Nel circuito dell'ALU ad un bit riportato in figura, la selezione mediante il multiplexer 4 a 1 del risultato dell'esecuzione dell'istruzione SUB si realizza:

A. Ponendo il valore del segnale di controllo OperationS1S0=00 per scegliere l'output del Sommatore

B. Ponendo il valore del segnale di controllo OperationS1S0=01 per scegliere l'output del Sommatore

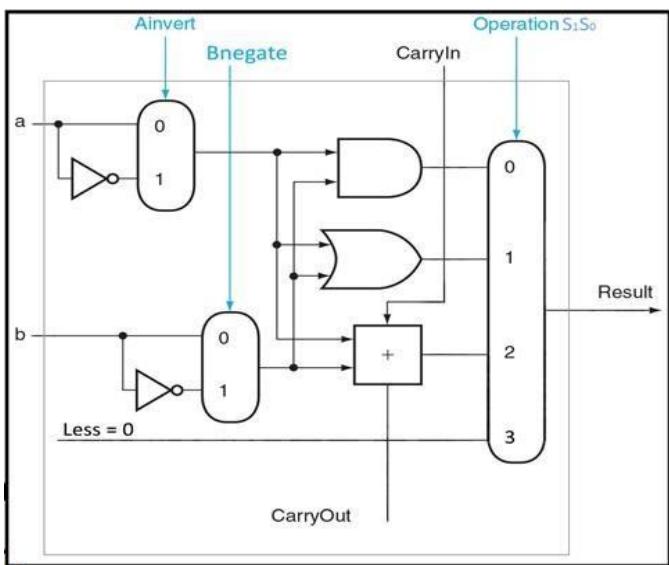
C. Ponendo il valore del segnale di controllo OperationS1S0=10 per scegliere l'output del Sommatore

D. Ponendo il valore del segnale di controllo OperationS1S0=11 per scegliere l'output del Sommatore



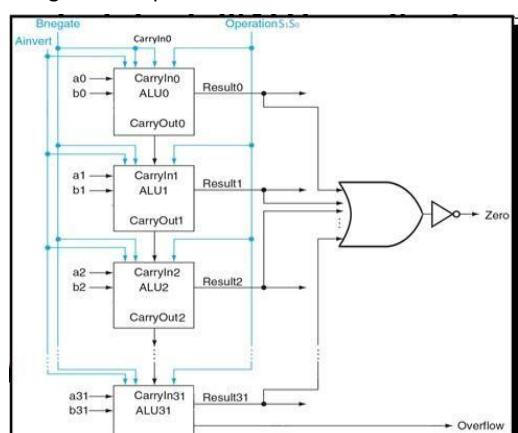
Nel circuito dell'ALU a 32 bit riportato in figura, il segnale di controllo Bnegate fornisce anche il valore del segnale di controllo CarryIn0 del riporto input dell'ALU a un bit relativa alla posizione meno significativa perché:

- A. Il valore del segnale CarryIn0 viene cambiato dal segnale Operation in base all'istruzione da eseguire
- B. Il valore del segnale Binvert viene cambiato dal segnale Operation in base all'istruzione da eseguire
- C. Il segnale CarryIn0 influenza solo le operazioni Aritmetiche che coinvolgono il Sommatore e in questi casi ha lo stesso valore del segnale Bnegate; negli altri casi il risultato non dipende dal valore di CarryIn0
- D. I valori assunti dai due segnali sono sempre uguali per tutte le operazione eseguite dall'ALU



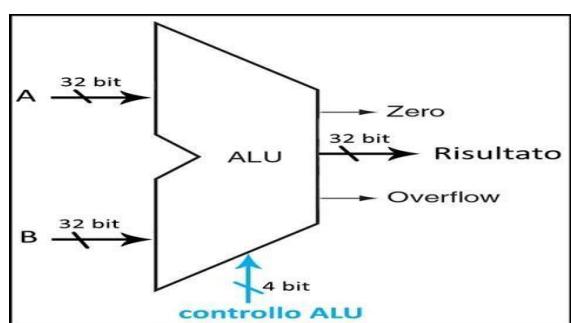
Nel circuito dell'ALU a un bit in figura, il risultato dell'"istruzione NOR si ottiene:

- A. Ponendo Ainvert=1 e Bnegate=1 per complementare bit a bit gli operandi, ed il segnale di controllo OperationS1S0=00 per scegliere l'output della porta AND
- B. Ponendo Ainvert=0 e Bnegate=0 per complementare bit a bit gli operandi, ed il segnale di controllo OperationS1S0=00 per scegliere l'output della porta AND
- C. Ponendo Ainvert=1 e Bnegate=1 per complementare bit a bit gli operandi, ed il segnale di controllo OperationS1S0=10 per scegliere l'output del Sommatore
- D. Ponendo Ainvert=0 e Bnegate=0 per complementare bit a bit gli operandi, ed il segnale di controllo OperationS1S0=10 per scegliere l'output del Sommatore



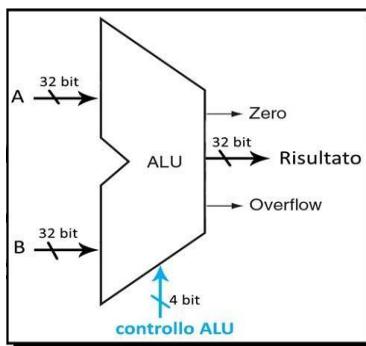
Nell'ALU a 32 bit in figura quando il segnale output Zero vale 1 si ha che:

- A. Nella esecuzione dell'istruzione ADD si è determinato un Errore di Overflow
- B. Nella esecuzione dell'istruzione BEQ di salto condizionato su uguaglianza il salto non viene effettuato perché gli operandi risultano diversi
- C. Nella esecuzione dell'istruzione BEQ di salto condizionato su uguaglianza il salto viene effettuato perché gli operandi risultano uguali
- D. Nella esecuzione dell'istruzione set on less then la relazione di minore tra gli operandi non è verificata



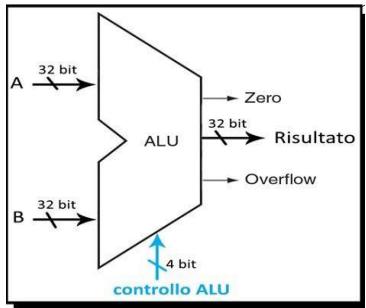
Nel simbolo grafico dell'ALU in figura, i 4 bit del segnale indicato col nome controllo ALU, che fornisce i valori dei segnali Ainvert, Bnegate, OperationS1S0, per l'esecuzione dell'istruzione ADD sono posti uguali a:

- A. Controllo ALU = 1100
- B. Controllo ALU = 0101
- C. Controllo ALU = 0000
- D. Controllo ALU = 0010



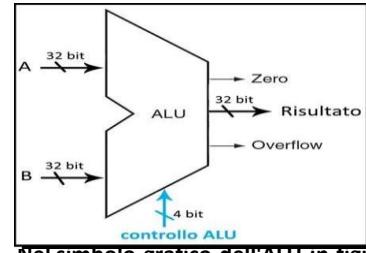
Nel simbolo grafico dell'ALU in figura, i 4 bit del segnale indicato col nome controllo ALU, che fornisce i valori dei segnali Ainvert, Bnegate, OperationS1S0, per l'esecuzione dell'istruzione SUB sono posti uguali a:

- A. Controllo ALU = 0010
- B. Controllo ALU = 0110
- C. Controllo ALU = 0000
- D. Controllo ALU = 0001



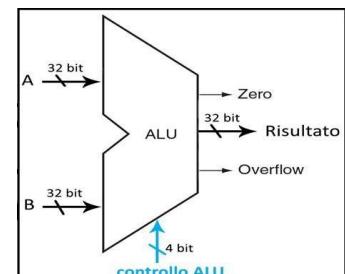
Nel simbolo grafico dell'ALU in figura, i 4 bit del segnale indicato col nome controllo ALU, che fornisce i valori dei segnali Ainvert, Bnegate, OperationS1S0, per l'esecuzione dell'istruzione AND sono posti uguali a:

- A. Controllo ALU = 0010
- B. Controllo ALU = 0000
- C. Controllo ALU = 0001
- D. Controllo ALU = 1100



Nel simbolo grafico dell'ALU in figura, i 4 bit del segnale indicato col nome controllo ALU, che fornisce i valori dei segnali Ainvert, Bnegate, OperationS1S0, per l'esecuzione dell'istruzione OR sono posti uguali a:

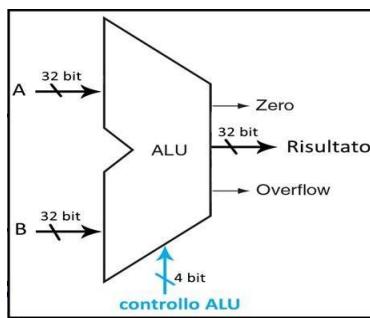
- A. Controllo ALU = 0001
- B. Controllo ALU = 1100
- C. Controllo ALU = 0110
- D. Controllo ALU = 0000



Nel simbolo grafico dell'ALU in figura, i 4 bit del segnale indicato col nome controllo ALU, che fornisce i valori dei segnali Ainvert, Bnegate,

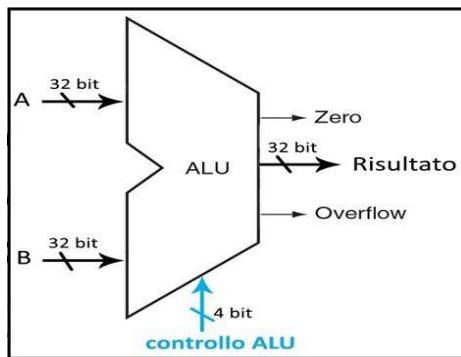
OperationS1SO, per l'esecuzione dell'istruzione NOR sono posti uguali a:

- A. Controllo ALU = 0000
- B. Controllo ALU = 0010
- C. Controllo ALU = 0110
- D. Controllo ALU = 1100



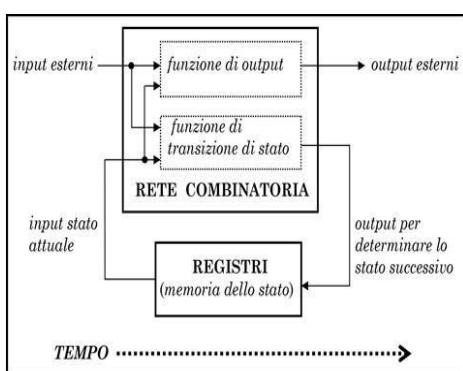
Nel simbolo grafico dell'ALU in figura, i 4 bit del segnale indicato col nome controllo ALU, che fornisce i valori dei segnali Ainvert, Bnegate, OperationS1SO, per l'esecuzione dell'istruzione BEQ sono posti uguali a:

- A. Controllo ALU = 0110
- B. Controllo ALU = 0010
- C. Controllo ALU = 1100
- D. Controllo ALU = 0000



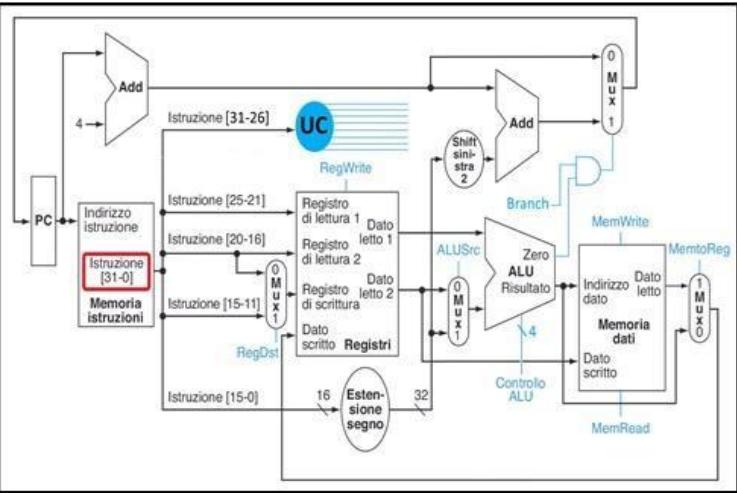
Nel simbolo grafico dell'ALU in figura, i 4 bit del segnale indicato col nome controllo ALU, che fornisce i valori dei segnali Ainvert, Bnegate, OperationS1SO, per l'esecuzione delle istruzioni SUB e BEQ sono:

- A. Diversi perché SUB ha Formato di Tipo R e BEQ ha Formato di Tipo I
- B. Diversi perché per SUB si scrive il Risultato nel registro destinazione e per BEQ si utilizza il segnale Zero calcolato dalla porta NOR
- C. Diversi perché SUB è una operazione Aritmetico-Logica e BEQ è una istruzione di salto
- D. Uguali perché l'ALU effettua in entrambi i casi una sottrazione, ma per SUB si utilizza il Risultato e per BEQ si utilizza il segnale Zero calcolato in base al Risultato.



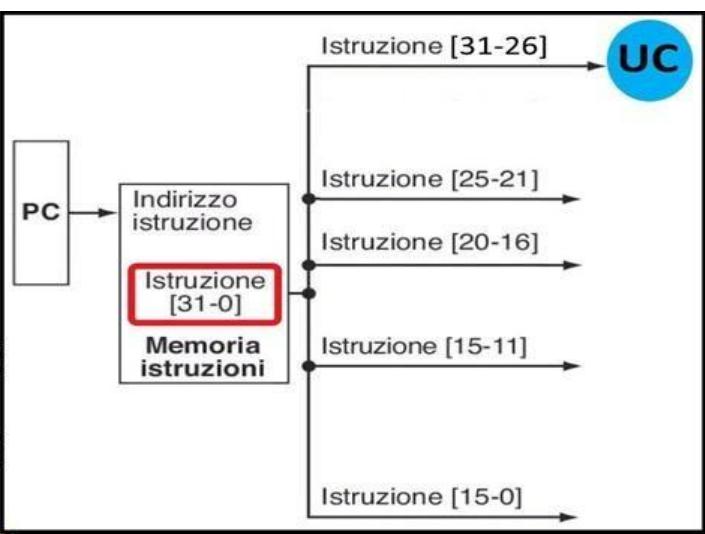
Lo schema in figura rappresenta la strutturazione di base del circuito che implementa:

- A. L'Unità Aritmetico-Logica (ALU)
- B. La Memoria del computer
- C. Una Rete Combinatoria
- D. Una Rete Sequentziale



Lo schema riportato in figura rappresenta:

- A. La struttura dei dispositivi di memorizzazione dell'Architettura MIPS
- B. La struttura del circuito dell'Unità di Controllo (UC) a ciclo singolo dell'Architettura MIPS per le istruzioni lw, sw, beq, Aritmetico-Logiche con Formato di Tipo R, e le sue connessioni con la Memoria
- C. La struttura del circuito dell'Unità Aritmetico-Logica (ALU) dell'Architettura MIPS, e le sue connessioni con la Memoria
- D. La struttura del circuito della Unità Centrale di Elaborazione (CPU) a ciclo singolo dell'Architettura MIPS per le istruzioni lw, sw, beq, Aritmetico-Logiche con Formato di Tipo R, e le sue connessioni con la Memoria



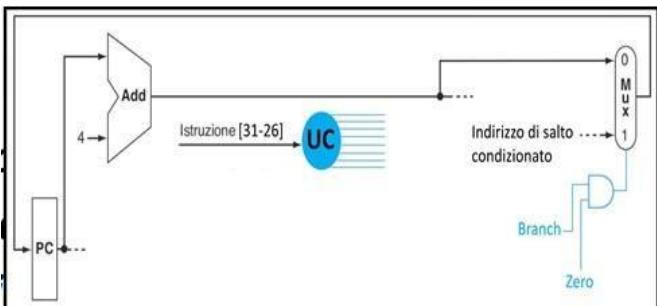
Lo schema in figura rappresenta la parte del circuito della CPU MIPS a ciclo singolo coinvolta in:

- A. Fase di Prelievo dell'istruzione da eseguire
- B. Aggiornamento del Program Counter con l'indirizzo della prossima istruzione da eseguire
- C. Lettura anticipata dei due Registri del processore i cui contenuti possono costituire operandi dell'istruzione in esecuzione
- D. Lettura di un operando dalla Memoria



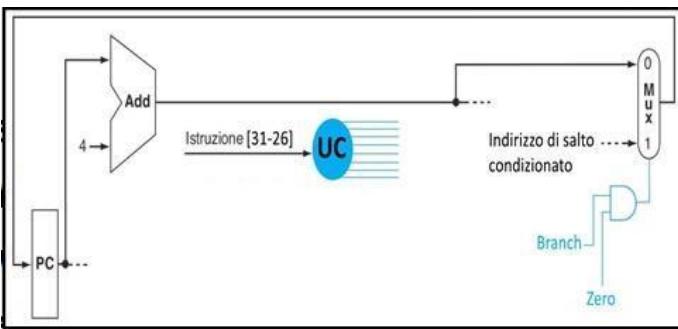
Nello schema in figura la notazione Istruzione[31-26] rappresenta:

- A. L'indirizzo dell'istruzione successiva che l'Unità di Controllo invia al Program Counter
- B. I 6 bit del campo Codice Operativo dell'istruzione in esecuzione inviati in input alla Unità di Controllo
- C. L'indirizzo del Registro da scrivere che l'Unità di Controllo invia al multiplexer
- D. Il dato da scrivere che l'Unità di Controllo invia al multiplexer



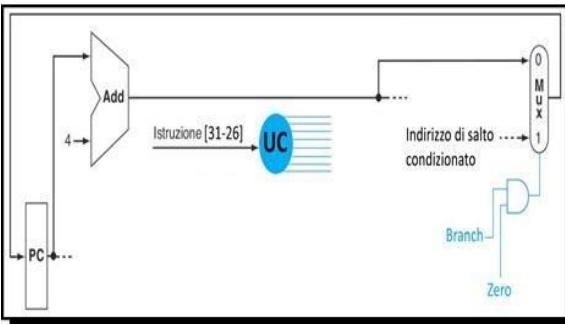
Lo schema in figura rappresenta la parte del circuito della CPU MIPS a ciclo singolo coinvolta in:

- Esecuzione dell'operazione di Addizione relativa all'istruzione add
- Fase di Prelievo dell'istruzione da eseguire
- Aggiornamento del Program Counter con l'indirizzo della prossima istruzione da eseguire
- Lettura anticipata dei due Registri del processore i cui contenuti possono costituire operandi dell'istruzione in esecuzione



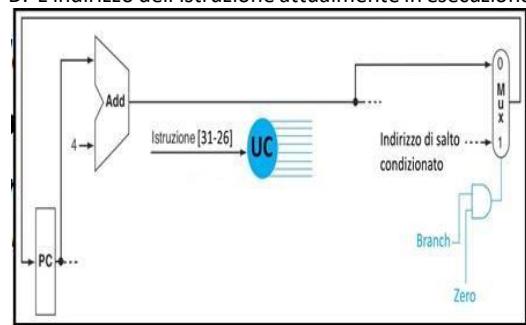
Nello schema in figura il multiplexer seleziona l'indirizzo con cui aggiornare il Program Counter scegliendo tra:

- Il risultato calcolato dall'ALU e il dato letto in una parola di memoria nell'esecuzione dell'istruzione lw
- L'indirizzo dell'istruzione successiva a quella in esecuzione e l'indirizzo di salto condizionato da usare nell'esecuzione dell'istruzione beq
- L'indirizzo dell'istruzione successiva a quella in esecuzione e l'indirizzo di accesso in Memoria da usare nell'esecuzione dell'istruzione sw
- L'indirizzo dell'istruzione successiva a quella in esecuzione e l'indirizzo di accesso in Memoria da usare nell'esecuzione dell'istruzione lw



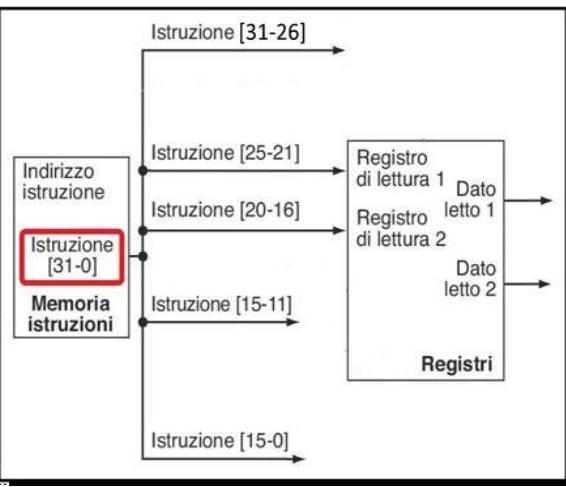
Nello schema in figura, quando si ha il valore del segnale di controllo Branch=0 il multiplexer seleziona:

- L'indirizzo calcolato dall'ALU inviato sull'ingresso dati 1
- L'indirizzo di salto condizionato inviato sull'ingresso dati 1 dalla parte del circuito della CPU che lo calcola
- L'indirizzo dell'istruzione successiva a quella in esecuzione calcolato dal Sommatore ed inviato sull'ingresso dati 0
- L'indirizzo dell'istruzione attualmente in esecuzione



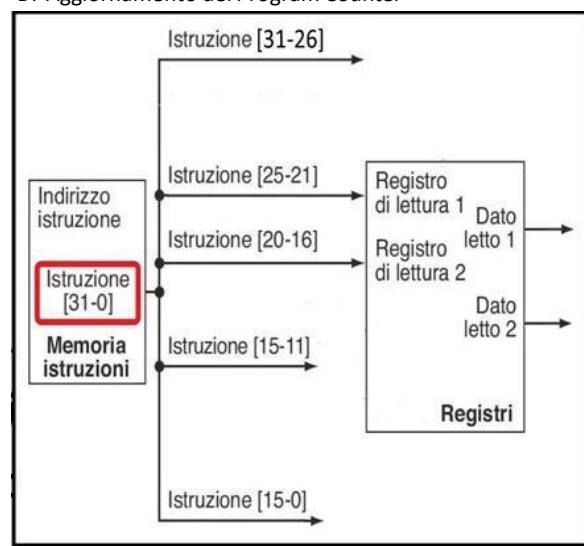
Nello schema in figura, quando si ha il valore del segnale di controllo Branch=1 e il segnale generato dall'ALU Zero=1 il multiplexer seleziona:

- L'indirizzo dell'istruzione attualmente in esecuzione
- L'indirizzo dell'istruzione successiva a quella in esecuzione calcolato dal Sommatore ed inviato sull'ingresso dati 0
- L'indirizzo calcolato dall'ALU inviato sull'ingresso dati 0
- L'indirizzo di salto condizionato inviato sull'ingresso dati 1 dalla parte del circuito della CPU che lo calcola



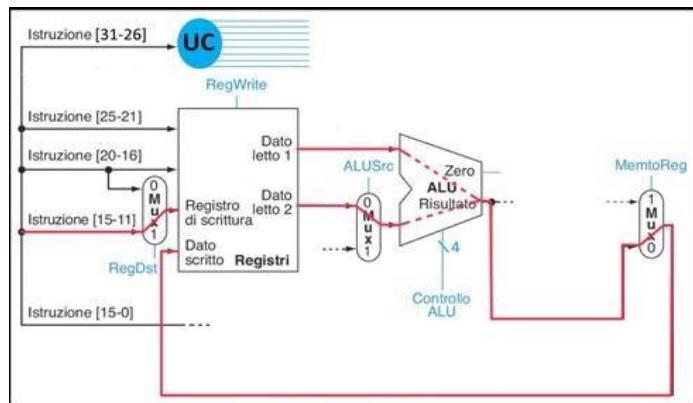
Lo schema in figura rappresenta la parte del circuito della CPU MIPS a ciclo singolo coinvolta in:

- A. Fase di prelievo dell'istruzione da eseguire
- B. Scrittura di un Registro del processore attivata dal valore 1 del segnale di controllo RegWrite
- C. Lettura anticipata dei due Registri del processore i cui contenuti possono costituire operandi dell'istruzione in esecuzione
- D. Aggiornamento del Program Counter



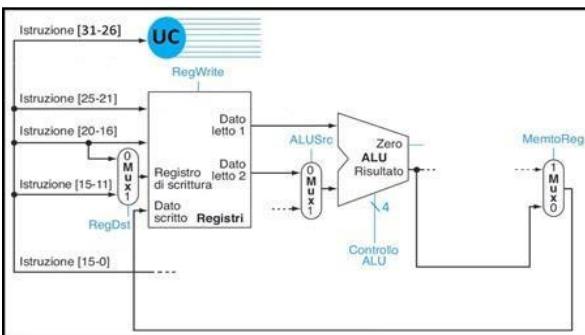
Nello schema in figura le notazioni Istruzioni[25-21] e Istruzioni[20-16] rappresentano:

- A. I due campi di 5 bit che forniscono gli indirizzi degli operandi da leggere anticipatamente nei Registri del processore
- B. I due campi che forniscono l'indirizzo ed il valore del dato da scrivere nel Registro del Processore
- C. L'indirizzo ed il valore del dato letto nel Registro del Processore
- D. I due valori degli operandi di 32 bit letti anticipatamente nei Registri del processore



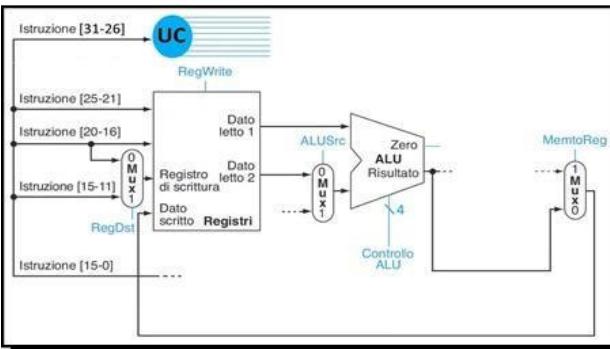
Lo schema in figura rappresenta la computazione che si svolge nella parte del circuito della CPU MIPS a ciclo singolo relativa a:

- A. Istruzione di salto condizionato BEQ
- B. Istruzioni di trasferimento dati Load word e Store word
- C. Istruzioni Aritmetico-Logiche di Tipo R
- D. Prelievo dell'istruzione da eseguire e lettura anticipata dei Registri



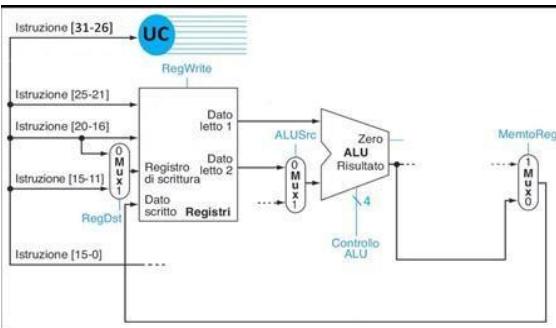
Nella parte della CPU a ciclo singolo relativa alle istruzioni Aritmetico-Logiche MIPS di Tipo R riportata in figura, il multiplexer controllato dal segnale AluSrc seleziona:

- A. L'indirizzo del Registro del processore in cui scrivere
- B. Il Dato da scrivere nel Registro del processore
- C. Il secondo operando dell'ALU
- D. L'operazione eseguita dall'ALU



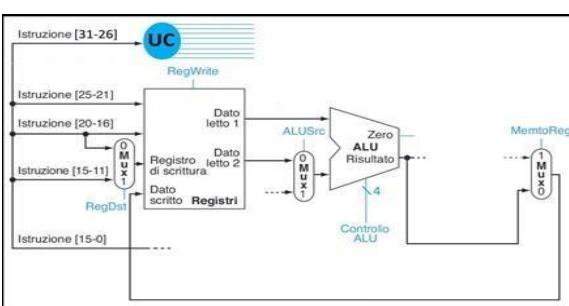
Nella parte della CPU a ciclo singolo relativa alle istruzioni Aritmetico-Logiche MIPS di Tipo R riportata in figura, il multiplexer controllato dal segnale MemtoReg seleziona:

- A. L'operazione eseguita dall'ALU
- B. Il secondo operando dell'ALU
- C. L'indirizzo del Registro del processore in cui scrivere
- D. Il Dato da scrivere nel Registro del processore



Nella parte della CPU a ciclo singolo relativa alle istruzioni Aritmetico-Logiche MIPS di Tipo R riportata in figura, il multiplexer controllato dal segnale RegDst seleziona:

- A. Il Dato da scrivere nel Registro del processore
- B. L'indirizzo del Registro del processore in cui scrivere
- C. Il secondo operando dell'ALU
- D. L'operazione eseguita dall'ALU

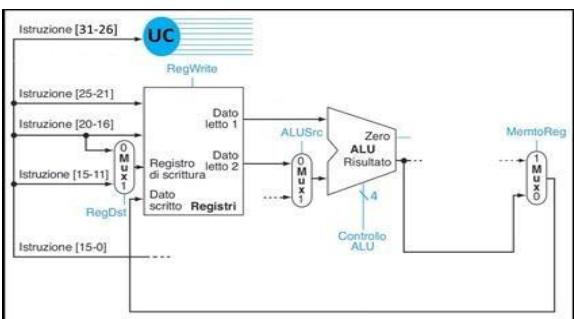


Nella parte della CPU a ciclo singolo relativa alle istruzioni Aritmetico-Logiche MIPS di Tipo R riportata in figura, con il valore del segnale di controllo MemtoReg=0 il multiplexer seleziona:

- A. Il risultato calcolato dall'ALU come Dato da scrivere nel registro
- B. Il contenuto del campo di 5 bit Istruzione[15-11] come indirizzo del Registro del processore in cui scrivere il Dato

C. Il contenuto del Registro del processore letto in anticipo disponibile sul termeminale output Dato letto 2 come secondo operando dell'ALU

D. L'operazione eseguita dall'ALU

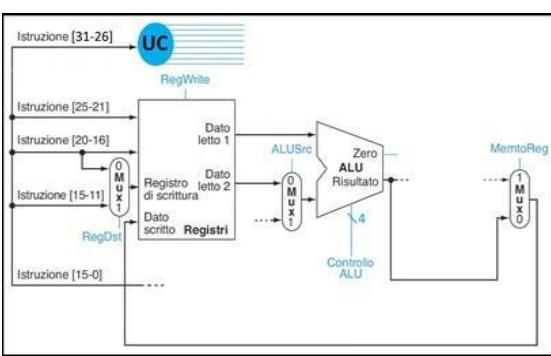


Nella parte della CPU a ciclo singolo relativa alle istruzioni Aritmetico-Logiche MIPS di Tipo R riportata in figura, con il valore del segnale di controllo **RegDst=1** il multiplexer seleziona:

- A. L'operazione eseguita dall'ALU
- B. Il risultato calcolato dall'ALU come Dato da scrivere nel registro

C. Il contenuto del campo di 5 bit Istruzione[15-11] come indirizzo del Registro del processore in cui scrivere il Dato

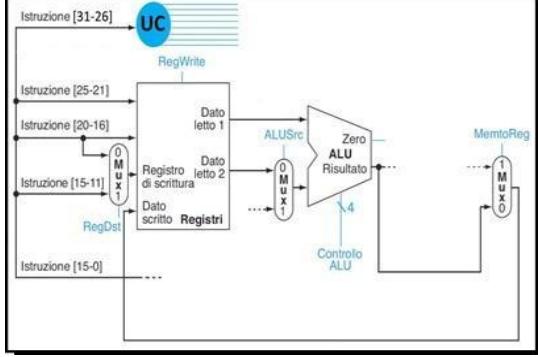
D. Il contenuto del Registro del processore letto in anticipo disponibile sul termeminale output Dato letto 2 come secondo operando dell'ALU



Nella parte della CPU a ciclo singolo relativa alle istruzioni Aritmetico-Logiche MIPS di Tipo R riportata in figura, con il valore del segnale di controllo **AluSrc=0** il multiplexer seleziona:

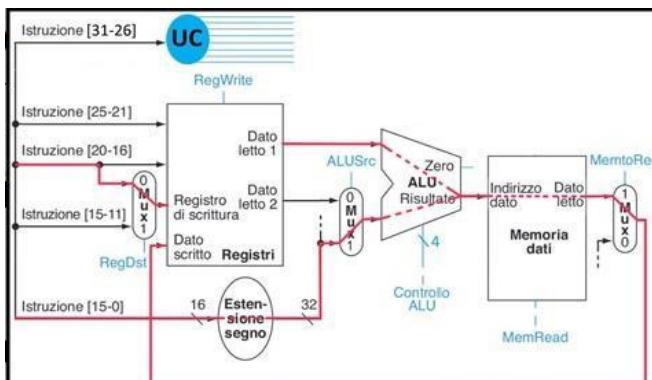
- A. L'operazione eseguita dall'ALU
- B. Il contenuto del campo di 5 bit Istruzione[15-11] come indirizzo del Registro del processore in cui scrivere il Dato
- C. Il risultato calcolato dall'ALU come Dato da scrivere nel registro

D. Il contenuto del Registro del processore letto in anticipo disponibile sul termeminale output Dato letto 2 come secondo operando dell'ALU



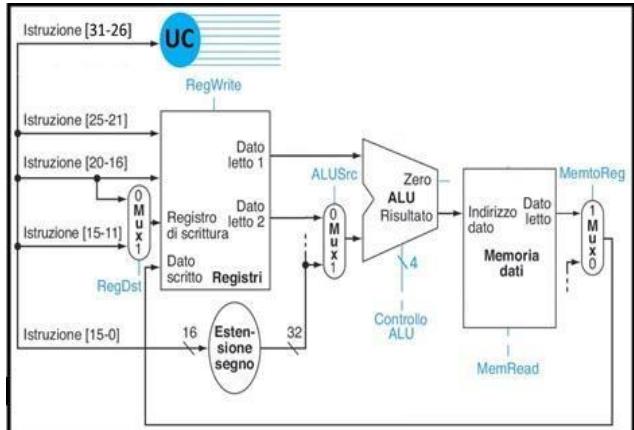
Nella parte della CPU a ciclo singolo relativa alle istruzioni Aritmetico-Logiche MIPS di Tipo R riportata in figura, i 4 bit del segnale **Controllo ALU** forniscono:

- A. L'indirizzo del Registro del processore in cui scrivere il risultato calcolato dall'ALU
- B. L'indirizzo del Registro del processore che contiene il secondo operando dell'ALU
- C. I valori dei segnali di controllo **Ainvert**, **Bnegate**, **OperazioneS1S0** che stabiliscono l'operazione che l'ALU deve eseguire
- D. I segnali di controllo del multiplexer che seleziona i due operandi dell'ALU



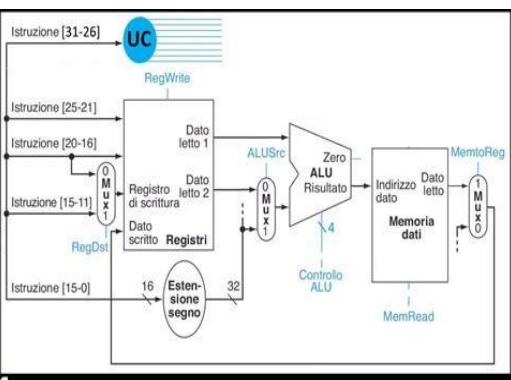
Lo schema in figura rappresenta la computazione che si svolge nella parte del circuito della CPU MIPS a ciclo singolo relativa a:

- Esecuzione dell'istruzione di salto condizionato su uguaglianza
- Esecuzione dell'istruzione Load word
- Esecuzione delle istruzioni Aritmetico-Logiche di Tipo R
- Esecuzione dell'istruzione Store word



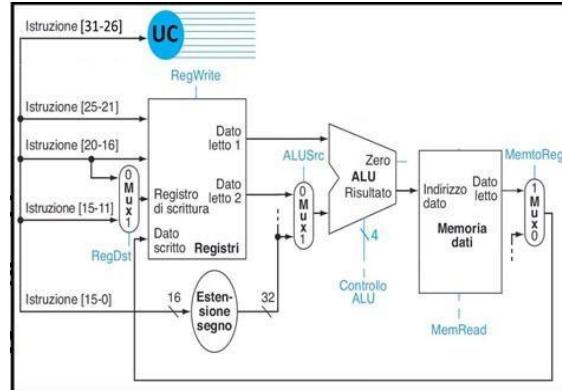
Nella parte della CPU a ciclo singolo relativa all'istruzione LOAD word riportata in figura, l'indirizzo di accesso in Memoria proviene:

- Direttamente dal terminale output dell'ALU che fornisce il risultato dell'addizione del contenuto nel Registro Base con indirizzo nel campo Istruzione[25-21] con il valore dell'Offset contenuto nel campo Istruzione[15-0] Esteso di segno a 32 bit
- Direttamente dal terminale output Dato letto 1 del blocco dei Registri, che fornisce il contenuto del registro di indirizzo Istruzione [25-21] letto anticipatamente
- Direttamente dal terminale output Dato letto 2 del blocco dei Registri, che fornisce il contenuto del registro di indirizzo Istruzione [20-16] letto anticipatamente
- Direttamente dal valore contenuto nel campo Istruzione[15-0] Esteso di segno a 32 bit



Nella parte della CPU a ciclo singolo relativa all'istruzione LOAD word riportata in figura, il multiplexer controllato dal segnale AluSrc effettua la selezione:

- In base al valore AluSrc=0, che instrada in output il valore nel campo Istruzione[15-0], come operando dell'ALU
- In base al valore AluSrc=0, che instrada in output il valore Dato letto 2 letto anticipatamente nel blocco dei Registri, come operando dell'ALU
- In base al valore AluSrc=1, che instrada in output il valore nel campo Istruzione[15-0] Esteso di segno a 32 bit, come operando dell'ALU
- In base al valore AluSrc=1, che instrada in output il valore Dato letto1 letto anticipatamente nel blocco dei Registri, come operando dell'ALU

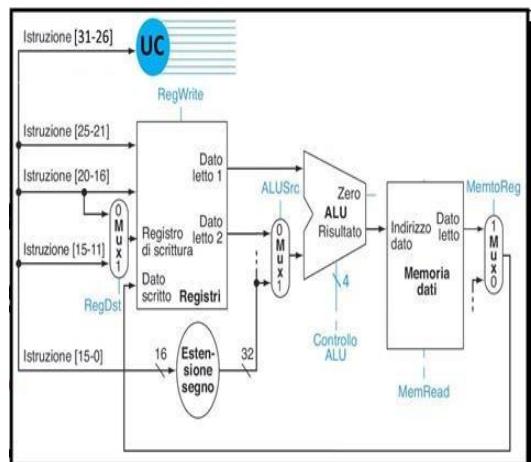


Nella parte della CPU a ciclo singolo relativa all'istruzione LOAD word riportata in figura, il multiplexer controllato dal segnale MemtoReg effettua la selezione:

- In base al valore MemtoReg=1, che instrada in output il valore letto in Memoria attivando la lettura con MemRead=1, come dato da scrivere nel Registro del processore
- In base al valore MemtoReg=0, che instrada in output il risultato calcolato dall'ALU, come dato da scrivere nel Registro del processore

C. In base al valore MemtoReg=0, che instrada in output il campo Istruzione[20-16], come indirizzo del Registro del processore dove scrivere il Dato

D. In base al valore MemtoReg=1, che instrada in output il campo Istruzione[15-11], come indirizzo del Registro del processore dove scrivere il Dato



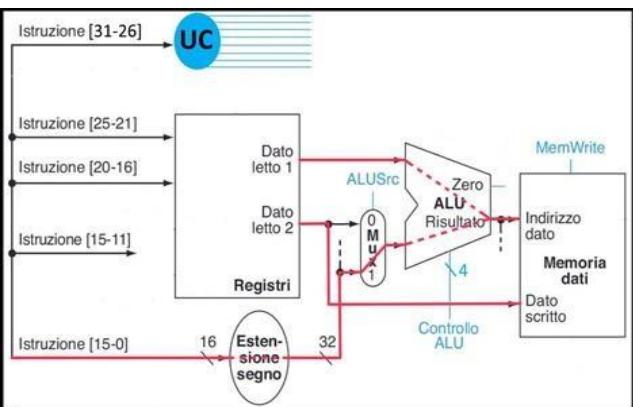
Nella parte della CPU a ciclo singolo relativa all'istruzione LOAD word riportata in figura, il multiplexer controllato dal segnale RegDst effettua la selezione:

A. In base al valore RegDst=0, che instrada in output il risultato calcolato dall'ALU, come dato da scrivere nel Registro del processore

B. In base al valore RegDst=1, che instrada in output il valore letto in Memoria, come dato da scrivere nel Registro del processore

C. In base al valore RegDst=0, che instrada in output il campo Istruzione[20-16], come indirizzo del Registro del processore dove scrivere

D. In base al valore RegDst=1, che instrada in output il campo Istruzione[15-11], come indirizzo del Registro del processore dove scrivere



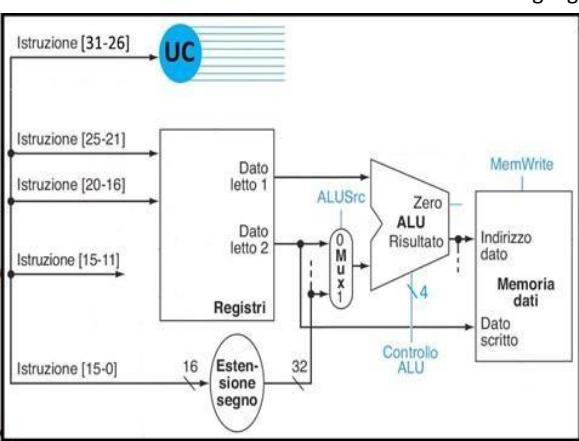
Lo schema in figura rappresenta la computazione che si svolge nella parte del circuito della CPU MIPS a ciclo singolo relativa a:

A. Esecuzione dell'istruzione Load word

B. Esecuzione dell'istruzione Store word

C. Esecuzione delle istruzioni Aritmetico-Logiche di Tipo R

D. Esecuzione dell'istruzione di salto condizionato su uguaglianza



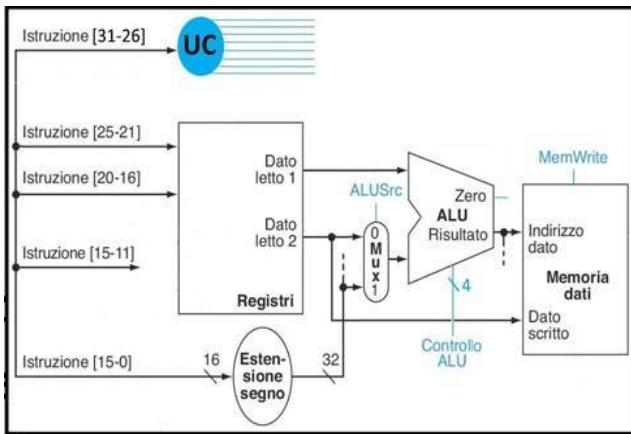
Nella parte della CPU a ciclo singolo relativa all'istruzione STORE word riportata in figura, il multiplexer controllato dal segnale AluSrc effettua la selezione:

A. In base al valore AluSrc=1, che instrada in output il valore nel campo Istruzione[15-0] Esteso di segno a 32 bit, come operando dell'ALU

B. In base al valore AluSrc=0, che instrada in output il valore Dato letto 2 letto anticipatamente nel blocco dei Registri, come operando dell'ALU

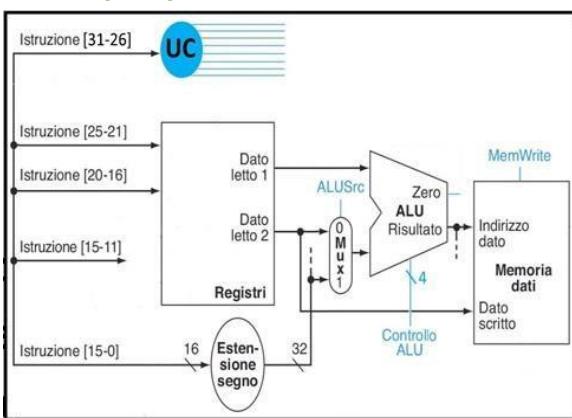
C. In base al valore AluSrc=1, che instrada in output il valore Dato letto 1 letto anticipatamente nel blocco dei Registri, come operando dell'ALU

D. In base al valore AluSrc=0, che instrada in output il valore nel campo Istruzione[15-0], come operando dell'ALU



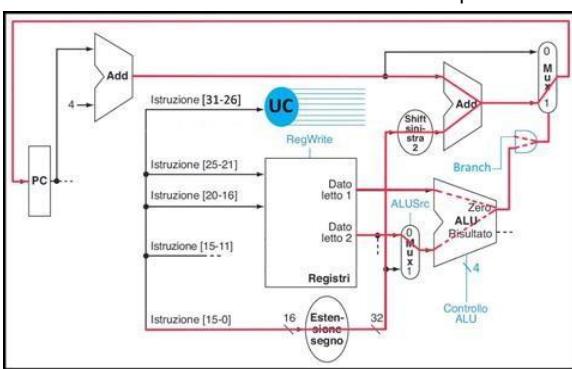
Nella parte della CPU a ciclo singolo relativa all'istruzione STORE word riportata in figura, l'indirizzo di accesso in Memoria proviene:

- Direttamente dal valore contenuto nel campo Istruzione[15-0] Esteso di segno a 32 bit
- Direttamente dal terminale output Dato letto 1 del blocco dei Registri, che fornisce il contenuto del registro di indirizzo Istruzione [25-21] letto anticipatamente
- Direttamente dal terminale output Dato letto 2 del blocco dei Registri, che fornisce il contenuto del registro di indirizzo Istruzione [20-16] letto anticipatamente
- Direttamente dal terminale output dell'ALU che fornisce il risultato dell'addizione del contenuto nel Registro Base con indirizzo nel campo Istruzione[25-21] con il valore dell'Offset contenuto nel campo Istruzione[15-0] Esteso di segno a 32 bit**



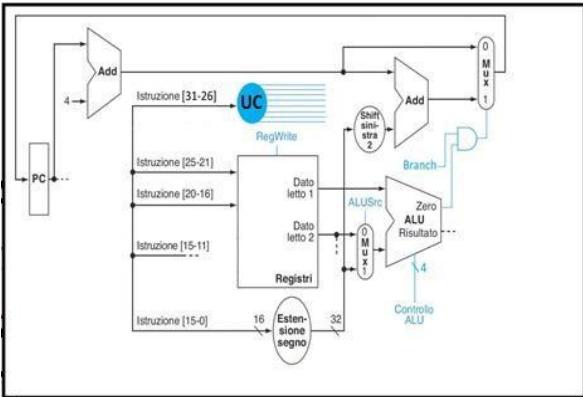
Nella CPU a ciclo singolo relativa all'istruzione STORE word in figura, il valore del Dato da scrivere in Memoria proviene:

- Direttamente dal terminale output dell'ALU che fornisce il risultato, e la scrittura è attivata con MemWrite=1
- Direttamente dal terminale output Dato letto 2 del blocco dei Registri, che fornisce il contenuto del registro di indirizzo Istruzione [20-16] letto anticipatamente, e la scrittura è attivata con MemWrite=1**
- C. Direttamente dal Program Counter aggiornato, e la scrittura è attivata con MemWrite=1
- D. Direttamente dal valore contenuto nel campo Istruzione[15-0] Esteso di segno a 32 bit, e la scrittura è attivata con MemWrite=1



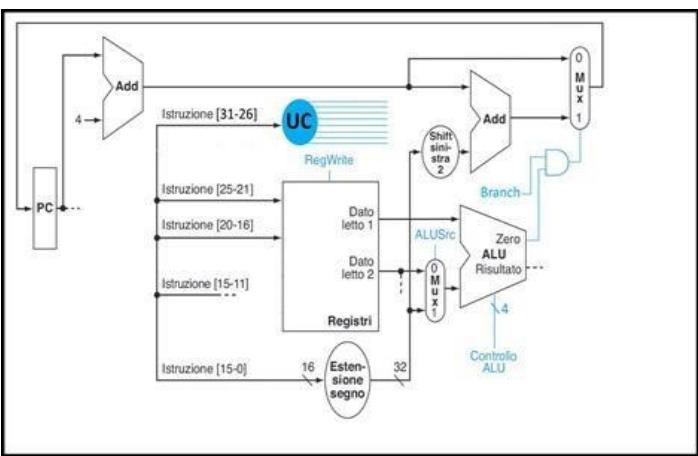
Lo schema in figura rappresenta la computazione che si svolge nella parte del circuito della CPU MIPS a ciclo singolo relativa a:

- Esecuzione dell'istruzione Store word
- Esecuzione dell'istruzione di salto condizionato su uguaglianza BEQ**
- Esecuzione delle istruzioni Aritmetico-Logiche di Tipo R
- Esecuzione dell'istruzione Load word



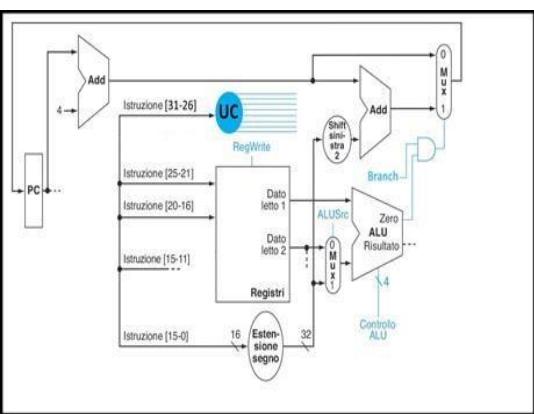
Nella parte della CPU a ciclo singolo relativa all'istruzione BEQ riportata in figura, l'indirizzo di salto condizionato su uguaglianza è calcolato:

- Dall'ALU che riceve come operandi il contenuto dei Registri di indirizzo Istruzione[25-21] e Istruzione[20-16] letti anticipatamente, ed effettua la sottrazione in base al valore del segnale Controllo ALU
- Dal Sommatore a destra che riceve come operandi il contenuto del Program Counter incrementato di 4 e il contenuto nel campo Istruzione[15-0] Esteso di segno a 32 bit e Shiftato a sinistra di 2 posizioni
- Dal Sommatore a sinistra che riceve come operando il contenuto del Program Counter e la costante 4
- Dall'ALU che riceve come operandi il contenuto del Program Counter incrementato di 4 e il contenuto nel campo Istruzione[15-0] Esteso di segno a 32 bit e Shiftato a sinistra di 2 posizioni, ed effettua l'operazione in base al valore del segnale Controllo ALU



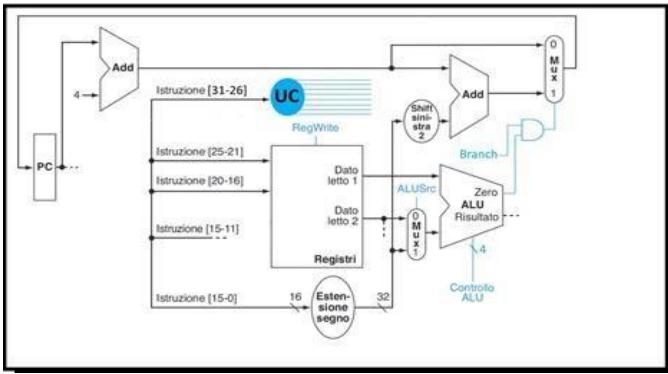
Nella CPU a ciclo singolo relativa all'istruzione BEQ in figura, l'esito del test sulla condizione di uguaglianza per il salto condizionato è calcolato:

- Dall'ALU che fornisce in output il valore del segnale Zero come NOR dei bit del risultato della sottrazione tra Dato letto 1 e Dato letto 2, che sono i contenuti dei Registri di indirizzo Istruzione[25-21] e Istruzione[20-16] letti in anticipo
- Dal Sommatore a destra che addiziona il contenuto del Program Counter incrementato di 4 e il contenuto nel campo Istruzione[15-0] Esteso di segno a 32 bit e Shiftato a sinistra di 2 posizioni
- Dal Sommatore a sinistra che addiziona il contenuto del Program Counter e la costante 4
- Dall'ALU che riceve come operandi il contenuto del Program Counter incrementato di 4 e il contenuto nel campo Istruzione[15-0] Esteso di segno a 32 bit e Shiftato a sinistra di 2 posizioni, ed effettua l'operazione in base al valore del segnale Controllo ALU



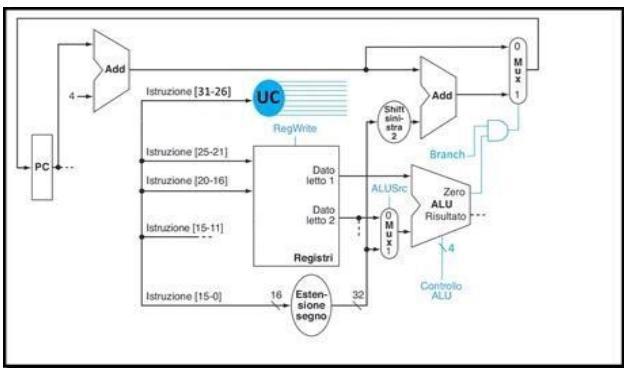
Nella CPU a ciclo singolo relativa all'istruzione BEQ riportata in figura, il multiplexer controllato dal segnale AluSrc effettua la selezione:

- In base al valore AluSrc=1, che instrada come operando dell'ALU il valore nel campo Istruzione[15-0] Esteso di segno
- In base al valore AluSrc=0, che instrada come operando dell'ALU il valore Dato letto 2 contenuto nel Registro di indirizzo Istruzione[20-16] letto in anticipo
- In base al valore AluSrc=1, che instrada come operando dell'ALU il valore Dato letto 1 contenuto nel Registro di indirizzo Istruzione[25-21] letto in anticipo
- In base al valore AluSrc=0, che instrada come operando dell'ALU il valore contenuto nel campo Istruzione[15-0]



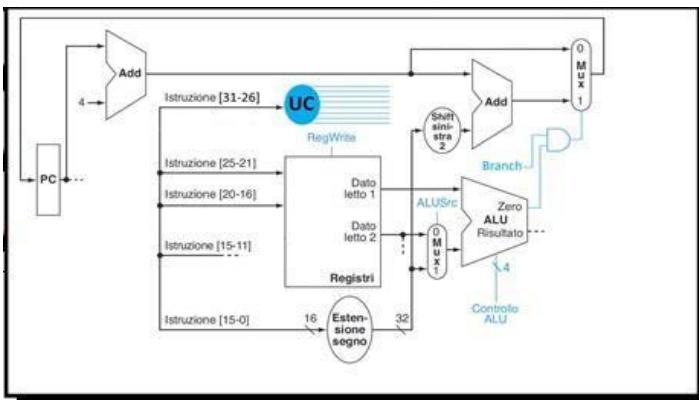
Nella parte della CPU a ciclo singolo relativa all'istruzione BEQ riportata in figura, il multiplexer che sceglie l'indirizzo della prossima istruzione da scrivere nel Program Counter effettua la selezione:

- In base al valore AluSrc=0, che instrada in output il valore Dato letto 2 contenuto nel Registro di indirizzo Istruzione[20-16] letto in anticipo, che fornisce l'indirizzo di salto
- In base al valore RegDst=0, che instrada in output il campo Istruzione[20-16], come indirizzo del Registro del processore che contiene l'indirizzo di salto
- In base al valore MemtoReg=1, che instrada in output il valore letto in Memoria, che fornisce l'indirizzo di salto
- In base al valore output della porta che fornisce l'and tra il segnale di controllo Branch=1 e il segnale Zero calcolato dall'ALU



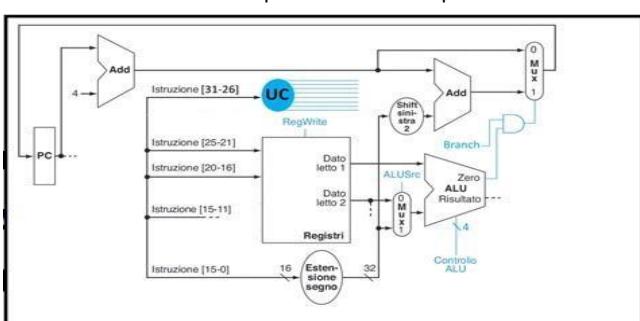
Nella parte della CPU a ciclo singolo relativa all'istruzione BEQ riportata in figura, il dispositivo combinatorio Estensione del segno effettua:

- L'estensione a 32 bit della sequenza input Istruzione[31-26] aggiungendo i bit nelle posizioni più significative con valore uguale al bit di segno
- L'estensione a 32 bit della sequenza input Istruzione[20-16] aggiungendo i bit nelle posizioni più significative con valore uguale al bit di segno
- L'estensione a 32 bit della sequenza input Istruzione[15-0] aggiungendo i bit nelle posizioni più significative con valore uguale al bit di segno
- L'estensione a 32 bit della sequenza input Istruzione[25-21] aggiungendo i bit nelle posizioni più significative con valore uguale al bit di segno



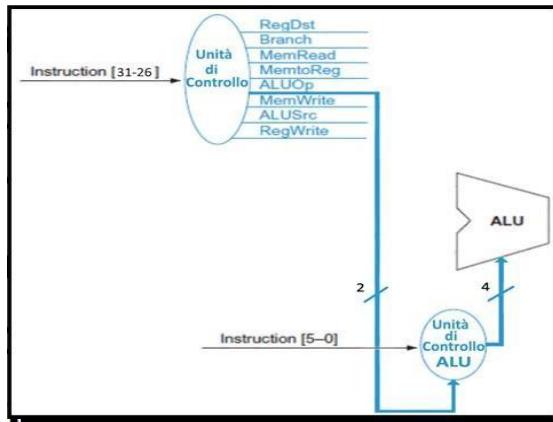
Nella parte della CPU a ciclo singolo relativa all'istruzione BEQ riportata in figura, il dispositivo combinatorio Shift a sinistra effettua:

- Lo shift a sinistra di 2 posizioni della sequenza di 32 bit ricevuta in input dal campo Istruzione[15-0], ponendo a 0 le posizioni lasciate libere
- Lo shift a sinistra di 2 posizioni della sequenza di 32 bit ricevuta in input dal dispositivo Estensione del segno, ponendo a 0 le posizioni lasciate libere
- Lo shift a sinistra di 2 posizioni della sequenza di 32 bit Dato letto 2 letta anticipatamente nel blocco dei Registri, ponendo a 0 le posizioni lasciate libere
- Lo shift a sinistra di 2 posizioni della sequenza di 32 bit del risultato calcolato dall'ALU, ponendo a 0 le posizioni lasciate libere



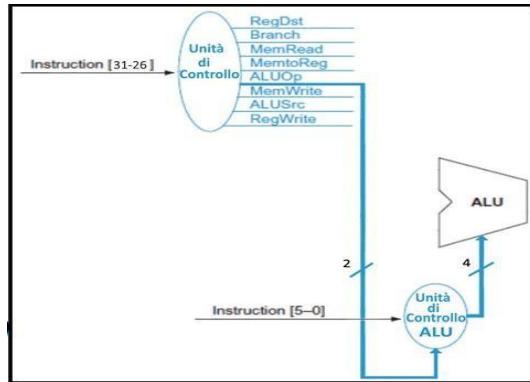
Nella parte della CPU a ciclo singolo relativa all'istruzione BEQ riportata in figura, il valore del segnale Zero influenza:

- A. La scelta effettuata dal multiplexer che seleziona l'indirizzo da scrivere nel Program Counter in base all'output della porta and che riceve in input il segnale di controllo Branch
- B. Il calcolo effettuato dal Sommatore per fornire l'indirizzo di salto condizionato
- C. Il calcolo effettuato dall'ALU per fornire l'indirizzo di salto condizionato
- D. La scelta effettuata dal multiplexer che seleziona l'indirizzo da scrivere nel Program Counter in base al segnale RegDst



1. I 4 segnali di controllo Ainvert, Bnegate, OperationS₁S₀ sono utilizzati per gestire:

- A. La scelta degli operandi dell'ALU
- B. Le operazioni di accesso alla Memoria in lettura o scrittura
- C. Le operazioni di accesso ai Registri del processore in lettura o scrittura
- D. Le operazioni effettuate dall'ALU

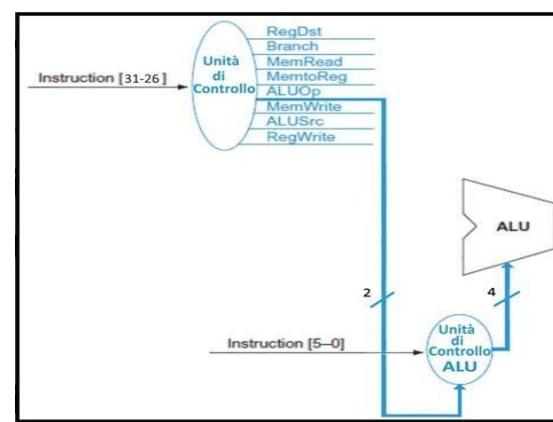


Lo schema riportato in figura con il colore azzurro rappresenta:

- A. La parte della CPU che esegue l'invio all'ALU degli operandi contenuti nei campi Istruzione[31-26] e Istruzione[5-0] per l'esecuzione dell'operazione relativa all'istruzione
- B. La parte della CPU che esegue le operazioni relative alle istruzioni Aritmetico-Logiche specificate nel campo Istruzione[31-26]
- C. I due livelli di decodifica delle informazioni rappresentate dal codice operativo Istruzione[31-26] e dal campo funct Istruzione[5-0], realizzati mediante due dispositivi separati che forniscono i valori dei segnali di controllo
- D. La parte della CPU che esegue il calcolo dell'indirizzo di salto condizionato su uguaglianza utilizzando il contenuto dei campi Istruzione[31-26] e Istruzione[5-0]

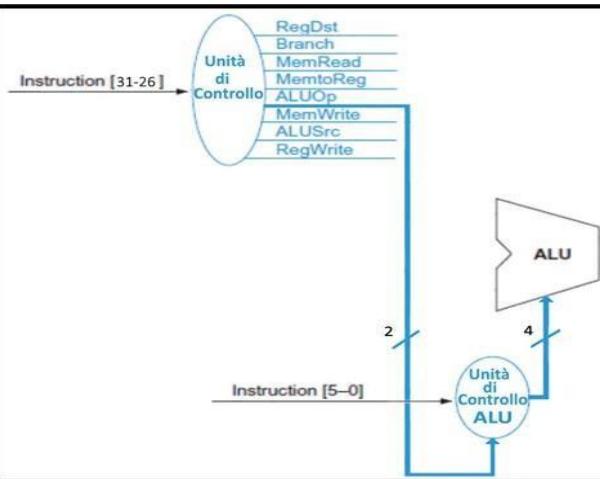
Il segnale di controllo ALUop fornito dall'Unità di Controllo è inviato in input:

- E. Ai Registri del processore per scrivere nel registro destinazione il risultato dell'operazione calcolato dall'ALU
- F. Al Program Counter per scrivere nel campo funct Istruzione[5-0] il valore che specifica l'operazione da eseguire
- G. All'ALU che in base ad esso esegue il calcolo dell'operazione relativa all'istruzione
- H. All'Unità di Controllo dell'ALU che effettua il secondo livello di decodifica e genera i valori dei segnali di controllo Ainvert, Bnegate, OperationS₁S₀



Nello schema riportato in figura, quando il primo livello di decodifica fornisce il segnale di controllo ALUop=01, il secondo livello di decodifica fornisce i valori dei 4 segnali di controllo dell'ALU in base ai quali l'ALU esegue:

- A. Una Addizione per calcolare l'indirizzo di accesso in Memoria nell'esecuzione di lw e sw , indipendentemente dal valore del campo funct
- B. Una Sottrazione per verificare la condizione di uguaglianza tra i Registri nell'esecuzione di beq, indipendentemente dal valore del campo funct
- C. Il calcolo del valore del segnale Zero utilizzato per l'esecuzione dell'istruzione di salto condizionato
- D. Una operazione stabilita in base al valore del campo funct Istruzione[5-0] per l'esecuzione delle istruzioni Aritmetico-Logiche di Tipo R



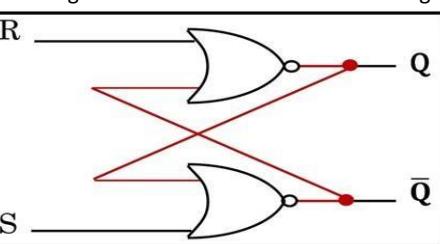
Nello schema riportato in figura, quando il primo livello di decodifica fornisce il segnale di controllo ALUop=10, il secondo livello di decodifica fornisce i valori dei 4 segnali di controllo dell'ALU in base ai quali l'ALU esegue:

- A. Il calcolo del valore del segnale Zero utilizzato per l'esecuzione dell'istruzione di salto condizionato
- B. Una Addizione per calcolare l'indirizzo di accesso in Memoria nell'esecuzione di lw e sw , indipendentemente dal valore del campo funct
- C. Una operazione stabilita in base al valore del campo funct Istruzione[5-0] per l'esecuzione delle istruzioni Aritmetico- Logiche di Tipo R
- D. Una Sottrazione per verificare la condizione di uguaglianza tra i Registri nell'esecuzione di beq, indipendentemente dal valore del campo funct

	INPUT	OUTPUT								
		ALUOp	Branch	MemWrite	MemRead	RegWrite	MemToReg	RegDst	ALUSrc	
Codice Operativo										
Formato R	000000	0	1	0	0	0	1	0	1	0
lw	100011	0	0	0	0	1	1	1	0	1
sw	101011	0	0	0	1	0	0	--	--	1
beq	000100	1	0	1	0	0	0	--	--	0

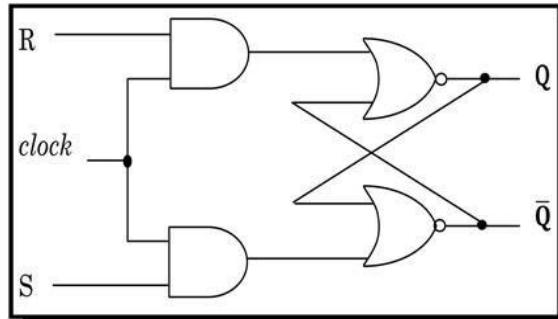
La tabella riportata in figura rappresenta:

- A. Il formato in Lingaggio Macchina MIPS delle istruzioni elencate in rosso
- B. La Tavola di verità della funzione Booleana costituita dalla relazione input-output dell'Unità di Controllo che effettua il PRIMO livello di decodifica nella CPU MIPS a ciclo singolo per l'esecuzione delle istruzioni elencate in rosso
- C. La Tavola di verità della funzione Booleana costituita dalla relazione input-output dell'Unità di Controllo che effettua il SECONDO livello di decodifica nella CPU MIPS a ciclo singolo per l'esecuzione delle istruzioni elencate in rosso
- D. I segnali di controllo memorizzati nei Registri del processore per l'esecuzione delle istruzioni elencate in rosso



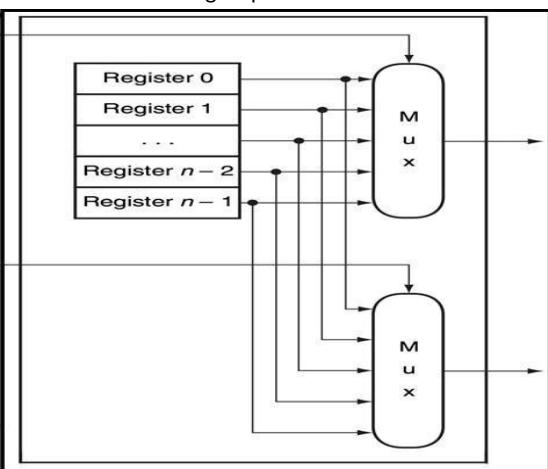
A partire dal LATCH SR riportato in figura, il circuito del Flip/Flop SR si ottiene:

- A. Ponendo R = NOT(S)
- B. Ponendo S = NOT(R)
- C. Introducendo il segnale clock posto in AND con ciascuno degli input S ed R
- D. Eliminando la temporizzazione



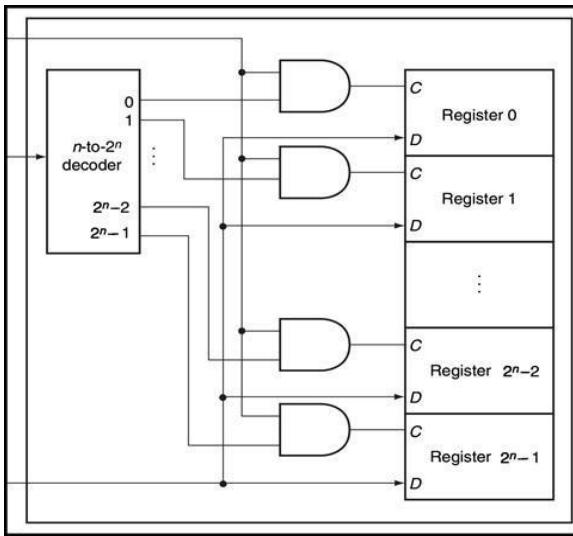
A partire dal Flip-Flop SR riportato in figura, il circuito del Flip/Flop D si ottiene:

- A. Eliminando la temporizzazione
- B. Ponendo $S = \text{NOT}(R)$
- C. Ponendo $R = \text{NOT}(S)$
- D. Ponendo in AND gli input S ed R



Il grafico in figura relativo al processore MIPS rappresenta:

- A. Il circuito di Scrittura dei Registri del processore
- B. Il circuito di Lettura dei Registri del processore
- C. La parte del circuito della CPU coinvolta nell'aggiornamento del Program Counter
- D. La parte del circuito della CPU coinvolta nella fase di Prelievo (Fetch) dell'istruzione



Il grafico in figura relativo al processore MIPS rappresenta:

- A. La parte del circuito della CPU coinvolta nell'aggiornamento del Program Counter
- B. Il circuito di Lettura dei Registri del processore
- C. La parte del circuito della CPU coinvolta nella fase di Prelievo (Fetch) dell'istruzione
- D. Il circuito di Scrittura dei Registri del processore

Considerando un computer con un periodo di clock di 2 ns/ciclo ed un programma che richiede 12000000 di cicli di esecuzione, il tempo di CPU utente è dato da:

1

2

$$\begin{aligned} \text{tempo di CPU utente} &= 12000000 \text{ ciclo} \times 1/(2 \text{ ns/ciclo}) = \\ &= 12 \times 10^6 \text{ ciclo} \times 1/(2 \times 10^{-9} \text{ sec/ciclo}) = \\ &= 6 \times 10^{-3} \text{ sec} = 0,006 \text{ sec} = 6 \text{ ms} \end{aligned}$$

3

$$\begin{aligned} \text{tempo di CPU utente} &= 12000000 \text{ ciclo} \times 2 \text{ ns/ciclo} = \\ &= 12 \times 10^6 \text{ ciclo} \times 2 \times 10^{-9} \text{ sec/ciclo} = \\ &= 24 \times 10^{-3} \text{ sec} = 0,024 \text{ sec} = 24 \text{ ms} \end{aligned}$$

4

$$\begin{aligned} \text{tempo di CPU utente} &= 12000000 \text{ ciclo} \times 1/(2 \text{ ns/ciclo}) = \\ &= 12 \times 10^6 \text{ ciclo} \times 1/(2 \times 10^9 \text{ ciclo/sec}) = \\ &= 6 \times 10^{-3} \text{ sec} = 0,006 \text{ sec} = 6 \text{ ms} \end{aligned}$$

$$\begin{aligned} \text{tempo di CPU utente} &= 12000000 \text{ ciclo} \times 2 \text{ ns/ciclo} = \\ &= 12 \times 10^6 \text{ ciclo} \times 2 \times 10^{-6} \text{ sec/ciclo} = \\ &= 24 \text{ sec} \end{aligned}$$

1. Considerando un computer con una frequenza di clock di 3 GHz ed un programma che richiede 12000000 di cicli di esecuzione, il tempo di CPU utente è dato da:

1

$$\begin{aligned} \text{tempo di CPU utente} &= 12000000 \text{ ciclo} \times \frac{1}{3\text{GHz}} = \\ &= 12 \times 10^6 \text{ ciclo} \times \frac{1}{3} 10^{-9} \text{ sec/ciclo} = \\ &= 4 \text{ sec} \end{aligned}$$

3

2

$$\begin{aligned} \text{tempo di CPU utente} &= 12000000 \text{ ciclo} \times 3 \text{ GHz} = \\ &= 12 \times 10^6 \text{ ciclo} \times 3 \times 10^{-9} \text{ sec/ciclo} = \\ &= 36 \times 10^{-3} \text{ sec} = 0,036 \text{ sec} = 36 \text{ ms} \end{aligned}$$

4

$$\begin{aligned} \text{tempo di CPU utente} &= 12000000 \text{ ciclo} \times 3 \text{ GHz} = \\ &= 12000000 \text{ ciclo} \times 3 \times 10^{-6} \text{ sec/ciclo} = \\ &= 36 \times 10^6 \text{ ciclo} \times 10^{-6} \text{ sec/ciclo} = 36 \text{ sec} \end{aligned}$$

$$\begin{aligned} \text{tempo di CPU utente} &= 12000000 \text{ ciclo} \times \frac{1}{3\text{GHz}} = \\ &= 12 \times 10^6 \text{ ciclo} \times \frac{1}{3} 10^{-9} \text{ sec/ciclo} = \\ &= 4 \times 10^{-3} \text{ sec} = 0,004 \text{ sec} = 4 \text{ ms} \end{aligned}$$

```
if(carattere == 'A' ) {  
    carattere = carattere + 32 ; }  
    carattere = 'B' ;
```

Ricordando che il carattere A ha codice ASCII decimale 65, una traduzione Assembly MIPS delle istruzioni in figura è data:

1

2

3

4

```
assegnando: carattere →$s1  
    subi $t0,$s1,65  
    bne $t0,$zero,END_IF  
    ori $s1,$zero,'a'  
END_IF    ori $s1,$zero,'B'
```

```
assegnando: carattere →$s1  
    subi $t0,$s1,65  
    bne $t0,$zero,END_IF  
    addi $s1,$zero,'a'  
END_IF    addi $s1,$zero,'B'
```

```
assegnando: carattere →$s1  
    subi $t0,$s1,65  
    bne $t0,$zero,END_IF  
    addi $s1,$s1,32  
END_IF    ori $s1,$zero,66
```

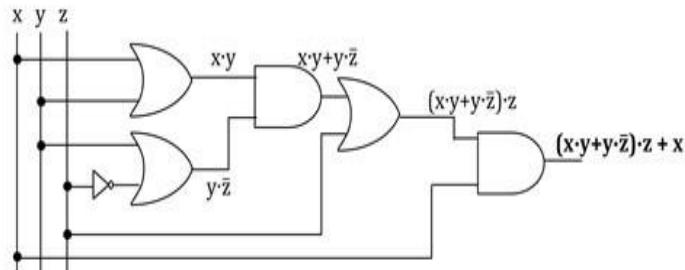
```
assegnando: carattere →$s1  
    bne $s1,'A',END_IF  
    addi $s1,$s1,32  
END_IF    ori $s1,$zero,66
```

$$(x \cdot y + y \cdot \bar{z}) \cdot z + x$$

La Rete Combinatoria equivalente che si ricava dalla Espressione Booleana riportata in figura è:

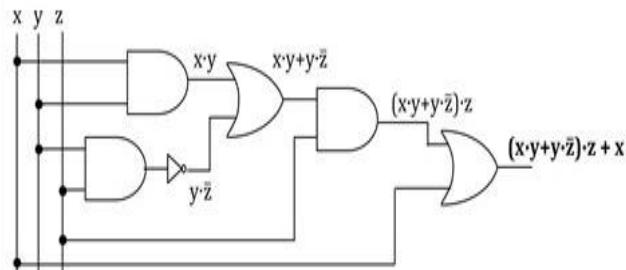
1

Espressione Booleana associata al terminale output della Rete Combinatoria



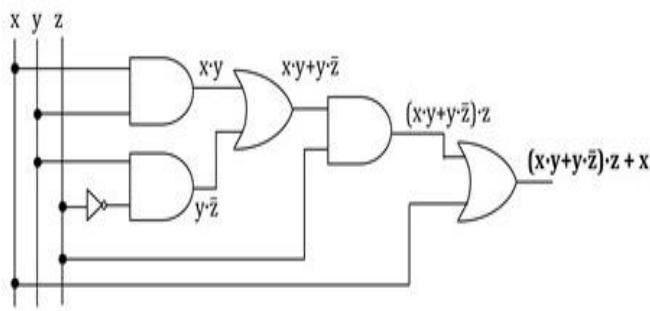
2

Espressione Booleana associata al terminale output della Rete Combinatoria



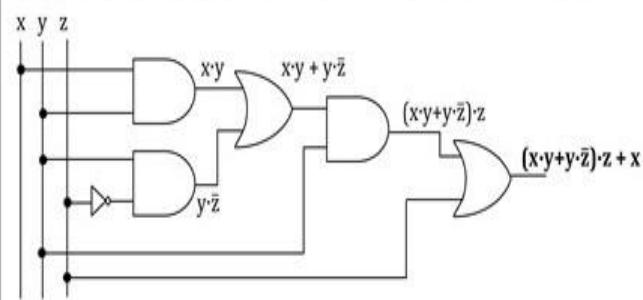
3

Espressione Booleana associata al terminale output della Rete Combinatoria



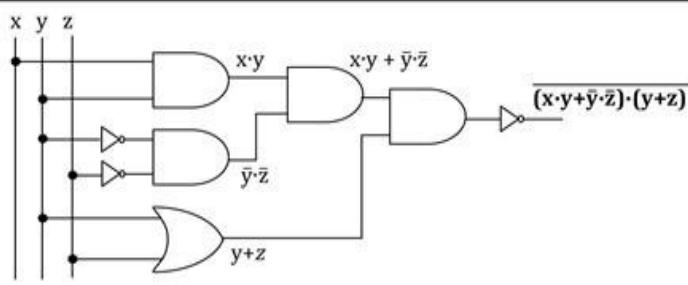
4

Espressione Booleana associata al terminale output della Rete Combinatoria

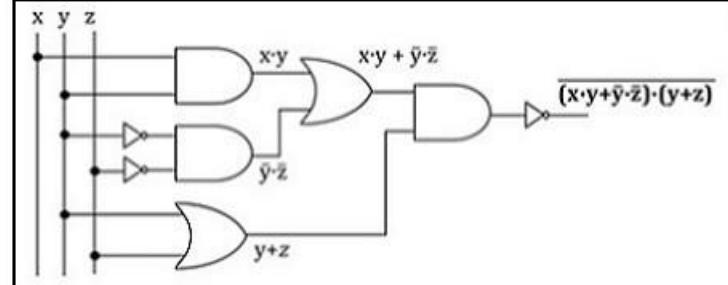


$$\overline{(x \cdot y + \bar{y} \cdot \bar{z}) \cdot (y+z)}$$

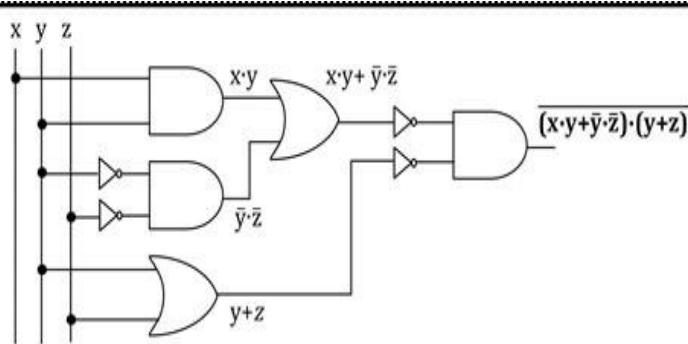
1



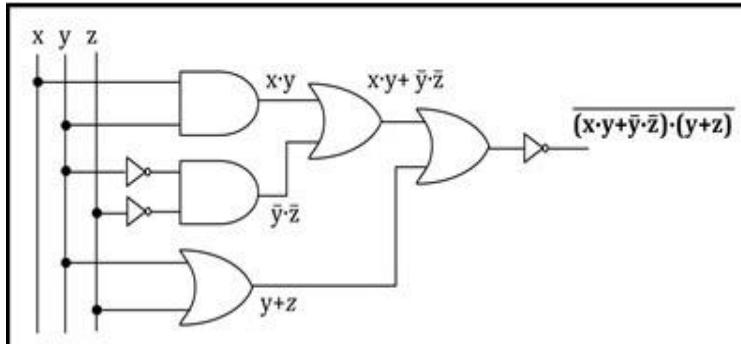
2



3

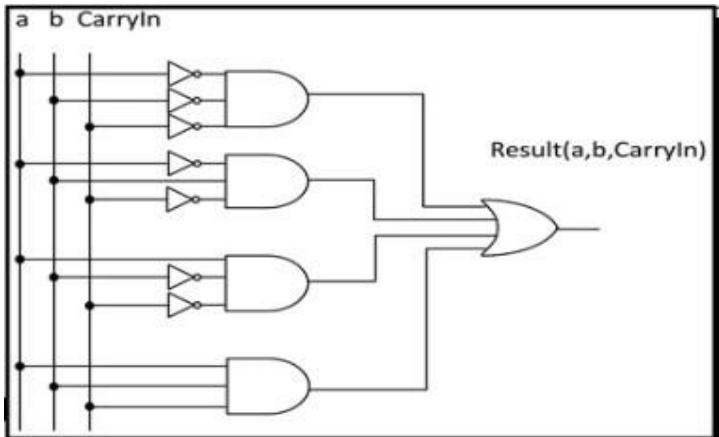


4

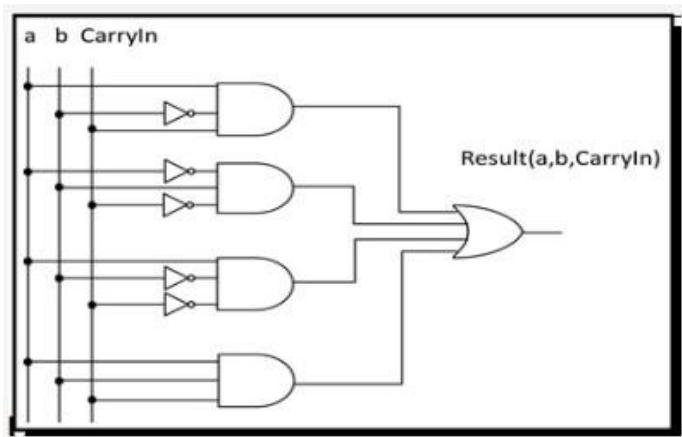


$$\text{Result} = \bar{a} \cdot \bar{b} \cdot \text{CarryIn} + \bar{a} \cdot b \cdot \overline{\text{CarryIn}} + a \cdot \bar{b} \cdot \overline{\text{CarryIn}} + a \cdot b \cdot \text{CarryIn}$$

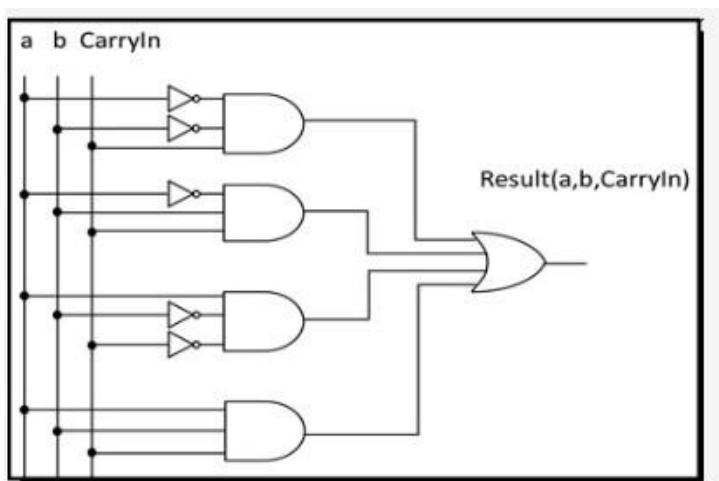
1



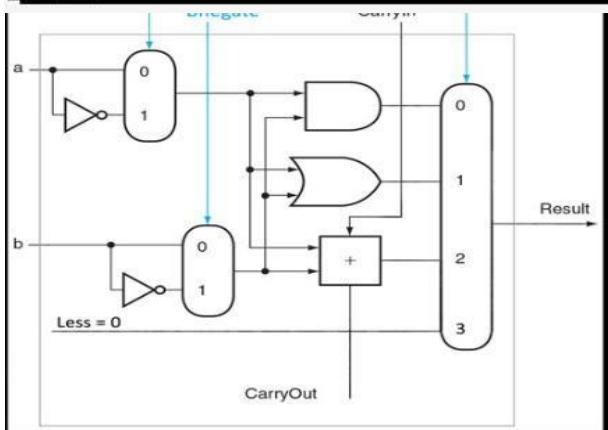
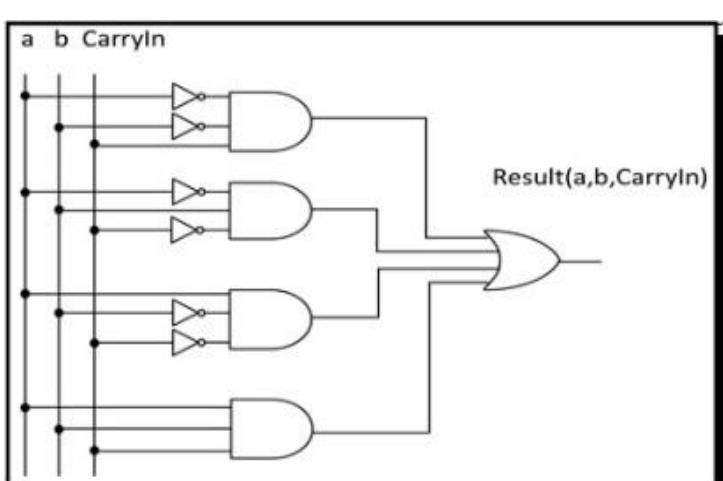
2



3

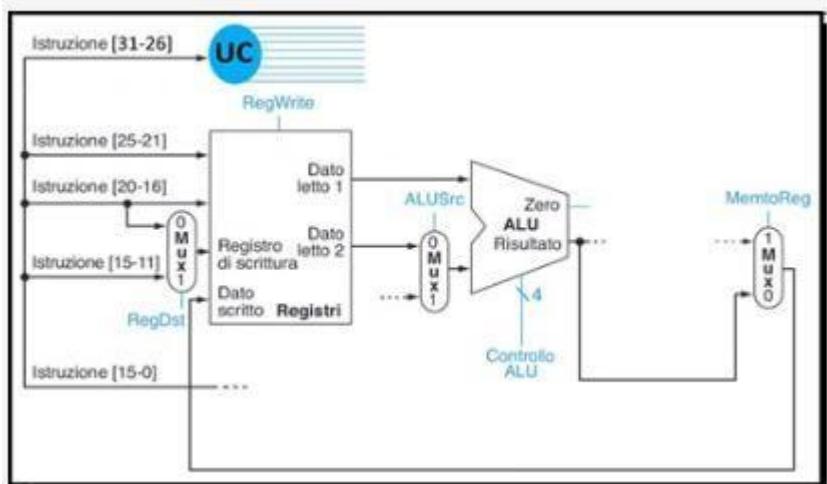


4



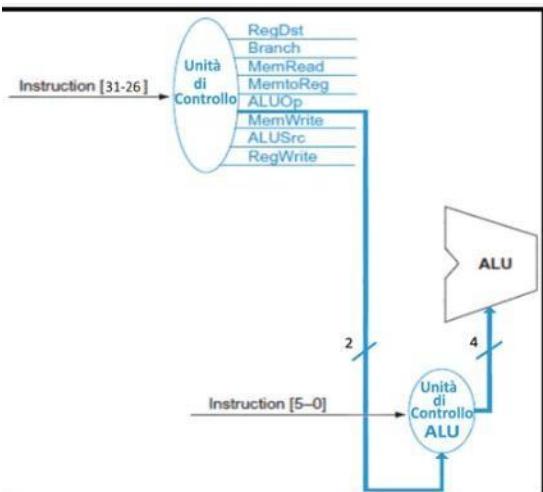
Nel circuito dell'ALU a un bit in figura, il risultato dell'"istruzione NOR si ottiene

- Ponendo $\text{Ainvert}=1$ e $\text{Bnegate}=1$ per complementare bit a bit gli operandi, ed il segnale di controllo Operation per scegliere l'output della porta AND $S_1S_0=00$
- Ponendo $\text{Ainvert}=0$ e $\text{Bnegate}=0$ per complementare bit a bit gli operandi, ed il segnale di controllo Operation per scegliere l'output della porta AND $S_1S_0=00$
- Ponendo $\text{Ainvert}=1$ e $\text{Bnegate}=1$ per complementare bit a bit gli operandi, ed il segnale di controllo Operation per scegliere l'output del Sommatore $S_1S_0=10$
- Ponendo $\text{Ainvert}=0$ e $\text{Bnegate}=0$ per complementare bit a bit gli operandi, ed il segnale di controllo Operation per scegliere l'output del Sommatore $S_1S_0=10$



Nella parte della CPU a ciclo singolo relativa alle istruzioni Aritmetico-Logiche MIPS di Tipo R riportata in figura, il multiplexer controllato dal segnale AluSrc seleziona:

- A. L'indirizzo del Registro del processore in cui scrivere
- B. Il Dato da scrivere nel Registro del processore
- C. Il secondo operando dell'ALU
- D. L'operazione eseguita dall'ALU



Nello schema riportato in figura, quando il primo livello di decodifica fornisce il segnale di controllo ALUop=10, il secondo livello di decodifica fornisce i valori dei 4 segnali di controllo dell'ALU in base ai quali l'ALU esegue:

- A. Il calcolo del valore del segnale Zero utilizzato per l'esecuzione dell'istruzione di salto condizionato
- B. Una Addizione per calcolare l'indirizzo di accesso in Memoria nell'esecuzione di lw e sw , indipendentemente dal valore del campo funct
- C. Una operazione stabilita in base al valore del campo funct Istruzione[5-0] per l'esecuzione delle istruzioni Aritmetico- Logiche di Tipo R
- D. Una Sottrazione per verificare la condizione di uguaglianza tra i Registri nell'esecuzione di beq, indipendentemente dal valore del campo funct

ARRAY di INTERI
temp = A[k] ;
A[k+1] = temp ;

Base A[] k temp
\$s0 \$t1 \$t3
sll \$t4, \$t1, 2 add \$t5, \$t4, \$s0 lw \$t3, 0(\$t5) sw \$t3, 4(\$t5)

Base A[] k temp
\$s0 \$t1 \$t3
sll \$t4, \$t1, 2 add \$t5, \$t4, \$s0 lw \$t3, 0(\$t5) sw \$t3, 0(\$t5)

Base A[] k temp
\$s0 \$t1 \$t3
add \$t5, \$t3, \$s0 lw \$t3, 0(\$t5) sw \$t3, 4(\$t5)

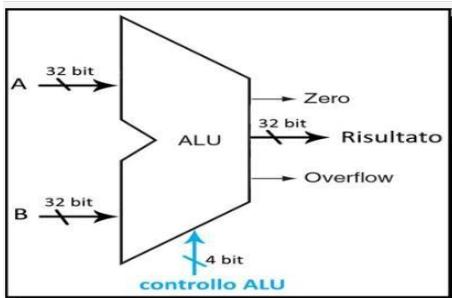
Base A[] k temp
\$s0 \$t1 \$t3
sll \$t4, \$t1, 2 add \$t5, \$t4, \$s0 sw \$t3, 0(\$t5) lw \$t3, 4(\$t5)

1

2

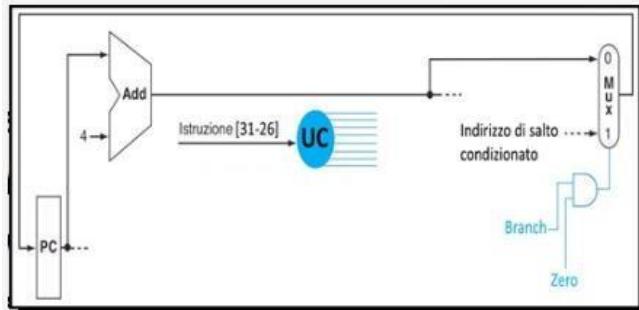
3

4



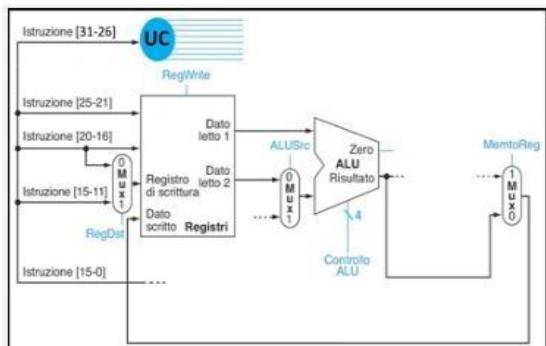
Nel simbolo grafico dell'ALU in figura, i 4 bit del segnale indicato col nome controllo ALU, che fornisce i valori dei segnali Ainvert, Bnegate, OperationS1S0, per l'esecuzione dell'istruzione OR sono posti uguali a:

- A. controllo ALU = 0001
- B. controllo ALU = 1100
- C. controllo ALU = 0110
- D. controllo ALU = 0000



Nello schema in figura il multiplexer seleziona l'indirizzo con cui aggiornare il Program Counter scegliendo tra:

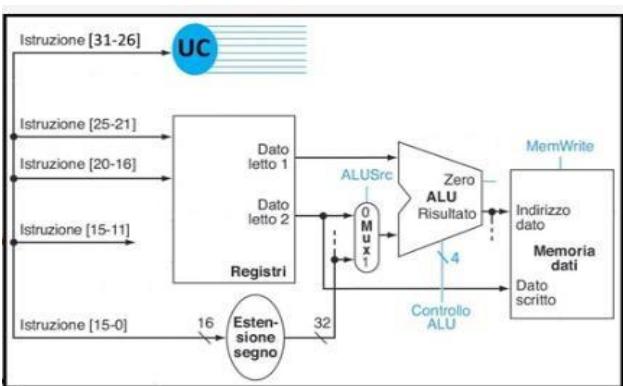
- A. Il risultato calcolato dall'ALU e il dato letto in una parola di memoria nell'esecuzione dell'istruzione lw
- B. L'indirizzo dell'istruzione successiva a quella in esecuzione e l'indirizzo di salto condizionato da usare nell'esecuzione dell'istruzione beq
- C. L'indirizzo dell'istruzione successiva a quella in esecuzione e l'indirizzo di accesso in Memoria da usare nell'esecuzione dell'istruzione sw
- D. L'indirizzo dell'istruzione successiva a quella in esecuzione e l'indirizzo di accesso in Memoria da usare nell'esecuzione dell'istruzione lw



Q504. > <immagine> Nella parte della CPU a ciclo singolo relativa alle istruzioni

Aritmetico-Logiche MIPS di Tipo R riportata in figura, con il valore del segnale di controllo AluSrc=0 il multiplexer seleziona:

- A. L'operazione eseguita dall'ALU
- B. Il contenuto del campo di 5 bit Istruzione[15-11] come indirizzo del Registro del processore in cui scrivere il Dato
- C. Il risultato calcolato dall'ALU come Dato da scrivere nel registro
- D. Il contenuto del Registro del processore letto in anticipo disponibile sul termeminale output Dato letto 2 come secondo operando dell'ALU

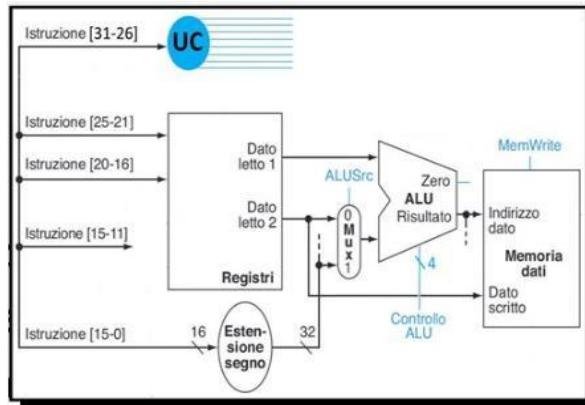


Q505. > <immagine> Nella parte della CPU a ciclo singolo relativa all'istruzione

STORE word riportata in figura, il multiplexer controllato dal segnale AluSrc effettua la selezione:

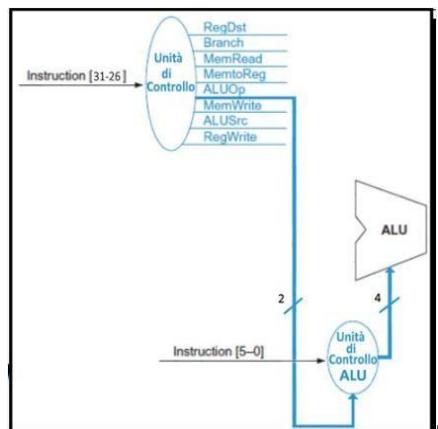
- A. In base al valore AluSrc=1, che instrada in output il valore nel campo Istruzione[15-0] Esteso di segno a 32 bit, come operando dell'ALU

- B. In base al valore AluSrc=0, che instrada in output il valore Dato letto 2 letto anticipatamente nel blocco dei Registri, come operando dell'ALU
 C. In base al valore AluSrc=1, che instrada in output il valore Dato letto1 letto anticipatamente nel blocco dei Registri, come operando dell'ALU
 D. In base al valore AluSrc=0, che instrada in output il valore nel campo Istruzione[15-0], come operando dell'ALU



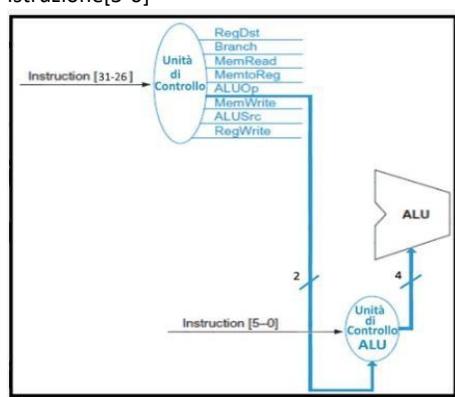
Q506. > <immagine> Nella parte della CPU a ciclo singolo relativa all'istruzione STORE word riportata in figura, il valore del Dato da scrivere in Memoria proviene:

- A. Direttamente dal terminale output dell'ALU che fornisce il risultato, e la scrittura è attivata con MemWrite=1
 B. Direttamente dal terminale output Dato letto 2 del blocco dei Registri, che fornisce il contenuto del registro di indirizzo Istruzione [20-16] letto anticipatamente, e la scrittura è attivata con MemWrite=1
 C. Direttamente dal Program Counter aggiornato, e la scrittura è attivata con MemWrite=1
 D. Direttamente dal valore contenuto nel campo Istruzione[15-0] Esteso di segno a 32 bit, e la scrittura è attivata con MemWrite=1



Lo schema riportato in figura con il colore azzurro rappresenta:

- A. La parte della CPU che esegue l'invio all'ALU degli operandi contenuti nei campi Istruzione[31-26] e Istruzione[5-0] per l'esecuzione dell'operazione relativa all'istruzione
 B. La parte della CPU che esegue le operazioni relative alle istruzioni Aritmetico-Logiche specificate nel campo Istruzione[31-26]
 C. I due livelli di decodifica delle informazioni rappresentate dal codice operativo Istruzione[31-26] e dal campo funct Istruzione[5-0], realizzati mediante due dispositivi separati che forniscono i valori dei segnali di controllo
 D. La parte della CPU che esegue il calcolo dell'indirizzo di salto condizionato su uguaglianza utilizzando il contenuto dei campi Istruzione[31-26] e Istruzione[5-0]

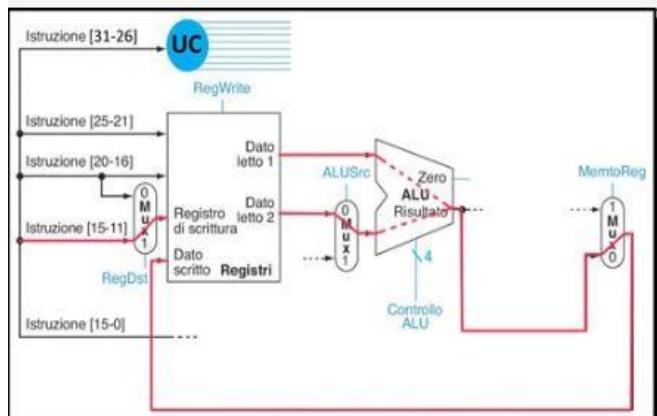
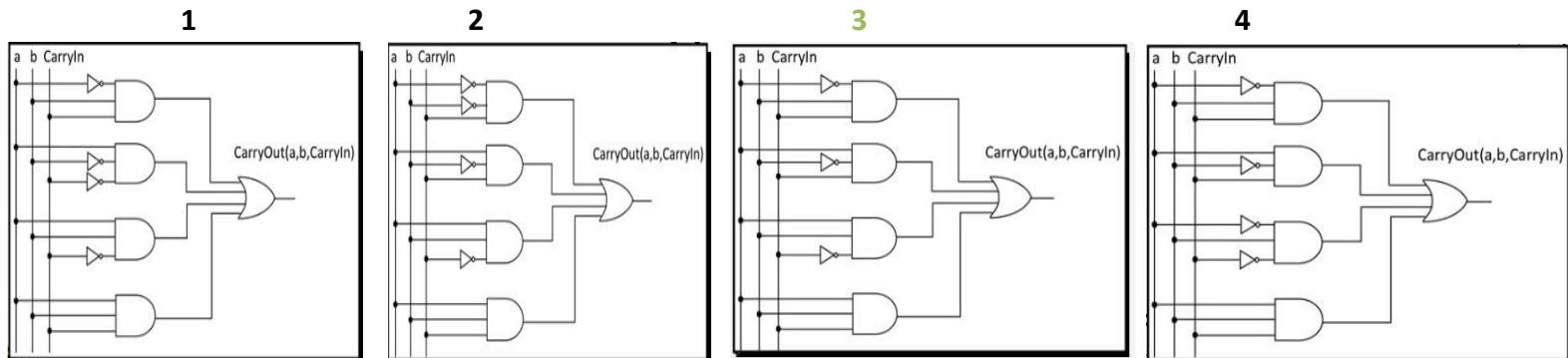


Q509. > <immagine> Nello schema

riportato in figura, quando il primo livello di decodifica fornisce il segnale di controllo ALUop=00, il secondo livello di decodifica fornisce i valori dei 4 segnali di controllo dell'ALU in base ai quali l'ALU esegue:

- A. Una Addizione per calcolare l'indirizzo di accesso in Memoria nell'esecuzione di lw e sw , indipendentemente dal valore del campo funct
 B. Una Sottrazione per verificare la condizione di uguaglianza tra i Registri nell'esecuzione di beq, indipendentemente dal valore del campo funct
 C. Una operazione stabilita in base al valore del campo funct Istruzione[5-0] per l'esecuzione delle istruzioni Aritmetico- Logiche di Tipo R
 D. Il calcolo del valore del segnale Zero utilizzato per l'esecuzione dell'istruzione di salto condizionato

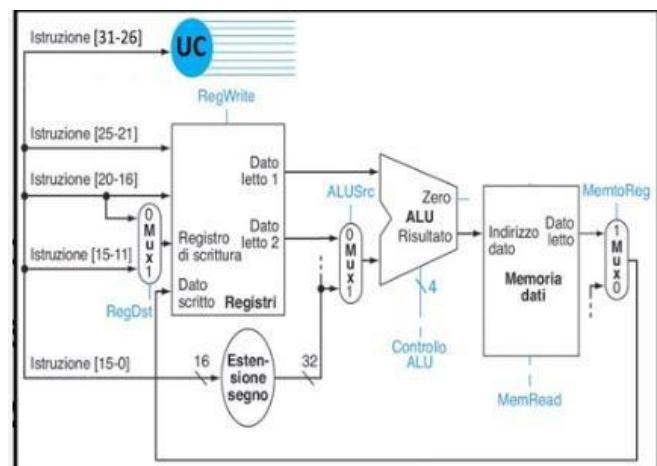
$$\text{CarryOut} = \bar{a} \cdot b \cdot \text{CarryIn} + a \cdot \bar{b} \cdot \text{CarryIn} + a \cdot b \cdot \overline{\text{CarryIn}} + a \cdot b \cdot \text{CarryIn}$$



Q536. > <immagine> Lo schema in figura rappresenta la computazione che si

svolge nella parte del circuito della CPU MIPS a ciclo singolo relativa a:

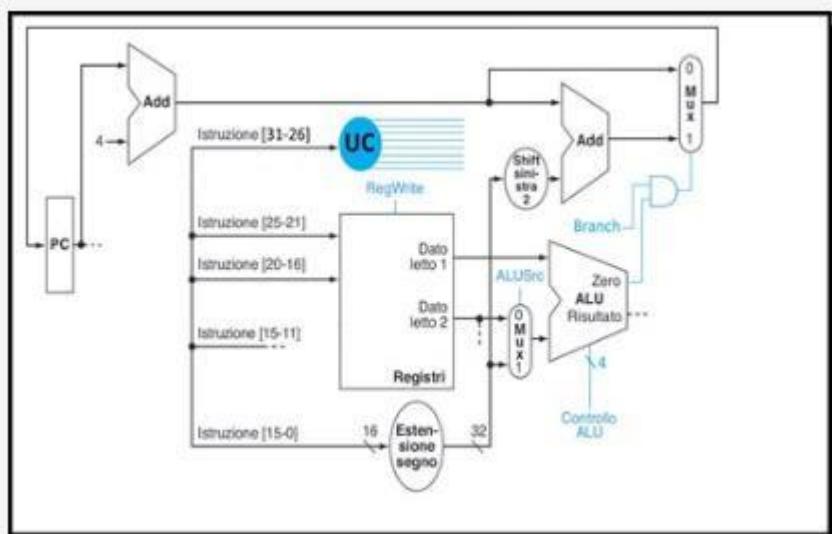
- A. Istruzione di salto condizionato BEQ
- B. Istruzioni di trasferimento dati Load word e Store word
- C. Istruzioni Aritmetico-Logiche di Tipo R
- D. Prelievo dell'istruzione da eseguire e lettura anticipata dei Registri



Q537. > <immagine> Nella parte

della CPU a ciclo singolo relativa all'istruzione LOAD word riportata in figura, il multiplexer controllato dal segnale MemtoReg effettua la selezione:

- A. In base al valore MemtoReg=1, che instrada in output il valore letto in Memoria attivando la lettura con MemRead=1, come dato da scrivere nel Registro del processore
- B. In base al valore MemtoReg=0, che instrada in output il risultato calcolato dall'ALU, come dato da scrivere nel Registro del processore
- C. In base al valore MemtoReg=0, che instrada in output il campo Istruzione[20-16], come indirizzo del Registro del processore dove scrivere il Dato
- D. In base al valore MemtoReg=1, che instrada in output il campo Istruzione[15-11], come indirizzo del Registro del processore dove scrivere il Dato



Q538. > Nella parte della

CPU a ciclo singolo relativa all'istruzione BEQ riportata in figura, il dispositivo combinatorio Estensione del segno effettua:

- A. L'estensione a 32 bit della sequenza input Istruzione[31-26] aggiungendo i bit nelle posizioni più significative con valore uguale al bit di segno
- B. L'estensione a 32 bit della sequenza input Istruzione[20-16] aggiungendo i bit nelle posizioni più significative con valore uguale al bit di segno
- C. L'estensione a 32 bit della sequenza input Istruzione[15-0] aggiungendo i bit nelle posizioni più significative con valore uguale al bit di segno
- D. L'estensione a 32 bit della sequenza input Istruzione[25-21] aggiungendo i bit nelle posizioni più significative con valore uguale al bit di segno

```

if ( val != Max ) {
    istruzione_1
} else {
    istruzione_2
}
istruzione_3

PONENDO:   val → $t0
            Max → $t1

```

Q546. > Per mantenere lo schema dell'istruzione IF-ELSE mostrato in figura la

traduzione in Assembly MIPS della condizione logica su diseguaglianza è:

- A. L'istruzione j ELSE, dove ELSE è l'etichetta di istruzione_2
- B. L'istruzione beq \$t0, \$t1, ELSE, dove ELSE è l'etichetta di istruzione_2
- C. L'istruzione j END_IF dove END_IF è l'etichetta di istruzione_3
- D. L'istruzione beq \$t0, \$t1, END_IF, dove END_IF è l'etichetta di istruzione_3

```

char primo = 'S', secondo = 'B', temp, min ;
if (secondo < primo ) {

    temp = primo ;

    primo = secondo ;

    secondo = temp ;

}

min = primo ;

```

1

assegnando:	primo	secondo	temp	min
	\downarrow	\downarrow	\downarrow	\downarrow
	\$s1	\$s2	\$t0	\$s5

```

        slt $t3, $s2, $s1
        beq $t3, $zero, END_IF
        addi $t0, $s1, 0
        addi $s1, $s2, 0
        addi $s2, $t0, 0
END_IF      add $s5, $zero, $s1

```

assegnando:	primo	secondo	temp	min
	\downarrow	\downarrow	\downarrow	\downarrow
	\$s1	\$s2	\$t0	\$s5

```

        slt $s2, $s1 END_IF
        addi $t0, $s1, 0
        addi $s1, $s2, 0
        addi $s2, $t0, 0
END_IF      add $s5, $zero, $s1

```

assegnando: primo \downarrow \$s1
secondo \downarrow \$s2
temp \downarrow \$t0
min \downarrow \$s5

```

bne $s2, $s1, END_IF
addi $t0, $s1, 0
addi $s1, $s2, 0
addi $s2, $t0, 0
END_IF add $s5, $zero, $s1

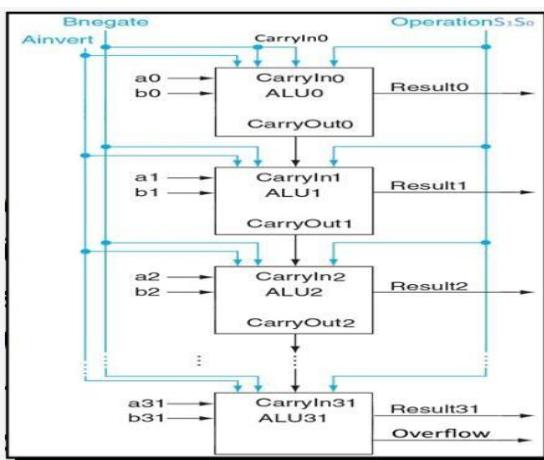
```

assegnando: primo \downarrow \$s1
secondo \downarrow \$s2
temp \downarrow \$t0
min \downarrow \$s5

```

slt $t3, $s2, $s1 END_IF
addi $t0, $s1, 0
addi $s1, $s2, 0
addi $s2, $t0, 0
END_IF add $s5, $zero, $s1

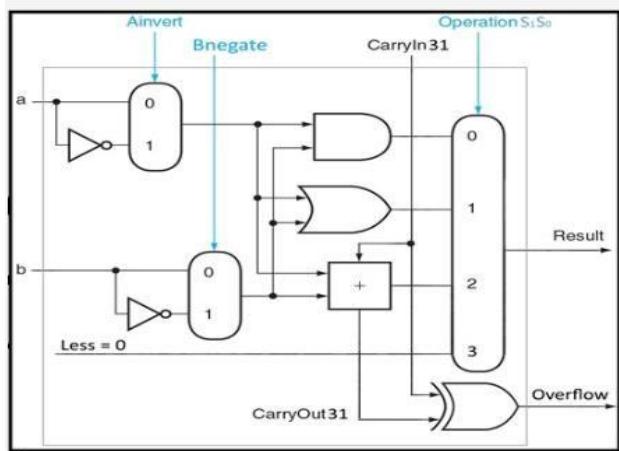
```



Q557. > <immagine> Nel circuito

dell'ALU a 32 bit riportato in figura, il segnale di controllo Bnegate fornisce anche il valore del segnale di controllo CarryIn0 del riporto input dell'ALU a un bit relativa alla posizione meno significativa perché:

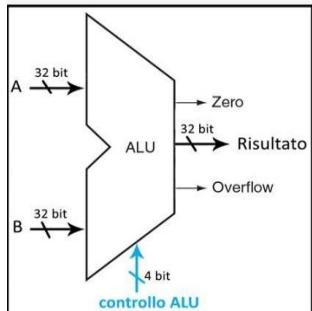
- A. Il valore del segnale CarryIn0 viene cambiato dal segnale Operation in base all'istruzione da eseguire
- B. Il valore del segnale Binvert viene cambiato dal segnale Operation in base all'istruzione da eseguire
- C. Il segnale CarryIn0 influenza solo le operazioni Aritmetiche che coinvolgono il Sommatore e in questi casi ha lo stesso valore del segnale Bnegate; negli altri casi il risultato non dipende dal valore di CarryIn0
- D. I valori assunti dai due segnali sono sempre uguali per tutte le operazione eseguite dall'ALU



Q558. > <immagine> Nell'ALU

ad un bit relativa alla posizione più significativa il figura, il segnale output Overflow generato dalla porta XOR è uguale a 1 quando:

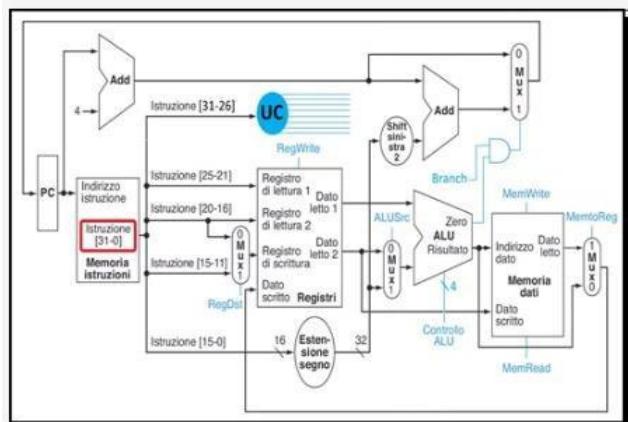
- A. Il risultato calcolato dal Sommatore è uguale a 0
- B. I riporti sono diversi perché il calcolo dell'Addizione ha determinato l'Errore di Overflow
- C. I segnali Ainvert e Bnegate sono entrambi uguali a 1
- D. I riporti sono uguali perché il calcolo dell'Addizione ha determinato l'Errore di Overflow



Q559. > <immagine> Nel simbolo grafico dell'ALU in figura, i 4 bit del segnale indicato col nome **controllo ALU**, che

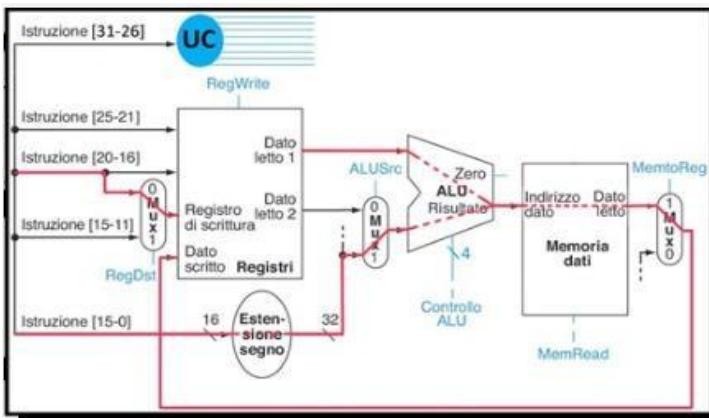
fornisce i valori dei segnali **Ainvert**, **Bnegate**, **OperationS1S0**, per l'esecuzione delle istruzioni **SUB** e **BEQ** sono:

- A. Diversi perché SUB ha Formato di Tipo R e BEQ ha Formato di Tipo I
- B. Diversi perché per SUB si scrive il Risultato nel registro destinazione e per BEQ si utilizza il segnale Zero calcolato dalla porta NOR
- C. Diversi perché SUB è una operazione Aritmetico-Logica e BEQ è una istruzione di salto
- D. Uguali perché l'ALU effettua in entrambi i casi una sottrazione, ma per SUB si utilizza il Risultato e per BEQ si utilizza il segnale Zero calcolato in base al Risultato.



Q560. > <immagine> Lo schema riportato in figura rappresenta:

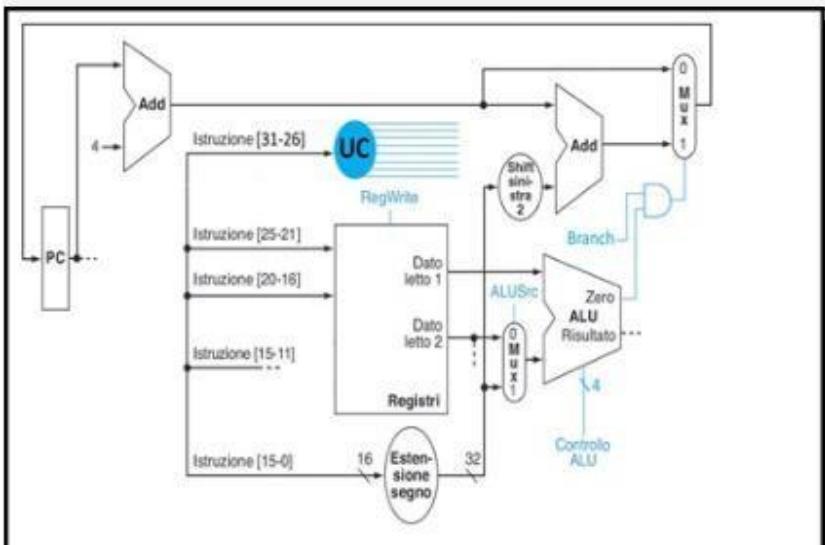
- A. La struttura dei dispositivi di memorizzazione dell'Architettura MIPS
- B. La struttura del circuito dell'Unità di Controllo (UC) a ciclo singolo dell'Architettura MIPS per le istruzioni lw, sw, beq, Aritmetico-Logiche con Formato di Tipo R, e le sue connessioni con la Memoria
- C. La struttura del circuito dell'Unità Aritmetico-Logica (ALU) dell'Architettura MIPS, e le sue connessioni con la Memoria
- D. La struttura del circuito della Unità Centrale di Elaborazione (CPU) a ciclo singolo dell'Architettura MIPS per le istruzioni lw, sw, beq, Aritmetico-Logiche con Formato di Tipo R, e le sue connessioni con la Memoria



Q563. > <immagine> Lo schema in figura rappresenta la computazione

che si svolge nella parte del circuito della CPU MIPS a ciclo singolo relativa a:

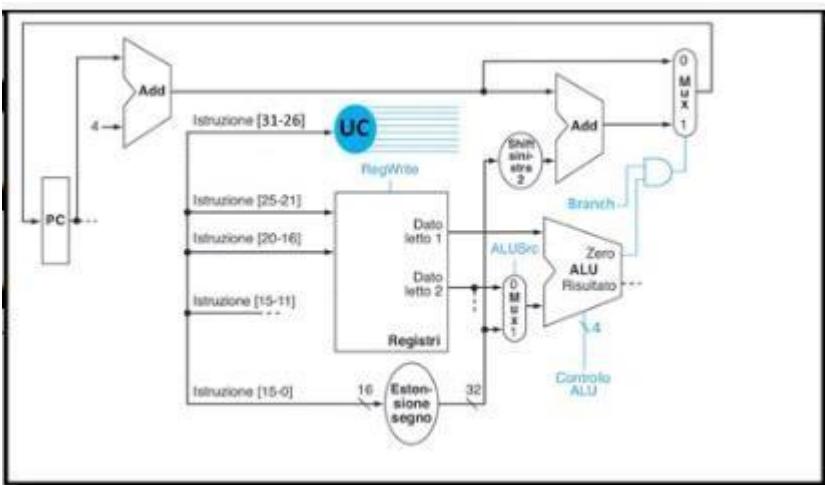
- A. Esecuzione dell'istruzione di salto condizionato su uguaglianza
- B. Esecuzione dell'istruzione Load word
- C. Esecuzione delle istruzioni Aritmetico-Logiche di Tipo R
- D. Esecuzione dell'istruzione Store word



Q564. > Nella parte della

CPU a ciclo singolo relativa all'istruzione BEQ riportata in figura, l'esito del test sulla condizione di uguaglianza per il salto condizionato è calcolato:

- A. Dall'ALU che fornisce in output il valore del segnale Zero come NOR dei bit del risultato della sottrazione tra Dato letto 1 e Dato letto 2, che sono i contenuti dei Registri di indirizzo Istruzione[25-21] e Istruzione[20-16] letti in anticipo
- B. Dal Sommatore a destra che addiziona il contenuto del Program Counter incrementato di 4 e il contenuto nel campo Istruzione[15-0] Esteso di segno a 32 bit e Shiftato a sinistra di 2 posizioni
- C. Dal Sommatore a sinistra che addiziona il contenuto del Program Counter e la costante 4
- D. Dall'ALU che riceve come operandi il contenuto del Program Counter incrementato di 4 e il contenuto nel campo Istruzione[15-0] Esteso di segno a 32 bit e Shiftato a sinistra di 2 posizioni, ed effettua l'operazione in base al valore del segnale Controllo ALU



Q565. > Nella parte della CPU a ciclo singolo

relativa all'istruzione BEQ riportata in figura, il dispositivo combinatorio Shift a sinistra effettua:

- A. Lo shift a sinistra di 2 posizioni della sequenza di 32 bit ricevuta in input dal campo Istruzione[15-0], ponendo a 0 le posizioni lasciate libere
- B. Lo shift a sinistra di 2 posizioni della sequenza di 32 bit ricevuta in input dal dispositivo Estensione del segno, ponendo a 0 le posizioni lasciate libere
- C. Lo shift a sinistra di 2 posizioni della sequenza di 32 bit Dato letto 2 letta anticipatamente nel blocco dei Registri, ponendo a 0 le posizioni lasciate libere
- D. Lo shift a sinistra di 2 posizioni della sequenza di 32 bit del risultato calcolato dall'ALU, ponendo a 0 le posizioni lasciate libere

Q565. In Notazione posizionale pesata il massimo valore rappresentabile con una sequenza binaria di lunghezza M è:

$\text{Il valore } 2^M$ $\text{Il valore } 2^{M-1}$ $\text{Il valore } 2^{M-1}$

A. Non limitato B

C

D

Q573. La traduzione in Assembly MIPS dell'assegnamento `testo[k] = val` con `testo[]` Array di caratteri ASCII è data:

testo []	val	k
\$t1	\$s5	\$t7
add \$t6, \$t7, \$t1		
lbu \$s5, 0 (\$t6)		

testo []	val	k
\$t1	\$s5	\$t7
add \$t6, \$t7, \$t1		
sb \$s5, 0 (\$t6)		

testo []	val	k
\$t1	\$s5	\$t7
add \$t6, \$t7, \$t1		
sb \$s5, \$t7 (\$t1)		

testo []	val	k
\$t1	\$s5	\$t7
add \$t6, \$t7, \$t1		
sb \$s5, \$t1 (\$t7)		

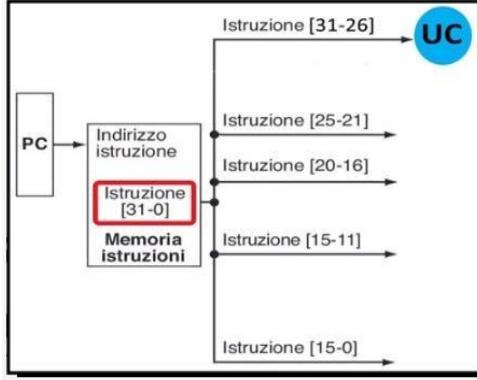
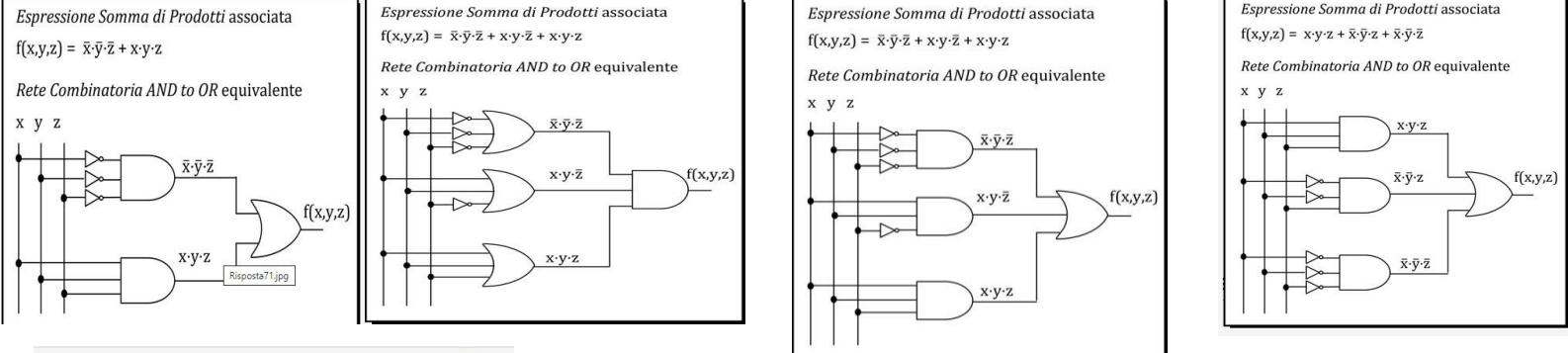
x	y	z	f(x,y,z)
000			1
001			0
010			0
011			0
100			0
101			0
110			1
111			1

1

2

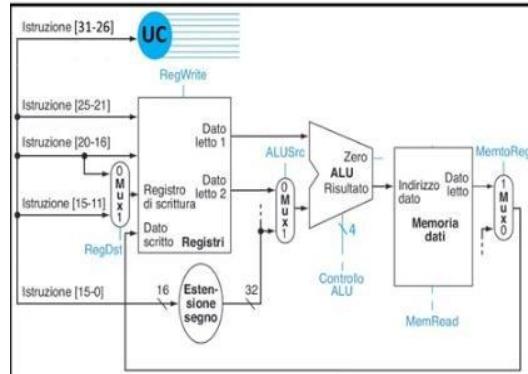
3

4



Nello schema in figura la notazione Istruzione[31-26] rappresenta:

- A. L'indirizzo dell'istruzione successiva che l'Unità di Controllo invia al Program Counter
- B. I 16 bit del campo Codice Operativo dell'istruzione in esecuzione inviati in input alla Unità di Controllo
- C. L'indirizzo del Registro da scrivere che l'Unità di Controllo invia al multiplexer
- D. Il dato da scrivere che l'Unità di Controllo invia al multiplexer



Q585. > Nella parte della CPU a ciclo singolo relativa all'istruzione LOAD word riportata in figura, il multiplexer controllato dal segnale RegDst effettua la selezione:

- A. In base al valore RegDst=0, che instrada in output il risultato calcolato dall'ALU, come dato da scrivere nel Registro del processore
- B. In base al valore RegDst=1, che instrada in output il valore letto in Memoria, come dato da scrivere nel Registro del processore
- C. In base al valore RegDst=0, che instrada in output il campo Istruzione[20-16], come indirizzo del Registro del processore dove scrivere
- D. In base al valore RegDst=1, che instrada in output il campo Istruzione[15-11], come indirizzo del Registro del processore dove scrivere

```

while (val == Max) {
    istruzione_1
}
istruzione_2

PONENDO: val → $t0
          Max → $t1

```

Q595. > Per ottenere lo schema dell'istruzione WHILE mostrato in figura la traduzione in Assembly MIPS della condizione logica su uguaglianza è:

- A. L'istruzione j END WHILE, dove END WHILE è l'etichetta della istruzione_2
- B. L'istruzione bne \$t0, \$t1, END WHILE, dove END WHILE è l'etichetta della istruzione_2
- C. L'istruzione j CICLO dove CICLO è l'etichetta che porta alla ripetizione del ciclo WHILE
- D. L'istruzione bne \$t0, \$t1, CICLO, dove CICLO è l'etichetta che porta alla ripetizione del ciclo WHILE

2^N a 1 è: Q606. La funzione realizzata dal Multiplexer

- A. Se la sequenza binaria degli N segnali di Controllo rappresenta il numero K, allora sul terminale output è instradato il valore del dato input relativo al terminale associato al numero K
- B. Se la sequenza binaria degli N segnali input rappresenta il numero K, allora solo il terminale output associato al numero K ha valore 1, tutti gli altri terminali output hanno valore 0
- C. Se la sequenza binaria degli N segnali dei dati input rappresenta il numero K, allora sul terminale output è instradato il valore del segnale di Controllo relativo al terminale associato al numero K
- D. Se la sequenza binaria degli N segnali input rappresenta il numero K, allora solo il terminale output associato al numero K ha valore 0, tutti gli altri terminali output hanno valore 1

2^N a 1 è: Q607. La funzione realizzata dal Decodificatore >

- A. Se la sequenza binaria degli N segnali input rappresenta il numero K, allora solo il terminale output associato al numero K ha valore 1, tutti gli altri terminali output hanno valore 0
- B. Se la sequenza binaria degli N segnali di Controllo rappresenta il numero K, allora sul terminale output è instradato il valore del dato input relativo al terminale associato al numero K
- C. Se la sequenza binaria degli N segnali dei dati input rappresenta il numero K, allora sul terminale output è instradato il valore del segnale di Controllo relativo al terminale associato al numero K
- D. Se la sequenza binaria degli N segnali input rappresenta il numero K, allora solo il terminale output associato al numero K ha valore 0, tutti gli altri terminali output hanno valore 1

La traduzione in Assembly MIPS dell'assegnamento val = testo[k] con testo[] Array di caratteri ASCII è data:

1	2	3	4
<pre> testo [] val k \$t1 \$s5 \$t7 add \$t6, \$t7, \$t1 lbu \$s5, \$t7 (\$t1) </pre>	<pre> testo [] val k \$t1 \$s5 \$t7 add \$t6, \$t7, \$t1 sw \$s5, 0 (\$t6) </pre>	<pre> testo [] val k \$t1 \$s5 \$t7 add \$t6, \$t7, \$t1 lbu \$s5, 0 (\$t6) </pre>	<pre> testo [] val k \$t1 \$s5 \$t7 add \$t6, \$t7, \$t1 lbu \$s5, \$t1 (\$t7) </pre>

x y z	f(x,y,z)
000	0
001	0
010	1
011	0
100	0
101	0
110	1
111	0

1

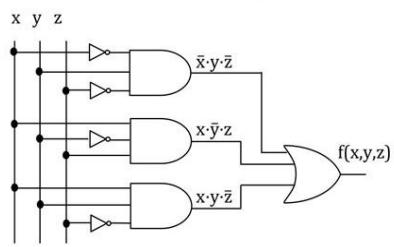
2

3

4

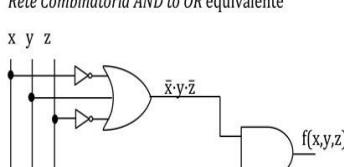
Espressione Somma di Prodotti associata
 $f(x,y,z) = \bar{x} \cdot y \cdot \bar{z} + x \cdot \bar{y} \cdot z + x \cdot y \cdot z$

Rete Combinatoria AND to OR equivalente



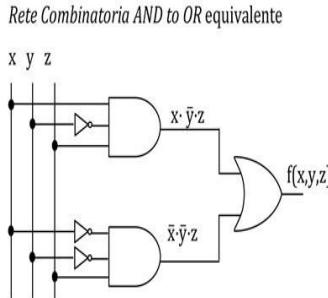
Espressione Somma di Prodotti associata
 $f(x,y,z) = \bar{x} \cdot y \cdot \bar{z} + x \cdot \bar{y} \cdot z$

Rete Combinatoria AND to OR equivalente



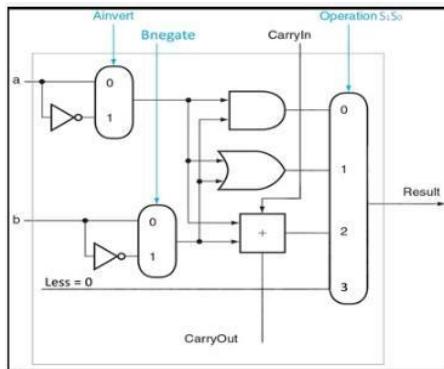
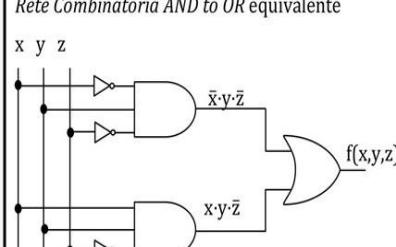
Espressione Somma di Prodotti associata
 $f(x,y,z) = x \cdot \bar{y} \cdot z + \bar{x} \cdot \bar{y} \cdot z$

Rete Combinatoria AND to OR equivalente



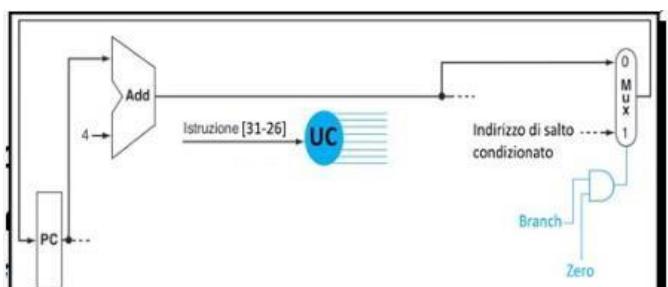
Espressione Somma di Prodotti associata
 $f(x,y,z) = \bar{x} \cdot y \cdot \bar{z} + x \cdot y \cdot \bar{z}$

Rete Combinatoria AND to OR equivalente



Q621. <immagine> Nell'ALU a un bit in figura, per selezionare il risultato dell'istruzione ADD ai due bit del segnale di controllo OperationS1S0 sono assegnati i valori:

- A. > OperationS₁S₀=00
- B. > OperationS₁S₀=01
- C. > OperationS₁S₀=10
- D. > OperationS₁S₀=11



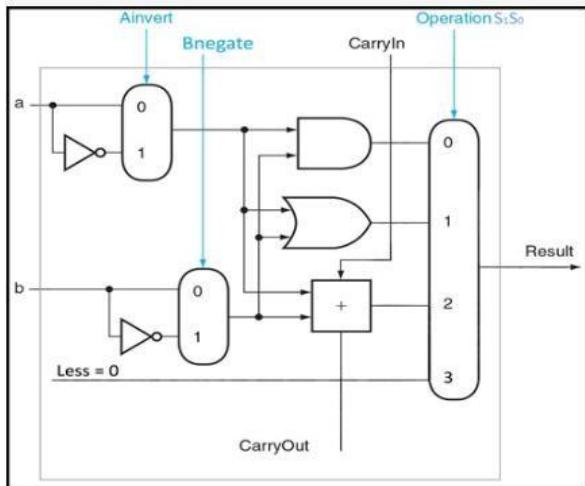
Q626. > <immagine> Lo schema in figura rappresenta la parte del circuito

della CPU MIPS a ciclo singolo coinvolta in:

- A. Esecuzione dell'operazione di Addizione relativa all'istruzione add
- B. Fase di Prelievo dell'istruzione da eseguire
- C. Aggiornamento del Program Counter con l'indirizzo della prossima istruzione da eseguire
- D. Lettura anticipata dei due Registri del processore i cui contenuti possono costituire operandi dell'istruzione in esecuzione

Q637. Nello standard IEEE 754 precisione DOPPIA il massimo modulo rappresentabile è:

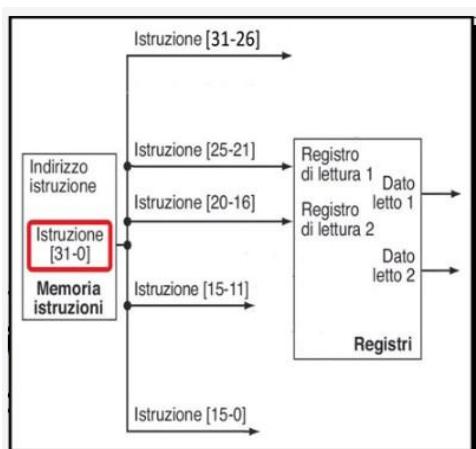
- A. Circa 4 miliardi
- B. Circa 2 miliardi
- C. > Circa 1.7×10^{308}
- D. > Circa 2^{64}



Q646. > <immagine> Nel circuito dell'ALU ad un bit riportato in figura, per

l'esecuzione dell'istruzione SUB l'opposto del secondo operando si ottiene:

- A. Ponendo a 1 il segnale di controllo Bnegate per selezionare il valore dell'operando complementato, e ponendo a 1 il segnale di controllo CarryIn0 dell'ALU ad un bit relativa alla posizione meno significativa per sommare il valore 1 alla sequenza complementata
- B. Ponendo a 1 segnale di controllo Bnegate per selezionare il valore dell'operando complementato, e ponendo a 1 il segnale di controllo Ainvert per sommare il valore 1 alla sequenza complementata
- C. Ponendo a 1 segnale di controllo Bnegate per selezionare il valore dell'operando complementato, e ponendo a 0 il riporto input CarryIn0 dell'ALU ad un bit relativa alla posizione meno significativa
- D. Ponendo a 1 segnale di controllo Bnegate per selezionare il valore dell'operando complementato, e ponendo il segnale di controllo Operation $S_1 S_0 = 01$ sommare il valore 1 alla sequenza complementata

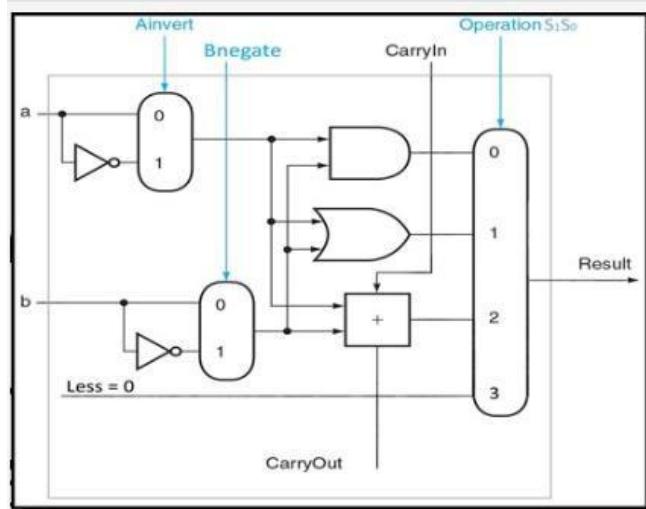


Nello schema in figura le notazioni Istruzioni[25-21] e Istruzioni[20-16] rappresentano:

- A. I due campi di 5 bit che forniscono gli indirizzi degli operandi da leggere anticipatamente nei Registri del processore
- B. I due campi che forniscono l'indirizzo ed il valore del dato da scrivere nel Registro del Processore
- C. L'indirizzo ed il valore del dato letto nel Registro del Processore
- D. I due valori degli operandi di 32 bit letti anticipatamente nei Registri del processore

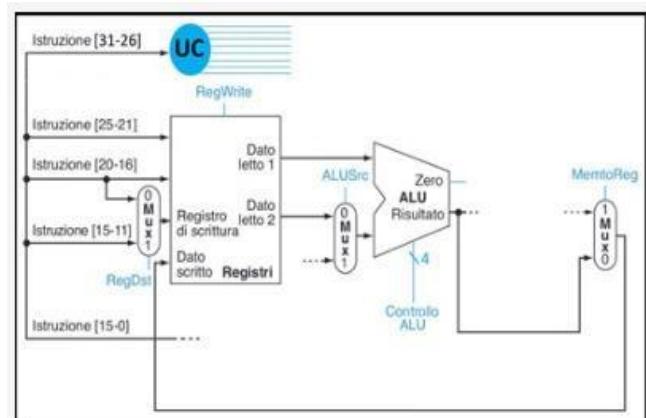
Q653. Il circuito di Lettura dei Registri del processore MIPS utilizza:

- A. Il campo Istruzione [5-0] che fornisce l'indirizzo che consente di selezionare il Registro
- B. > Un decodificatore 5 a 2^5 per selezionare il Registro mediante il relativo indirizzo di 5 bit
- C. Il contenuto del Program Counter che contiene l'indirizzo di accesso in lettura al Registro
- D. > Un multiplexer 2⁵ a 1 per selezionare il Registro mediante il relativo indirizzo di 5 bit



Nel circuito dell'ALU ad un bit riportato in figura, la selezione mediante il multiplexer 4 a 1 del risultato dell'esecuzione dell'istruzione SUB si realizza:

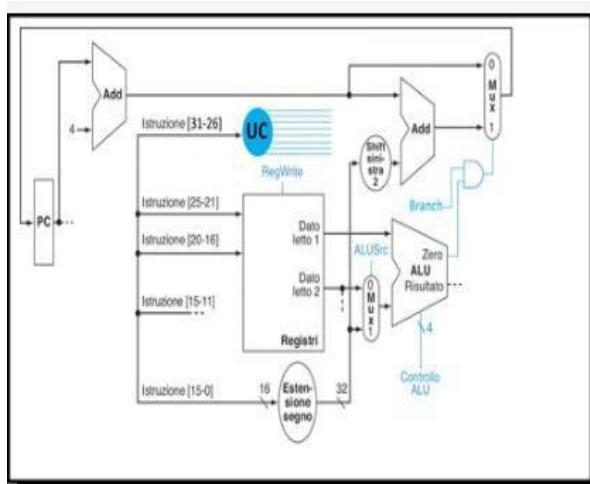
- A. Ponendo il valore del segnale di controllo OperationS1SO=00 per scegliere l'output del Sommatore A
 - B. Ponendo il valore del segnale di controllo OperationS1SO=01 per scegliere l'output del Sommatore B
 - C. Ponendo il valore del segnale di controllo OperationS1SO=10 per scegliere l'output del Sommatore C**
 - D. Ponendo il valore del segnale di controllo OperationS1SO=11 per scegliere l'output del Sommatore D



Nella parte della CPU a ciclo singolo relativa alle istruzioni Aritmetico-Logiche

MIPS di Tipo R riportata in figura, con il valore del segnale di controllo MemtoReg=0 il multiplexer seleziona:

- A. Il risultato calcolato dall'ALU come Dato da scrivere nel registro
 - B. Il contenuto del campo di 5 bit Istruzione[15-11] come indirizzo del Registro del processore in cui scrivere il Dato
 - C. Il contenuto del Registro del processore letto in anticipo disponibile sul termeminale output Dato letto 2 come secondo operando dell'ALU
 - D. L'operazione eseguita dall'ALU



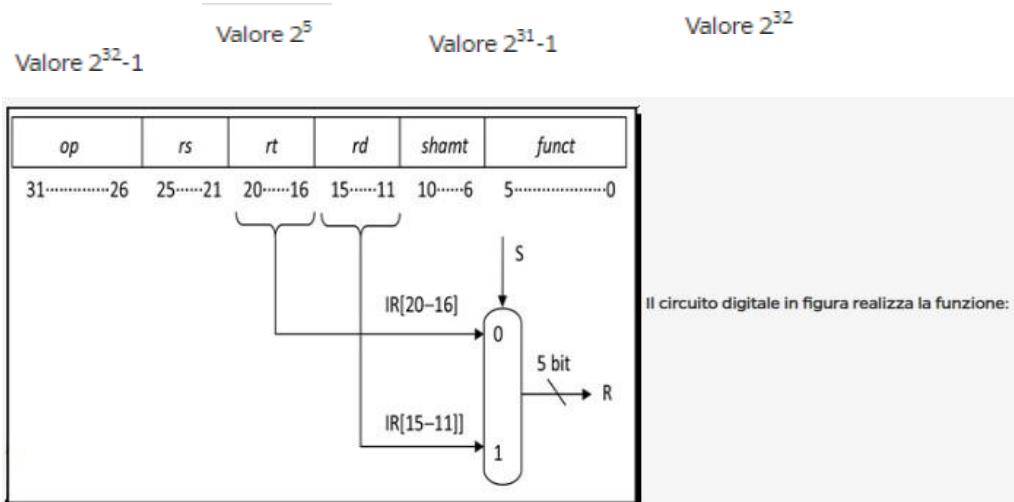
Nella parte della CPU a ciclo singolo relativa all'istruzione BEQ riportata in figura, il

multiplexer controllato dal segnale AluSrc effettua la selezione:

- A. In base al valore AluSrc=1, che instrada come operando dell'ALU il valore nel campo Istruzione[15-0] Esteso di segno
 - B. In base al valore AluSrc=0, che instrada come operando dell'ALU il valore Dato letto 2 contenuto nel Registro di indirizzo Istruzione[20-16] letto in anticipo
 - C. In base al valore AluSrc=1, che instrada come operando dell'ALU il valore Dato letto 1 contenuto nel Registro di indirizzo Istruzione[25-21] letto in anticipo
 - D. In base al valore AluSrc=0, che instrada come operando dell'ALU il valore contenuto nel campo Istruzione[15-0]

Q673. Il numero di locazioni della Memoria principale MIPS è dato dal:

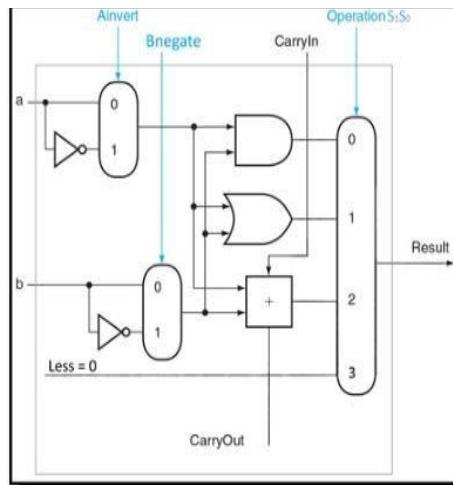
- 1 2 3 4



- A. Se $x=0$ allora $R=IR[20,16]$, se $x=1$ allora $R=IR[15,11]$
- B. Se $y=0$ allora $R=IR[20,16]$, se $y=1$ allora $R=IR[15,11]$
- C. Se $S=0$ allora $R=IR[20,16]$, se $S=1$ allora $R=IR[15,11]$
- D. Se $S=0$ allora $R=IR[15,11]$, se $S=1$ allora $R=IR[20,16]$

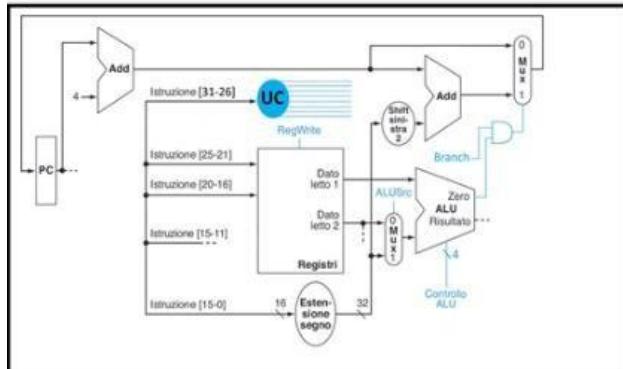
Q685. Per selezionare uno dei 32 registri del processore MIPS si utilizza:

- A > Un Multiplexer $2^5 = 32$ a 1 perché l'indirizzo dei registri è di 5 bit
 - B > Un Multiplexer 2^{32} a 1 perché bisogna scegliere tra 32 registri
 - C > Un Decodificatore 32 a 2^{32} perché bisogna scegliere tra 32 registri
 - D >
- Un Decodificatore 5 a $2^5 = 32$ perché l'indirizzo dei registri è di 5 bit



Nell'ALU a un bit in figura, per selezionare il risultato delle istruzioni AND e OR ai due bit del segnale di controllo $OperationS_1S_0$ sono assegnati i valori:

- A > $OperationS_1S_0=01$ per AND, e $OperationS_1S_0=00$ per OR
 - B > $OperationS_1S_0=10$ per AND, e $OperationS_1S_0=11$ per OR
 - C > $OperationS_1S_0=01$ per AND, e $OperationS_1S_0=10$ per OR
 - D >
- $OperationS_1S_0=00$ per AND, e $OperationS_1S_0=01$ per OR



Q689. > Nella parte della CPU a ciclo singolo relativa all'istruzione BEQ

riportata in figura, il multiplexer che sceglie l'indirizzo della prossima istruzione da scrivere nel Program Counter effettua la selezione:

- A. In base al valore AluSrc=0, che instrada in output il valore Dato letto 2 contenuto nel Registro di indirizzo Istruzione[20-16] letto in anticipo, che fornisce l'indirizzo di salto
- B. In base al valore RegDst=0, che instrada in output il campo Istruzione[20-16], come indirizzo del Registro del processore che contiene l'indirizzo di salto
- C. In base al valore MemtoReg=1, che instrada in output il valore letto in Memoria, che fornisce l'indirizzo di salto
- D. **In base al valore output della porta che fornisce l'and tra il segnale di controllo Branch=1 e il segnale Zero calcolato dall'ALU**

ARRAY di INTERI

`if (0 < A[k])`

Base A[]	k	temp
\$s0	\$t1	\$t3

```
sll $t4, $t1, 2
add $t5, $t4, $s0
sw $s6, 0 ($t5)
slt $t0, $zero, $s6
beq $t0, $zero, END_IF
```

Base A[]	k	temp
\$s0	\$t1	\$t3

```
lw $s6, $t1 ($s0)
slt $t0, $zero, $s6
beq $t0, $zero, END_IF
```

Base A[]	k	temp
\$s0	\$t1	\$t3

```
sll $t4, $t1, 2
add $t5, $t4, $s0
lw $s6, 0 ($t5)
slt $t0, $zero, $s6
beq $t0, $zero, END_IF
```

Base A[]	k	temp
\$s0	\$t1	\$t3

```
add $t5, $t3, $s0
lw $s6, 0 ($t5)
slt $t0, $zero, $s6
beq $t0, $zero, END_IF
```

1

2

3

4

ARRAY di CARATTERI ASCII

`while (testo[k] != 0)`

Base testo[]	k
\$s3	\$t7

```
WHILE add $t2, $t7, $s3
    lbu $t1, 0 ($t2)
    beq $t1, $zero, END WHILE
```

Base testo[]	k
\$s3	\$t7

```
WHILE add $t2, $t7, $s3
    sb $t1, 0 ($t2)
    beq $t1, $zero, END WHILE
```

1

2

Base testo[]	k
\$s3	\$t7

```
WHILE lbu $t1, $t7 ($s3)
    beq $t1, $zero, END WHILE
```

Base testo[]	k
\$s3	\$t7

```
WHILE add $t2, $t7, $s3
    lbu $t2, 0 ($t7)
    beq $t1, $zero, END WHILE
```

ARRAY di CARATTERI ASCII

$x[k] = y[k]$;

1

2

3

4

Base x[] \$s1	Base y[] \$s2	k \$t3
-------------------	-------------------	-----------

```

add $t6, $t3, $s2
lbu $t0, 0($t6)
add $t5, $t3, $t5
sb $t0, 0($t5)
Risp101.jpg
  
```

Base x[] \$s1	Base y[] \$s2	k \$t3
-------------------	-------------------	-----------

```

add $t6, $t3, $s2
lw $t0, 0($t6)
add $t5, $t3, $s1
sw $t0, 0($t5)
  
```

Base x[] \$s1	Base y[] \$s2	k \$t3
-------------------	-------------------	-----------

```

add $t6, $t3, $s2
lhu $t0, 0($t6)
add $t5, $t3, $s1
sh $t0, 0($t5)
  
```

Base x[] \$s1	Base y[] \$s2	k \$t3
-------------------	-------------------	-----------

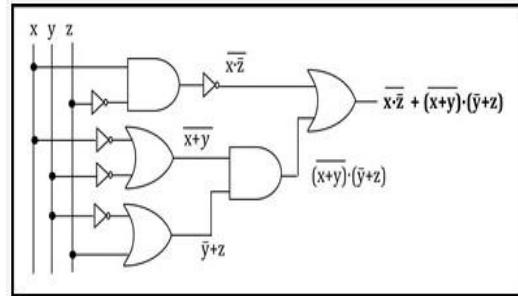
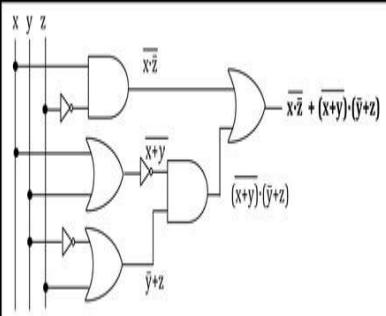
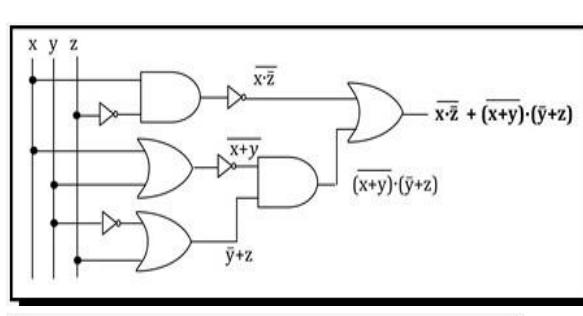
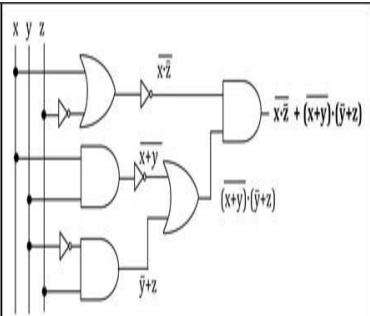
```

lbu $t0, $t3($t6)
sb $t0, $t3($t5)
  
```

$$\overline{x}\overline{z} + (\overline{x+y}) \cdot (\overline{y}+z)$$

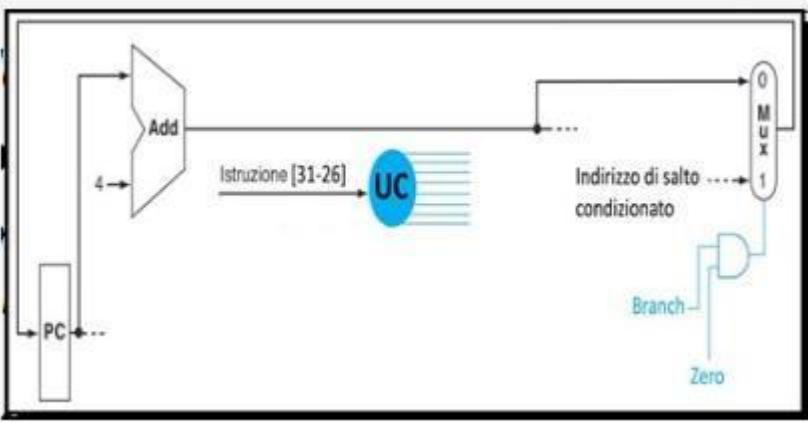
1

2



3

4

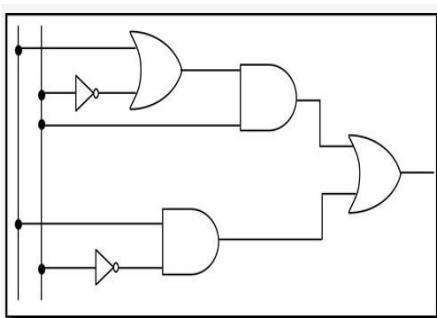


Q722. > Nello schema in figura, quando si ha il

valore del segnale di controllo Branch=1 e il segnale generato dall'ALU Zero=1 il multiplexer seleziona:

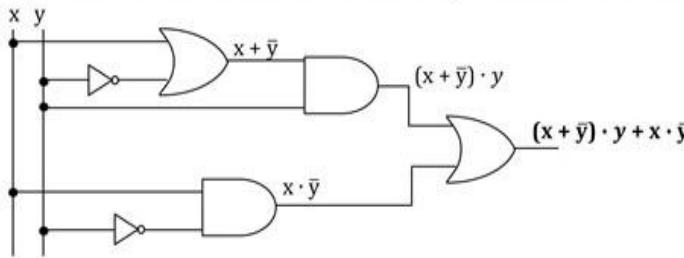
- A. L'indirizzo dell'istruzione attualmente in esecuzione
- B. L'indirizzo dell'istruzione successiva a quella in esecuzione calcolato dal Sommatore ed inviato sull'ingresso dati 0
- C. L'indirizzo calcolato dall'ALU inviato sull'ingresso dati 0

D. L'indirizzo di salto condizionato inviato sull'ingresso dati 1 dalla parte del circuito della CPU che lo calcola



1

Espressione Booleana associata al terminale output della Rete Combinatoria



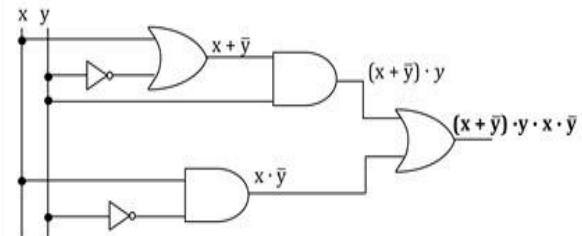
Funzione Booleana $f(x,y)$ associata alla Rete Combinatoria:

xy	$x + \bar{y}$	$(x + \bar{y}) \cdot y$	$x \cdot \bar{y}$	$(x + \bar{y}) \cdot y + x \cdot \bar{y}$
00	1	0	0	0
01	0	0	0	0
10	1	0	1	1
11	1	1	0	1

2

D5 - R3

Espressione Booleana associata al terminale output della Rete Combinatoria



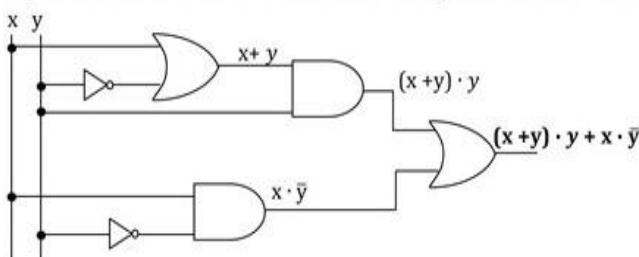
Funzione Booleana $f(x,y)$ associata alla Rete Combinatoria:

xy	$x + \bar{y}$	$(x + \bar{y}) \cdot y$	$x \cdot \bar{y}$	$(x + \bar{y}) \cdot y + x \cdot \bar{y}$
00	1	0	0	0
01	0	0	0	0
10	1	0	1	0
11	1	1	0	0

3

D5 - R2

Espressione Booleana associata al terminale output della Rete Combinatoria



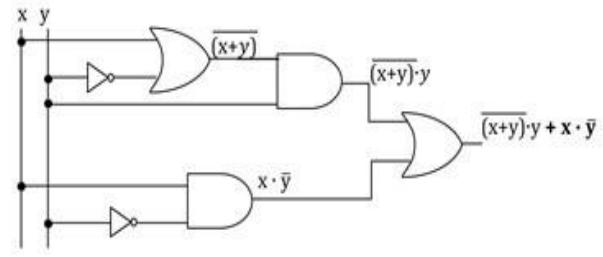
Funzione Booleana $f(x,y)$ associata alla Rete Combinatoria:

xy	$x + y$	$(x + y) \cdot y$	$x \cdot \bar{y}$	$(x + y) \cdot y + x \cdot \bar{y}$
00	0	0	0	0
01	1	1	0	1
10	1	0	1	1
11	1	1	0	1

4

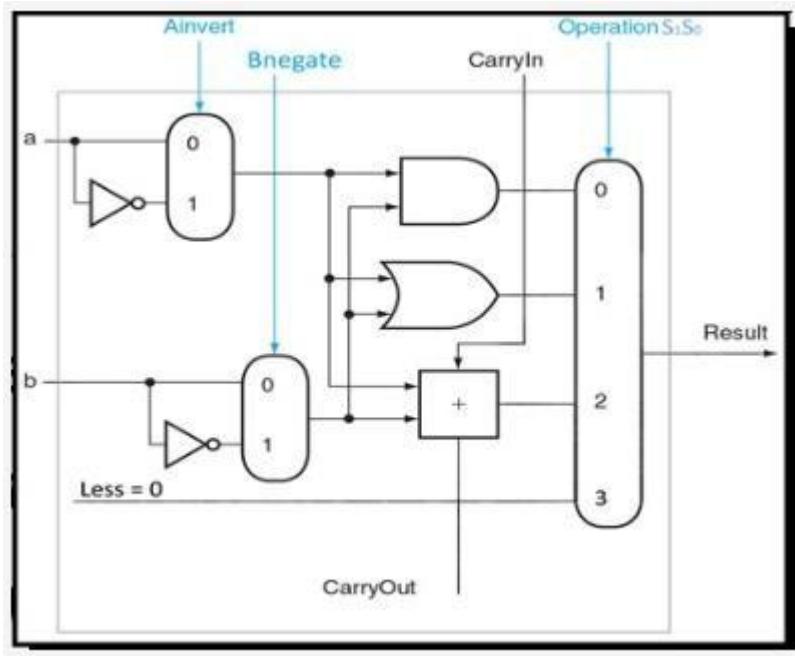
D5 - R4

Espressione Booleana associata al terminale output della Rete Combinatoria



Funzione Booleana $f(x,y)$ associata alla Rete Combinatoria:

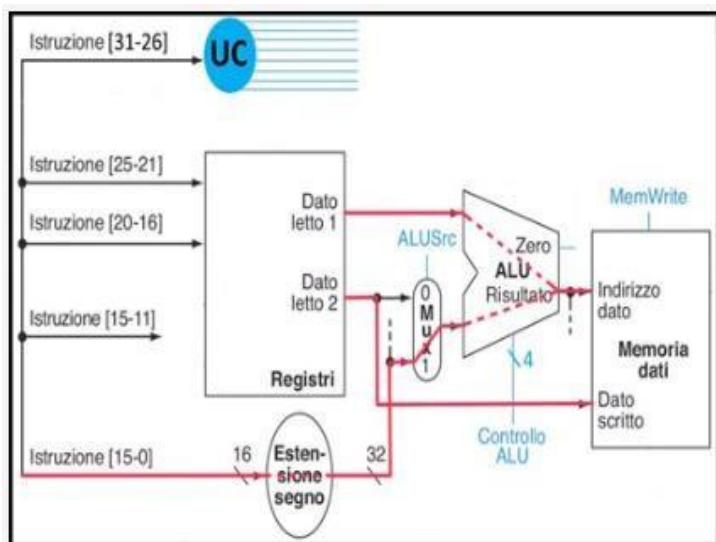
xy	$\bar{x} + \bar{y}$	$(\bar{x} + \bar{y}) \cdot \bar{y}$	$x \cdot \bar{y}$	$(\bar{x} + \bar{y}) \cdot \bar{y} + x \cdot \bar{y}$
00	1	0	0	0
01	1	1	1	1
10	1	0	0	0
11	0	0	0	0



- A. L'ALU ad un bit relativa alla singola posizione della sequenza binaria
 B. Il Sommatore completo che calcola l'Addizione dei bit relativi alla singola posizione della sequenza binaria
 C. L'ALU a 32 bit
 D. Un multiplexer 2 a 1

Q743. La propagazione del riporto nell'ALU si effettua:

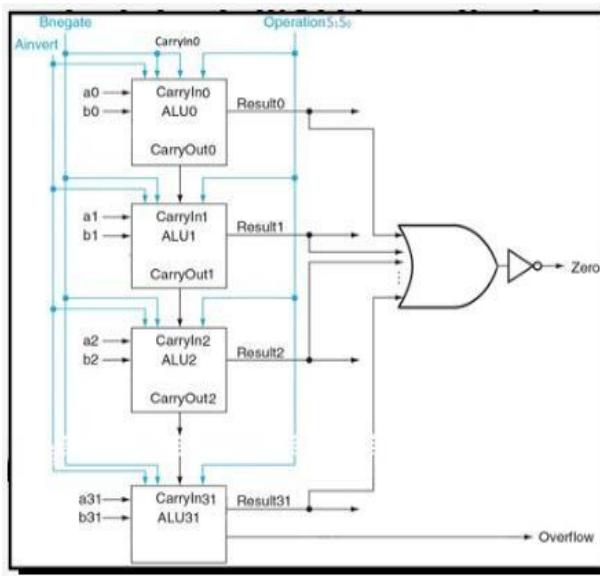
- A. Inviando il segnale di controllo Operation S_1S_0 a tutte le ALU ad un bit relative alle singole posizioni
 B. Inviando il riporto CarryOut calcolato dall'ALU ad un bit relativa ad una posizione come dato input del multiplexer 4 a 1 dell'ALU ad un bit relativa alla posizione successiva
 C. Inviando il riporto CarryOut calcolato dall'ALU ad un bit relativa ad una posizione come segnale di controllo del multiplexer 4 a 1 dell'ALU ad un bit relativa alla posizione successiva
 D. Inviando il riporto CarryOut calcolato dall'ALU ad un bit relativa ad una posizione come riporto input CarryIn dell'ALU ad un bit relativa alla posizione successiva



Q748. > <immagine> Lo schema in figura rappresenta la computazione

che si svolge nella parte del circuito della CPU MIPS a ciclo singolo relativa a:

- A. Esecuzione dell'istruzione Load word
 B. Esecuzione dell'istruzione Store word
 C. Esecuzione delle istruzioni Aritmetico-Logiche di Tipo R
 D. Esecuzione dell'istruzione di salto condizionato su uguaglianza



Nell'ALU a 32 bit in figura quando il segnale output Zero vale 1 si ha che:

- A. Nella esecuzione dell'istruzione ADD si è determinato un Errore di Overflow
- B. Nella esecuzione dell'istruzione BEQ di salto condizionato su uguaglianza il salto non viene effettuato perché gli operandi risultano diversi
- C. Nella esecuzione dell'istruzione BEQ di salto condizionato su uguaglianza il salto viene effettuato perché gli operandi risultano uguali
- D. Nella esecuzione dell'istruzione set on less then la relazione di minore tra gli operandi non è verificata

Q764. In Notazione in complemento a 2, l'overflow viene segnalato quando gli ultimi due riporti

C_N e C_{N-1}:

- A. Sono uguali
- B. Hanno entrambi valore 0
- C. Sono diversi
- D. Hanno entrambi valore 1

Q765. Il segnale di controllo ALUop fornito dall'Unità di Controllo è inviato in input:

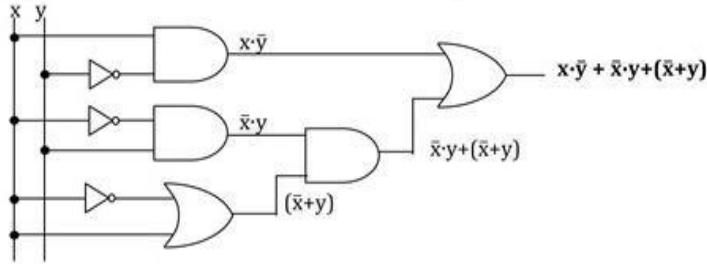
- A. Ai Registri del processore per scrivere nel registro destinazione il risultato dell'operazione calcolato dall'ALU
- B. Al Program Counter per scrivere nel campo funct Istruzione[5-0] il valore che specifica l'operazione da eseguire
- C. All'ALU che in base ad esso esegue il calcolo dell'operazione relativa all'istruzione

D. All'Unità **c₁s₀** controllo dell'ALU che effettua il secondo livello di decodifica e genera i valori dei segnali di controllo dell'ALU Ainvert, Bnegate, Operation>

Q766. Nel Formato di Tipo I, i valori del campo immediato sono compresi tra:

- A > Il minimo -2¹⁶ ed il massimo 2¹⁶
- B > Il minimo -2¹⁵ ed il massimo 2¹⁵
- C > Il minimo -2¹⁵ ed il massimo 2¹⁵-1
- D > Il minimo 0 ed il massimo 2¹⁶-1

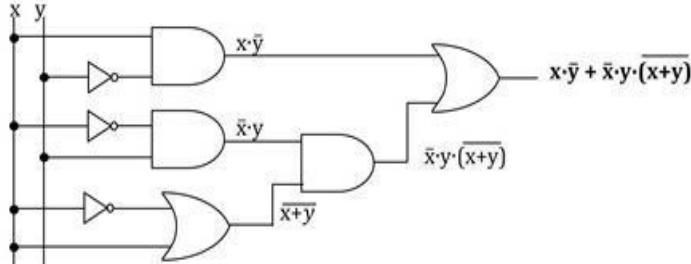
Espressione Booleana associata al terminale *output* della Rete Combinatoria



Funzione Booleana $f(x,y)$ associata alla Rete Combinatoria:

xy	$x \cdot \bar{y}$	$\bar{x} \cdot y$	$(\bar{x}+y)$	$\bar{x} \cdot y + (\bar{x}+y)$	$x \cdot \bar{y} + \bar{x} \cdot y + (\bar{x}+y)$
00	0	0	1	1	1
01	0	1	1	1	1
10	1	0	0	0	1
11	0	0	1	1	1

Espressione Booleana associata al terminale *output* della Rete Combinatoria

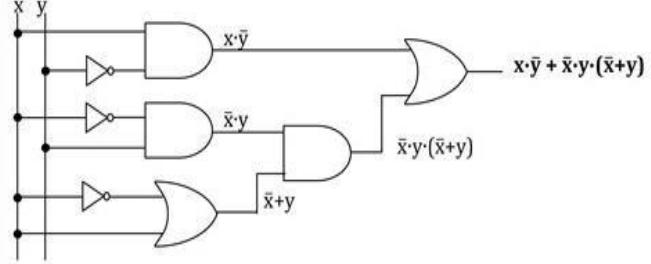


Funzione Booleana $f(x,y)$ associata alla Rete Combinatoria:

xy	$x \cdot \bar{y}$	$\bar{x} \cdot y$	$(\bar{x}+y)$	$\bar{x} \cdot y \cdot (\bar{x}+y)$	$x \cdot \bar{y} + \bar{x} \cdot y \cdot (\bar{x}+y)$
00	0	0	1	0	0
01	0	1	0	0	0
10	1	0	0	0	1
11	0	0	0	0	0

Risposta 6

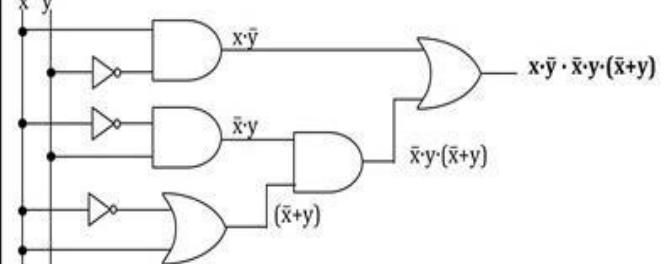
Espressione Booleana associata al terminale *output* della Rete Combinatoria



Funzione Booleana $f(x,y)$ associata alla Rete Combinatoria:

xy	$x \cdot \bar{y}$	$\bar{x} \cdot y$	$(\bar{x}+y)$	$\bar{x} \cdot y \cdot (\bar{x}+y)$	$x \cdot \bar{y} + \bar{x} \cdot y \cdot (\bar{x}+y)$
00	0	0	1	0	0
01	0	1	1	1	1
10	1	0	0	0	1
11	0	0	1	0	0

Espressione Booleana associata al terminale *output* della Rete Combinatoria



Funzione Booleana $f(x,y)$ associata alla Rete Combinatoria:

xy	$x \cdot \bar{y}$	$\bar{x} \cdot y$	$(\bar{x}+y)$	$\bar{x} \cdot y \cdot (\bar{x}+y)$	$x \cdot \bar{y} + \bar{x} \cdot y \cdot (\bar{x}+y)$
00	0	0	1	0	0
01	0	1	1	1	0
10	1	0	0	0	0
11	0	0	1	0	0

x y z	f(x,y,z)
000	0
001	1
010	0
011	1
100	1
101	1
110	0
111	0

1

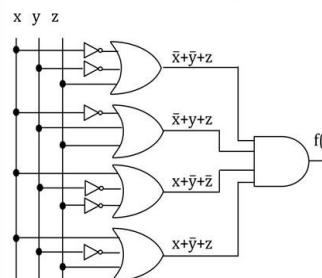
2

3

4

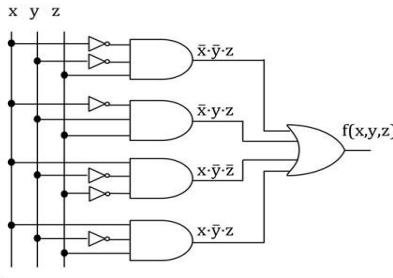
Espressione Somma di Prodotti associata
 $f(x,y,z) = (\bar{x}+\bar{y}+z) \cdot (\bar{x}+y+\bar{z}) \cdot (x+\bar{y}+\bar{z}) \cdot (x+y+z)$

Rete Combinatoria AND to OR equivalente



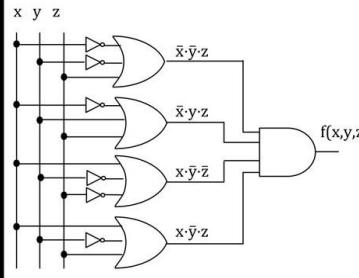
Espressione Somma di Prodotti associata
 $f(x,y,z) = \bar{x} \cdot \bar{y} \cdot z + \bar{x} \cdot y \cdot z + x \cdot \bar{y} \cdot z + x \cdot y \cdot z$

Rete Combinatoria AND to OR equivalente



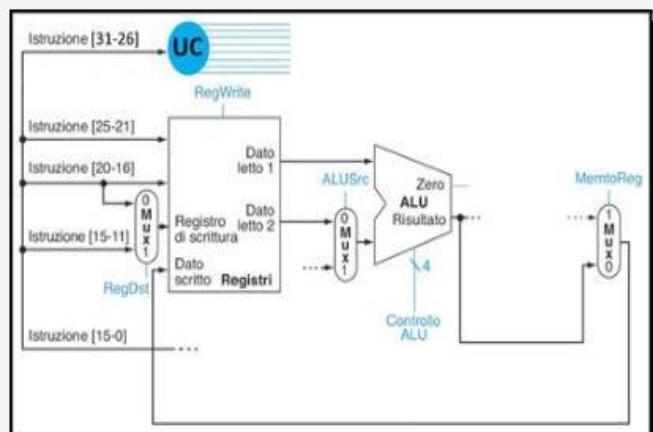
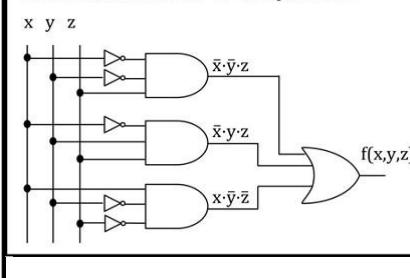
Espressione Somma di Prodotti associata
 $f(x,y,z) = \bar{x} \cdot \bar{y} \cdot z + \bar{x} \cdot y \cdot z + x \cdot \bar{y} \cdot z + x \cdot y \cdot z$

Rete Combinatoria AND to OR equivalente



Espressione Somma di Prodotti associata
 $f(x,y,z) = \bar{x} \cdot \bar{y} \cdot z + \bar{x} \cdot y \cdot z + x \cdot \bar{y} \cdot z$

Rete Combinatoria AND to OR equivalente



Q773. > <immagine> Nella parte della CPU a ciclo singolo relativa alle

istruzioni Aritmetico-Logiche MIPS di Tipo R riportata in figura, il multiplexer controllato dal segnale MemtoReg seleziona:

- A. L'operazione eseguita dall'ALU
- B. Il secondo operando dell'ALU
- C. L'indirizzo del Registro del processore in cui scrivere
- D. Il Dato da scrivere nel Registro del processore

Q776. Il numero di sequenze binarie diverse di lunghezza K è:

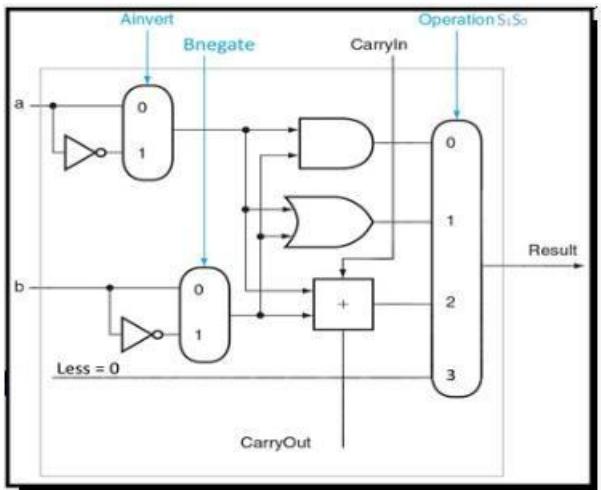
Il numero $2^{K-1}-1$
A. >

Il numero $2^{K-1}-1$

B. >

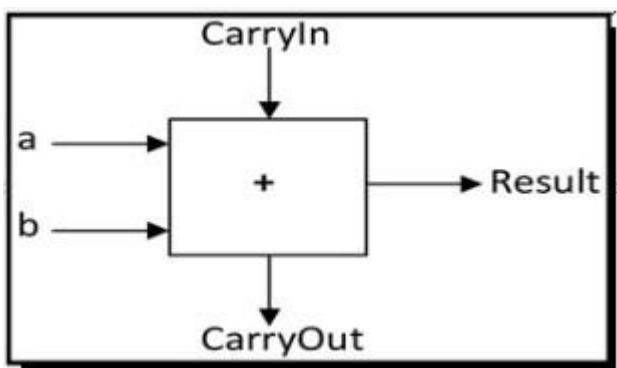
Il numero 2^{K-1}
C. >

D. Il numero 2^K



Il segnale output denotato con CarryOut nella Rete Combinatoria in figura è inviato:

- A. All'ALU ad un bit relativa alla posizione successiva come segnale di controllo del multiplexer 2 a 1 per la scelta della forma dell'operando da utilizzare
- B. All'ALU ad un bit relativa alla posizione successiva come segnale di controllo del multiplexer 4 a 1 per la scelta del risultato
- C. All'ALU ad un bit relativa alla posizione successiva come riporto input CarryIn
- D. Come output dell'ALU a 32 bit



Il simbolo grafico riportato in figura rappresenta:

- A. Un multiplexer 2 a 1
- B. L'ALU a un bit relativa alla singola posizione della sequenza
- C. L'ALU a 32 bit
- D. Il Sommatore completo che calcola l'Addizione dei bit relativi alla singola posizione della sequenza binaria, costruito utilizzando le Reti Combinatorie minimali

La traduzione in Assembly MIPS dell'assegnamento $A[k] = \text{val}$ con $A[]$ Array di INTERI è data:

1

A []	val	k
\$s3	\$t0	\$t7

```
sll $t1, $t7, 2
add $t6, $t1, $s3
sw $t0, 0 ($t6)
```

2

A []	val	k
\$s3	\$t0	\$t7

```
sll $t1, $t7, 2
sw $t0, 0 ($t1)
```

3

A []	val	k
\$s3	\$t0	\$t7

```
add $t6, $t7, $s3
sw $t0, 0 ($t6)
```

4

A []	val	k
\$s3	\$t0	\$t7

```
sw $t0, $t7 ($s3)
```

Q804. In Notazione in virgola fissa il valore della sequenza binaria 1001,011 è calcolato:

1

$$\begin{aligned}
 1001,011 &= \\
 &= 1x2^3 + 0x2^2 + 0x2^1 + 1x2^0 + 0x2^{-3} + 1x2^{-2} + 1x2^{-1} = \\
 &= 8 + 0 + 0 + 1 + 0 + 0.25 + 0,5 = \\
 &= 9,75
 \end{aligned}$$

2

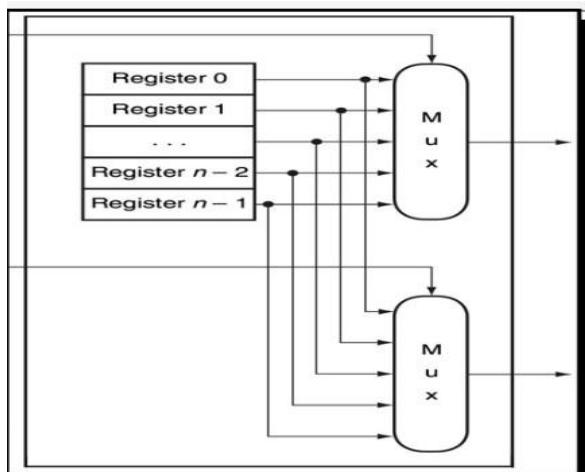
$$\begin{aligned}
 1001,011 &= \\
 &= 1x2^3 + 0x2^2 + 0x2^1 + 1x2^0 + 0x2^{-1} + 1x2^{-2} + 1x2^{-3} = \\
 &= 8 + 0 + 0 + 1 + 0 + 0.25 + 0,125 = \\
 &= 9,375
 \end{aligned}$$

3

$$\begin{aligned}
 1001,101 &= \\
 &= -1x2^3 + 0x2^2 + 0x2^1 + 1x2^0 + 0x2^{-3} + 1x2^{-2} + 1x2^{-1} = \\
 &= -8 + 1 + 0 + 0,25 + 0,5 = \\
 &= -6,25
 \end{aligned}$$

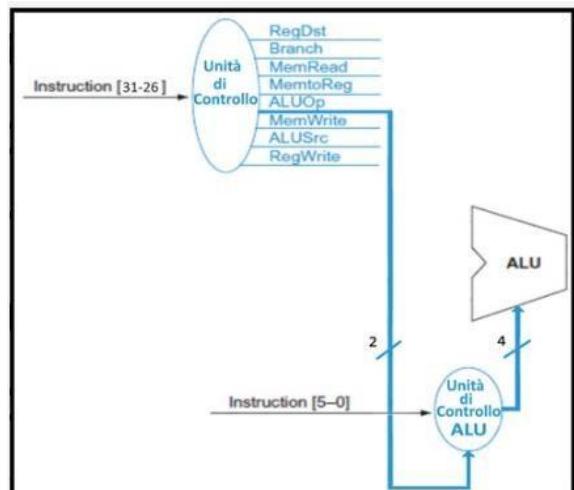
4

$$\begin{aligned}
 1001,101 &= \\
 &= -1x2^3 + 0x2^2 + 0x2^1 + 1x2^0 + 0x2^{-1} + 1x2^{-2} + 1x2^{-3} = \\
 &= -8 + 0 + 0 + 1 + 0 + 0,25 + 0,125 = \\
 &= -6,625
 \end{aligned}$$



Il grafico in figura relativo al processore MIPS rappresenta:

- A. Il circuito di Scrittura dei Registri del processore
- B. Il circuito di Lettura dei Registri del processore
- C. La parte del circuito della CPU coinvolta nell'aggiornamento del Program Counter
- D. La parte del circuito della CPU coinvolta nella fase di Prelievo (Fetch) dell'istruzione



In figura, quando il primo livello di decodifica fornisce il segnale di controllo ALUop=01, il secondo livello di decodifica fornisce i valori dei 4 segnali di controllo dell'ALU in base ai quali l'ALU esegue:

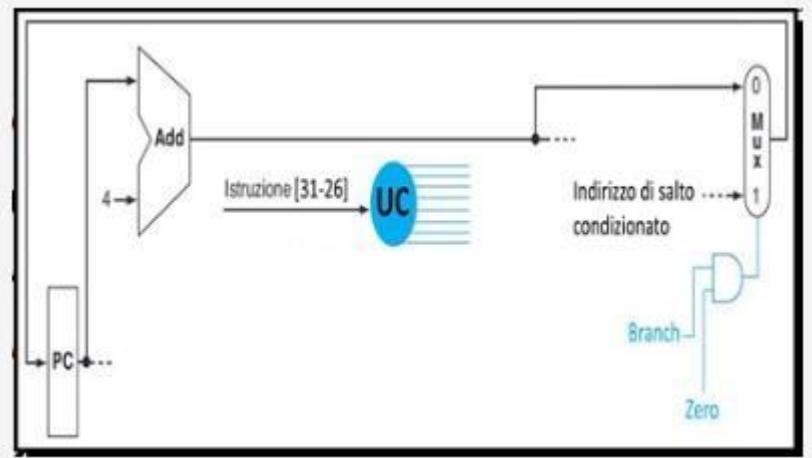
- A. Una Addizione per calcolare l'indirizzo di accesso in Memoria nell'esecuzione di lw e sw , indipendentemente dal valore del campo funct
- B. Una Sottrazione per verificare la condizione di uguaglianza tra i Registri nell'esecuzione di beq, indipendentemente dal valore del campo funct
- C. Il calcolo del valore del segnale Zero utilizzato per l'esecuzione dell'istruzione di salto condizionato
- D. Una operazione stabilita in base al valore del campo funct Istruzione[5-0] per l'esecuzione delle istruzioni Aritmetico-Logiche di Tipo R

Q814. L'istruzione WHILE nel Linguaggio Assembly:

- A. Viene tradotta da una analoga istruzione WHILE che permette l'esecuzione ripetuta di istruzioni fino a quando il valore di verità di una espressione logica è vero
- B. Viene tradotta scrivendo un numero di copie delle istruzioni da ripetere pari al numero di ripetizioni
- C. Viene tradotta come una chiamata di procedura che permette l'esecuzione ripetuta di istruzioni fino a quando il valore di verità di una espressione logica è vero
- D. Non ha una corrispondente istruzione e viene tradotta combinando istruzioni di salto condizionato ed incondizionato

Q814. Le istruzioni in Linguaggio Macchina MIPS che traducono le istruzioni logiche andi e ori:

- A. Hanno Formato di Tipo R e Indirizzamento immediato
- B. Hanno Formato di Tipo I e Indirizzamento tramite Base e Offset
- C. Hanno Formato di Tipo I e Indirizzamento immediato
- D. Hanno Formato di Tipo I e Indirizzamento pseudodiretto



Nello schema in figura, quando si ha il valore del segnale di

controllo Branch=1 e il segnale generato dall'ALU Zero=0 il multiplexer seleziona:

- A. L'indirizzo di salto condizionato inviato sull'ingresso dati 1 dalla parte del circuito della CPU che lo calcola
- B. L'indirizzo dell'istruzione successiva a quella in esecuzione calcolato dal Sommatore ed inviato sull'ingresso dati 0
- C. L'indirizzo calcolato dall'ALU inviato sull'ingresso dati 1
- D. L'indirizzo dell'istruzione attualmente in esecuzione

Q816. La traduzione in Assembly MIPS dell'assegnamento val = A[k] con A[] Array di INTERI è data:

1

2

3

4

A []	val	k
\$s3	\$t0	\$t7

```
sll $t1, $t7, 2
add $t6, $t1, $s3
lw $t0, 0 ($t6)
```

A []	val	k
\$s3	\$t0	\$t7

```
add $t6, $t7, $s3
lw $t0, 0 ($t6)
```

A []	val	k
\$s3	\$t0	\$t7

```
lw $t0, $t7 ($s3)
```

A []	val	k
\$s3	\$t0	\$t7

```
sll $t1, $t7, 2
lw $t0, 0 ($t1)
```

Q821. Per assegnare il valore 1 al terminale output individuato tra 16 possibili uscite disponibili e il valore 0 ai rimanenti output si utilizza il modulo combinatorio:

A >

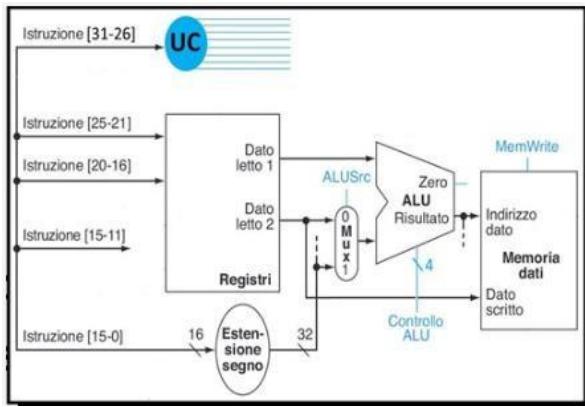
Decodificatore 16 a 2^{16}

B > Decodificatore 4 a $2^4 = 16$

C > Multiplexer 2⁴=16 a 1

Multiplexer 2¹⁶ a 1

D >



Q827. > Nella parte della CPU a ciclo singolo relativa all'istruzione STORE word riportata in figura, l'indirizzo di accesso in Memoria proviene:

- A. Direttamente dal valore contenuto nel campo Istruzione[15-0] Esteso di segno a 32 bit
- B. Direttamente dal terminale output Dato letto 1 del blocco dei Registri, che fornisce il contenuto del registro di indirizzo Istruzione [25-21] letto anticipatamente
- C. Direttamente dal terminale output Dato letto 2 del blocco dei Registri, che fornisce il contenuto del registro di indirizzo Istruzione [20-16] letto anticipatamente
- D. Direttamente dal terminale output dell'ALU che fornisce il risultato dell'addizione del contenuto nel Registro Base con indirizzo nel campo Istruzione[25-21] con il valore dell'Offset contenuto nel campo Istruzione[15-0] Esteso di segno a 32 bit

Q828. La misura dell'errore di arrotondamento della rappresentazione troncata $79854,84000$ del numero $79854,84219$ è limitata superiormente:

1

Dal valore $10^5 = 0,00001$

2

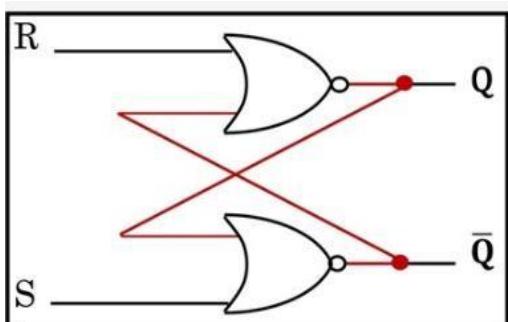
Dal valore $10^{-1} = 0,1$

3

Dal valore $10^{-3} = 0,001$

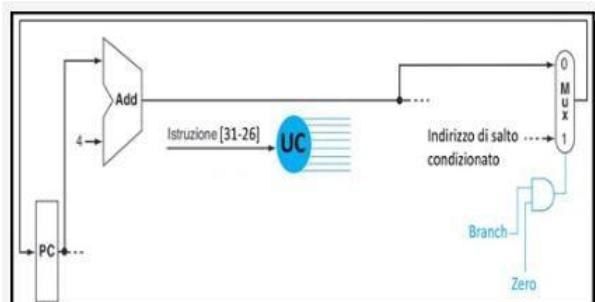
4

Dal valore $10^{-2} = 0,01$



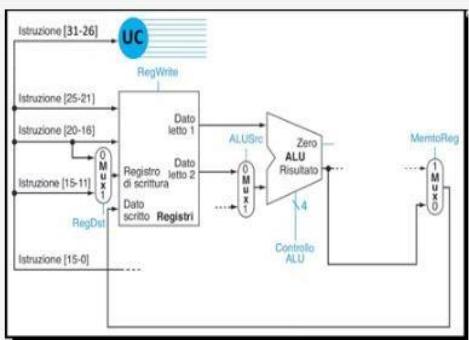
A partire dal LATCH SR riportato in figura, il circuito del Flip/Flop SR si ottiene:

- A. Ponendo $R = \text{NOT}(S)$
- B. Ponendo $S = \text{NOT}(R)$
- C. Introducendo il segnale clock posto in AND con ciascuno degli input S ed R
- D. Eliminando la temporizzazione



quando si ha il valore del segnale di controllo Branch=0 il multiplexer seleziona:

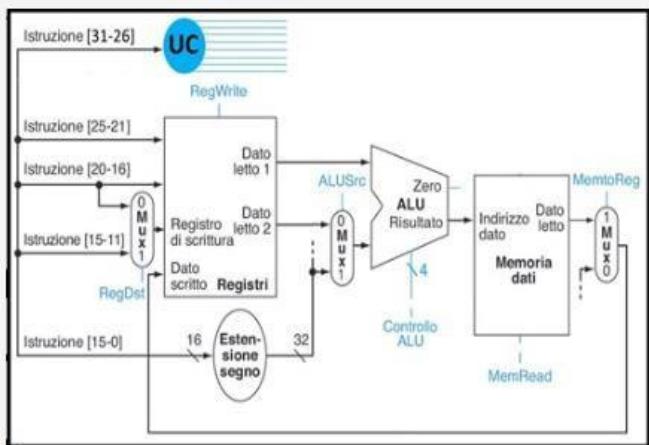
- A. L'indirizzo calcolato dall'ALU inviato sull'ingresso dati 1
- B. L'indirizzo di salto condizionato inviato sull'ingresso dati 1 dalla parte del circuito della CPU che lo calcola
- C. L'indirizzo dell'istruzione successiva a quella in esecuzione calcolato dal Sommatore ed inviato sull'ingresso dati 0
- D. L'indirizzo dell'istruzione attualmente in esecuzione



Q835. > Nella parte della CPU a ciclo singolo relativa alle istruzioni Aritmetico-Logiche

MIPS di Tipo R riportata in figura, con il valore del segnale di controllo RegDst=1 il multiplexer seleziona:

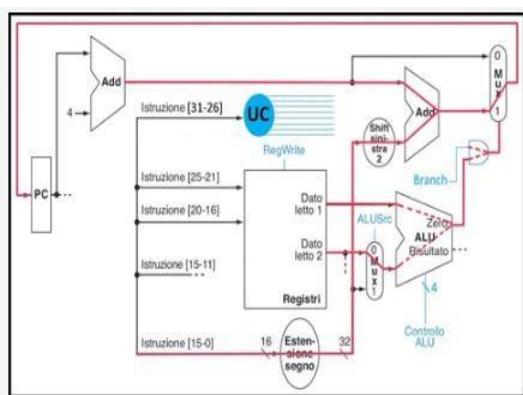
- L'operazione eseguita dall'ALU
- Il risultato calcolato dall'ALU come Dato da scrivere nel registro
- Il contenuto del campo di 5 bit Istruzione[15-11] come indirizzo del Registro del processore in cui scrivere il Dato
- Il contenuto del Registro del processore letto in anticipo disponibile sul terminale output Dato letto 2 come secondo operando dell'ALU



Q852. > Nella parte della CPU a ciclo singolo relativa all'istruzione

LOAD word riportata in figura, il multiplexer controllato dal segnale AluSrc effettua la selezione:

- In base al valore AluSrc=0, che instrada in output il valore nel campo Istruzione[15-0], come operando dell'ALU
- In base al valore AluSrc=0, che instrada in output il valore Dato letto 2 letto anticipatamente nel blocco dei Registri, come operando dell'ALU
- In base al valore AluSrc=1, che instrada in output il valore nel campo Istruzione[15-0] Esteso di segno a 32 bit, come operando dell'ALU
- In base al valore AluSrc=1, che instrada in output il valore Dato letto 1 letto anticipatamente nel blocco dei Registri, come operando dell'ALU



la computazione che si svolge nella parte del circuito della CPU MIPS a ciclo singolo relativa a:

- Esecuzione dell'istruzione Store word
- Esecuzione dell'istruzione di salto condizionato su uguaglianza BEQ
- Esecuzione delle istruzioni Aritmetico-Logiche di Tipo R
- Esecuzione dell'istruzione Load word

La tecnica detta mappatura diretta di una cache costituita da $K=2^S$ locazioni pone:

- Gli indirizzi della cache costituita da $>$ locazioni uguali agli indirizzi delle prime K locazioni della Memoria principale
- Gli indirizzi della cache uguali allo stesso valore degli indirizzi delle locazioni della Memoria principale
- Gli indirizzi della cache costituita da $>$ locazioni uguali al valore modulo s degli indirizzi delle locazioni della Memoria principale, che coincide con il valore rappresentato dagli s bit meno significativi di tali indirizzi
- Gli indirizzi della cache costituita da $>$ locazioni uguali al valore degli indirizzi delle ultime K locazioni della Memoria principale

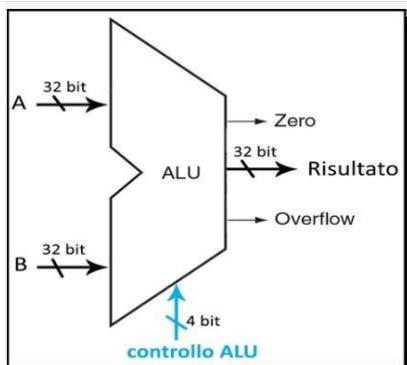
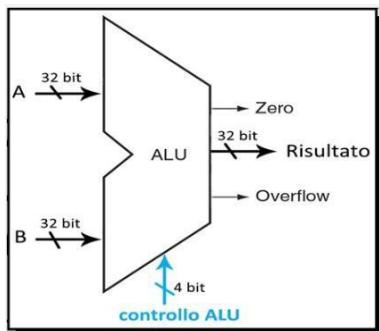


grafico dell'ALU in figura, i 4 bit del segnale indicato col nome controllo ALU, che fornisce i valori dei segnali Ainvert, Bnegate, OperationS1S0, per l'esecuzione dell'istruzione ADD sono posti uguali a:

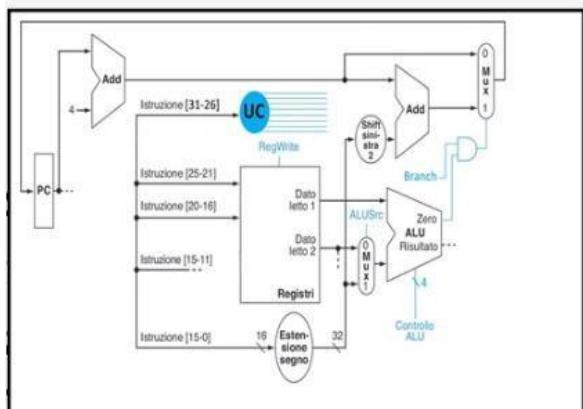
- A. controllo ALU = 1100
- B. controllo ALU = 0101
- C. controllo ALU = 0000
- D. controllo ALU = 0010



Q863. > <immagine> Nel simbolo grafico dell'ALU in figura, i 4 bit del segnale indicato col nome controllo

ALU, che fornisce i valori dei segnali Ainvert, Bnegate, OperationS1S0, per l'esecuzione dell'istruzione NOR sono posti uguali a:

- A. controllo ALU = 0000
- B. controllo ALU = 0010
- C. controllo ALU = 0110
- D. controllo ALU = 1100



Q871. > <immagine> Nella parte della CPU a ciclo singolo relativa all'istruzione BEQ

riportata in figura, l'indirizzo di salto condizionato su uguaglianza è calcolato:

- A. Dall'ALU che riceve come operandi il contenuto dei Registri di indirizzo Istruzione[25-21] e Istruzione[20-16] letti anticipatamente, ed effettua la sottrazione in base al valore del segnale Controllo ALU
- B. Dal Sommatore a destra che riceve come operandi il contenuto del Program Counter incrementato di 4 e il contenuto nel campo Istruzione[15-0] Esteso di segno a 32 bit e Shiftato a sinistra di 2 posizioni
- C. Dal Sommatore a sinistra che riceve come operando il contenuto del Program Counter e la costante 4
- D. Dall'ALU che riceve come operando il contenuto del Program Counter incrementato di 4 e il contenuto nel campo Istruzione[15-0] Esteso di segno a 32 bit e Shiftato a sinistra di 2 posizioni, ed effettua l'operazione in base al valore del segnale Controllo ALU

Q872. Nella Notazione in complemento a 2, il minimo numero rappresentabile con sequenze di lunghezza M è:

- 1**
- 2**
- 3**
- 4**

Il valore -2^{M-1} Il valore -2^M Il valore -2^M Il valore $-2^{M-1} - 1$

```

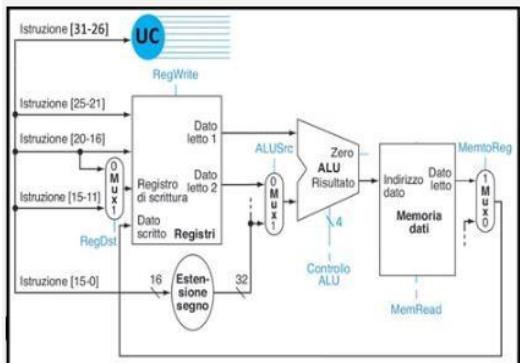
for (n = 0; n != Max; n = n+1) {
    istruzione_1
}
istruzione_2

PONENDO:   n      → $t0
            Max → $t1

```

Q873. > <immagine> Per ottenere lo schema dell'istruzione FOR mostrato in figura la traduzione in Assembly MIPS della condizione logica su disuguaglianza è:

- A. L'istruzione beq \$t0, \$t1, END_FOR, dove END_FOR è l'etichetta della istruzione_2
- B. L'istruzione j END_FOR, dove END_FOR è l'etichetta della istruzione istruzione_2
- C. L'istruzione beq \$t0, \$t1, CICLO, dove CICLO è l'etichetta che porta alla ripetizione del ciclo FOR
- D. L'istruzione j CICLO dove CICLO è l'etichetta che porta alla ripetizione del ciclo FOR



Q877. > <immagine> Nella parte della CPU a ciclo singolo relativa all'istruzione LOAD word riportata in figura, l'indirizzo di accesso in Memoria proviene:

- A. Direttamente dal terminale output dell'ALU che fornisce il risultato dell'addizione del contenuto nel Registro Base con indirizzo nel campo Istruzione[25-21] con il valore dell'Offset contenuto nel campo Istruzione[15-0] Esteso di segno a 32 bit
- B. Direttamente dal terminale output Dato letto 1 del blocco dei Registri, che fornisce il contenuto del registro di indirizzo Istruzione [25-21] letto anticipatamente
- C. Direttamente dal terminale output Dato letto 2 del blocco dei Registri, che fornisce il contenuto del registro di indirizzo Istruzione [20-16] letto anticipatamente
- D. Direttamente dal valore contenuto nel campo Istruzione[15-0] Esteso di segno a 32 bit

	INPUT		OUTPUT
	campo funct	ALUop	segnali di controllo ALU
lw	somma	-----	00 0010
sw	somma	-----	00 0010
beq	sottrazione	-----	01 0110
add	Tipo R	somma	100000 10 0010
sub	Tipo R	sottrazione	100010 10 0110
and	Tipo R	AND	100100 10 0000
or	Tipo R	OR	100101 10 0001
nor	Tipo R	NOR	100111 10 1100

La tabella riportata in figura rappresenta:

- A. La Tavola di verità della funzione Booleana costituita dalla relazione input-output dell'Unità di Controllo che effettua il SECONDO livello di decodifica nella CPU MIPS a ciclo singolo per l'esecuzione delle istruzioni elencate in rosso
- B. La Tavola di verità della funzione Booleana costituita dalla relazione input-output dell'Unità di Controllo che effettua il PRIMO livello di decodifica nella CPU MIPS a ciclo singolo per l'esecuzione delle istruzioni elencate in rosso
- C. I segnali di controllo memorizzati nei Registri del processore per l'esecuzione delle istruzioni elencate in rosso
- D. Il formato in Linguaggio Macchina delle istruzioni elencate in rosso

```

if ( test == 0 ) {
    istruzione_1
} else {
    istruzione_2
}
istruzione_3

PONENDO: test → $t0

```

Q890. > <immagine> Per mantenere lo schema dell'istruzione IF-ELSE mostrato in figura la traduzione in Assembly MIPS della condizione logica su ugualianza è:

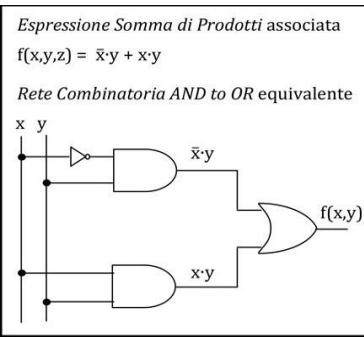
- A. L'istruzione bne \$t0, \$zero, ELSE, dove ELSE è l'etichetta di istruzione_2
- B. L'istruzione j ELSE, dove ELSE è l'etichetta di istruzione_2
- C. L'istruzione bne \$t0, \$zero, END_IF, dove END_IF è l'etichetta di istruzione_3

D. L'istruzione j END_IF dove END_IF è l'etichetta di istruzione_3

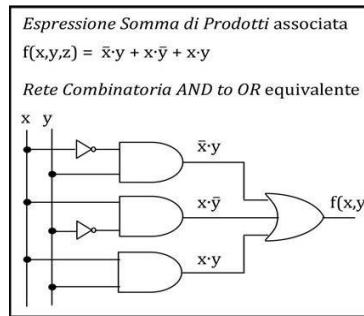
x	y	f(x,y)
00	0	0
01	1	1
10	1	1
11	0	0

La Sintesi di una Rete Combinatoria che calcola la funzione Booleana in figura è ottenuta mediante:

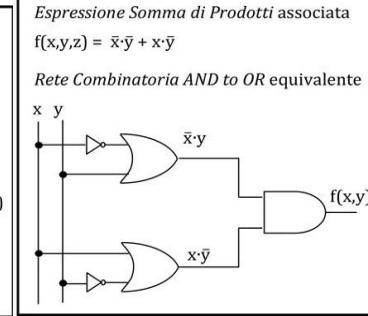
1



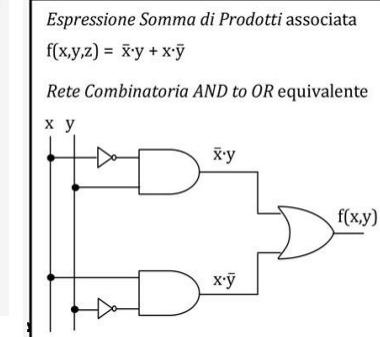
2



3



4



Q900. In Notazione posizionale pesata la cifra più significativa di una sequenza di lunghezza K è associata al peso:

1

Dato dal valore 2^K

2

Dato dal valore 2^0

3

Dato dal valore 2^{K-1}

4

Dato dal valore 2^{K-1}

Q901. Nella Notazione in complemento a 2, il peso del bit più a sinistra in una sequenza di lunghezza K è:

1

2

3

4

Il valore -2^{K-1}

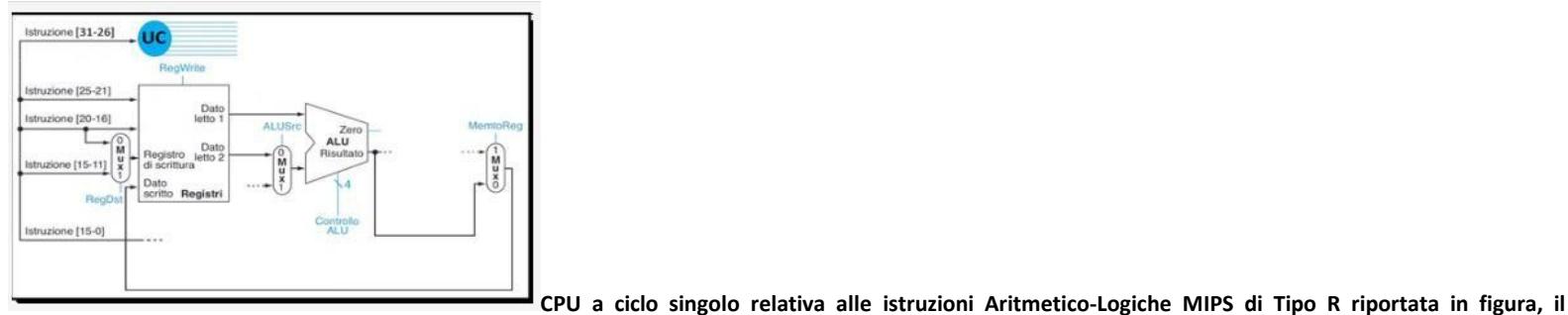
Il valore -2^{K-1}

Il valore -2^{K-1}

Il valore $+2^{K-1}$

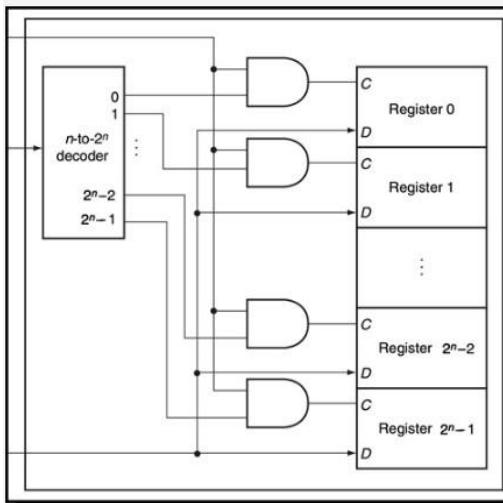
Il valore -2^{K-1}

Il valore -2^{K-1}



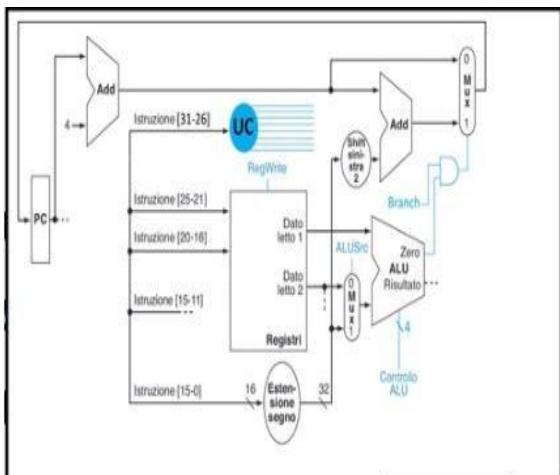
multiplexer controllato dal segnale RegDst seleziona:

- A. Il Dato da scrivere nel Registro del processore
- B. L'indirizzo del Registro del processore in cui scrivere
- C. Il secondo operando dell'ALU
- D. L'operazione eseguita dall'ALU



Il grafico in figura relativo al processore MIPS rappresenta:

- A. La parte del circuito della CPU coinvolta nell'aggiornamento del Program Counter
- B. Il circuito di Lettura dei Registri del processore
- C. La parte del circuito della CPU coinvolta nella fase di Prelievo (Fetch) dell'istruzione
- D. Il circuito di Scrittura dei Registri del processore



Nella parte della CPU a ciclo singolo relativa all'istruzione BEQ riportata in figura, il valore

del segnale Zero influenza:

- A. La scelta effettuata dal multiplexer che seleziona l'indirizzo da scrivere nel Program Counter in base all'output della porta and che riceve in input il segnale di controllo Branch
- B. Il calcolo effettuato dal Sommatore per fornire l'indirizzo di salto condizionato
- C. Il calcolo effettuato dall'ALU per fornire l'indirizzo di salto condizionato
- D. La scelta effettuata dal multiplexer che seleziona l'indirizzo da scrivere nel Program Counter in base al segnale RegDst