# Esami 2018

giovedì 30 maggio 2019 16:42

**Esercizio 1 (3 punti):**
Dati i numeri:

a) 000111

e

b) 101101

effettuarne la somma binaria usando la rappresentazione in base 2 per numeri senza segno ed usando la codifica in complemento a 2 per i numeri relativi. In entrambi i casi, specificare il numero decimale corrispondente agli addendi ed al risultato.

BINARIO PURO

000111 → 7
101101 → 45
110100 ⇒ 52

COMP A 2

000111 → 7
101101 ⇒ -19
110100 ⇒ -12

**Esercizio 2 (3 punti):**
Considerando la codifica nello standard IEEE 754 in precisione singola dire quale tra i numeri

a) 1 01111111 11100000000000000000000
b) 1 01111111 11000000000000000000000

è il minore.

a: Segno: -(1)
Exp = 127 - 127 = 0
Mantissa: 111
Numero: 1,111
−1,875

b: Segno: -(1)
Exp = 0
Mantissa: 11
Numero: 1,11
−1,75

**Esercizio 3 (3 punti):**
Dimostrare la verità o meno della proprietà distributiva dell'OR rispetto allo XOR

A OR (B XOR C) = (A OR B) XOR (A OR C)

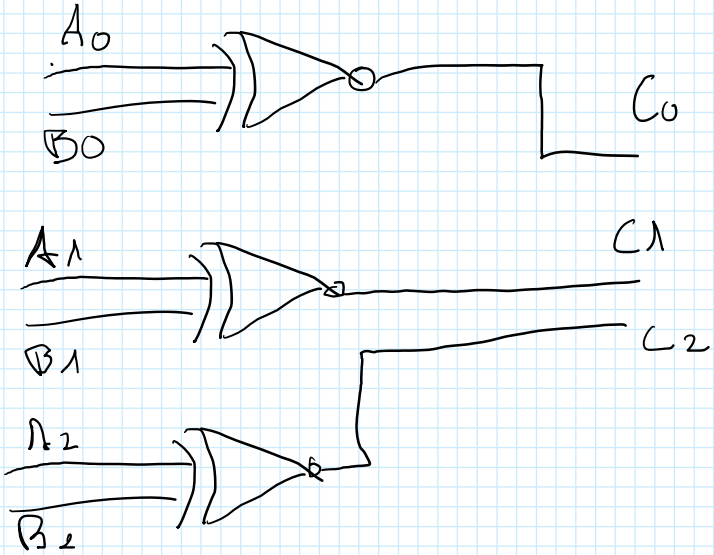
usando la tabella di verità presente nel modulo risposte.

Non distributiva $F_0 \neq F_1$

| A | B | C | B xor C | $F_0$ | A OR B | A OR C | $F_1$ |
|---|---|---|---|---|---|---|---|
| 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| 0 | 0 | 1 | 1 | 1 | 0 | 1 | 1 |
| 0 | 1 | 0 | 1 | 1 | 1 | 0 | 1 |
| 0 | 1 | 1 | 0 | 0 | 1 | 1 | 0 |
| 1 | 0 | 0 | 0 | 1 | 1 | 1 | 0 |
| 1 | 0 | 1 | 1 | 0 | 1 | 1 | 0 |

| | | | | | | | |
|---|---|---|---|---|---|---|---|
| 1 | 0 | 0 | 0 | 1 | 1 | 1 | 0 |
| 1 | 0 | 1 | 1 | 1 | 1 | 1 | 0 |
| 1 | 1 | 0 | 1 | 1 | 1 | 1 | 0 |
| 1 | 1 | 1 | 0 | 1 | 1 | 1 | 0 |

**Esercizio 4 (3 punti):**
Disegnare la rete logica che realizza il circuito combinatorio comparatore di 2 ingressi a 3 bit




**Esercizio 5 (3 punti):**
Con riferimento all'interprete micro-programmato Mic-1, quali delle seguenti affermazioni sono vere?

A. Durante l'esecuzione della micro-istruzione Main1 viene sempre richiesto il fetch dell'argomento dell'istruzione in esecuzione; F
B. Durante l'esecuzione della micro-istruzione Main1 può essere richiesto il fetch del codice operativo della prossima micro-istruzione; V
C. Durante l'esecuzione di una micro-istruzione ~~che contenga~~ che richieda una lettura dalla memoria (rd), nessuna altra lettura dalla memoria o scrittura verso la memoria deve essere già in corso; F
D. Il valore dei flag N e Z dell'ALU non sono alterati dalla micro-istruzione MDR=TOS; F

**Esercizio 7 (3 punti):**
Nell'ambito dell'architettura MIC-1 si descriva la relazione fra gli indirizzi nel control store delle due micro-istruzioni raggiungibili come destinazioni alternative di una istruzione di tipo jump (JAMN e/o JAMZ uguali ad 1).

La relazione tra i due indirizzi è, se ad esempio prendiamo un indirizzo base 0x32, per saltare dall'altra parte della tabella lo mettiamo in OR con 0x100 (256)

**Esercizio 9 - laboratorio (4 punti)**
Utilizzando il linguaggio assemblativo nel formato JAS visto in laboratorio, scrivere un metodo COMP1 con 3 parametri formali (chiamateli X, Y e K) che restituisca al chiamante il piu' grande tra X e Y se l'espressione 2X+Y-K è negativa, oppure, in caso contrario, che restituisca il piu' piccolo tra X e Y. Scrivere anche il main contenente il codice che realizzi la chiamata di tale metodo con parametri attuali rispettivamente -10, 4, 3 (in notazione decimale) e che scriva il risultato restituito dal metodo in una sua variabile locale chiamata value. Si limiti al minimo l'introduzione di variabili inutili.

COMP1(X, Y, K)

.var

.endvar

ILOAD X
DUP
IADD
ILOAD Y
IADD
ILOAD K
ISUB
IFLT L1 ← se val < 0
ILOAD X
ILOAD Y
ISUB

IFLT L2 ← se X < Y
ILOAD Y
IRETURN
L2: ILOAD X
IRETURN
L1: ILOAD X
ILOAD Y
ISUB
IFLT L3 ← se X < Y
ILOAD X
IRETURN
L3: ILOAD Y
IRETURN

**Esercizio 10 - laboratorio (4 punti)**
Scrivere il microcodice MIC1 dell'istruzione senza operandi **LOCAND**, che scrive sulla cima dello stack il risultato dell'AND bit-a-bit tra le due variabili locali con scostamento 1 e 2 da LV, assumendo che tale microcodice vada a modificare il microinterprete. Si descrivano quindi anche quali modifiche devono essere fatte al file di configurazione dell'emulatore Mic1MMV e al codice del microinterprete stesso affinché l'emulatore possa eseguire un programma IJVM (.jas) contenente l'istruzione **LOCAND**.

LOCAND1 MAR = LV + 1; rd
LOCAND2 MAR = LV + 2; rd
LOCAND3 H = MDR;
LOCAND4 MAR = SP = SP +1
LOCAND5 MDR = H AND MDR; wr; goto Main1

**Esercizio 9 - laboratorio (4 punti)**
Utilizzando il linguaggio assemblativo nel formato JAS visto in laboratorio, scrivere un programma che dati due numeri interi positivi e maggiori di zero X e Y scrive sullo stack i numeri ottenuti dalla progressiva sottrazione di Y da X, fino a quando X non assume un valore negativo (la serie inizia sempre con X). Il programma deve implementare l'esecuzione dell'esercizio con i dati di esempio X=9, Y=2. In questo caso, alla fine dell'esecuzione lo stack dovrebbe contenere:

1 ← SP
3
5
7
9

```
BIPUSH 9        ILOAD x
ISTORE x        ILOAD y
BIPUSH 2        ISUB
ISTORE y        DUP
                ISTORE x
Sub:            Goto Sub
ILOAD x         End:
IFLT end        halt
```

*...izioni diverse devono rivolgersi ai docenti.*

**Esercizio 1 (3 punti):**
Dati i numeri:

a) 10011100

e

b) 10011101

Dire quale dei due è maggiore nel caso in cui si usi la rappresentazione in base 2 per numeri senza segno e nel caso in cui si usi la codifica in complemento a 2 per i numeri relativi. Motivare la risposta con spiegazioni, passaggi e calcoli. Il solo risultato finale non sarà considerato sufficiente in fase di valutazione.

convertito in binario

1001 1100 → 8 + 16 + 32 + 128 = 184

1001 1101 → 1 + 4 + 8 + 16 + 128 = 157

In rappresentazione in 2 per numeri senza segno è maggiore 10011100

10011100 (complemento a 2) } -100
01100100 (invertito)
100

10011101 } -99
01100011
1632 1

Un complemento a 2 è più grande 1001 1101

**Esercizio 2 (3 punti):**
Calcolare la codifica nello standard IEEE 754 in precisione singola del numero -1,75

$1_{10} = 1_2$

| 0,75 | 1 |
|---|---|
| 0,5 | 1 |
| 0 | |

$1,75 = 1,11$

segno: $-(1)$

Esponente: $0 + 127 = 127$ (Esponente in eccesso a $2^{8-1} - 1$)

0111 1111

Mantissa: 11

Numero:

1011 1111 1110 0000 0000 0000 0000 0000

**Esercizio 3 (3 punti):**
Scrivere in forma normale disgiuntiva ed in forma normale congiuntiva la seguente funzione booleana

F = (JAMN AND N) OR (JAMZ AND Z)

usando la tabella di verità presente nel modulo risposte.

Ritaglio schermata acquisito: 31/05/2019 18:33

| JAMN | JAMZ | N | Z | JAMN AND N | JAMZ AND Z | F |
|---|---|---|---|---|---|---|
| 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| 0 | 0 | 0 | 1 | 0 | 0 | 0 |
| 0 | 0 | 1 | 0 | 0 | 0 | 0 |
| 0 | 0 | 1 | 1 | 0 | 0 | 0 |
| 0 | 1 | 0 | 0 | 0 | 0 | 0 |
| 0 | 1 | 0 | 1 | 0 | 1 | 1 |
| 0 | 1 | 1 | 0 | 0 | 0 | 0 |
| 0 | 1 | 1 | 1 | 0 | 1 | 1 |
| 1 | 0 | 0 | 0 | 0 | 0 | 0 |
| 1 | 0 | 0 | 1 | 0 | 0 | 0 |

| | | | | | | |
|---|---|---|---|---|---|---|
| 1 | 0 | 0 | 0 | 0 | 0 | 0 |
| 1 | 0 | 0 | 1 | 1 | 0 | 1 |
| 1 | 0 | 1 | 0 | 1 | 0 | 1 |
| 1 | 0 | 1 | 1 | 0 | 0 | 0 |
| 1 | 1 | 0 | 0 | 0 | 1 | 1 |
| 1 | 1 | 0 | 1 | 0 | 0 | 1 |
| 1 | 1 | 1 | 0 | 1 | 0 | 1 |
| 1 | 1 | 1 | 1 | 1 | 1 | 1 |

FNC: $(J_A N + J_A Z + N + Z)$

$(J_N + J_Z + N + \bar{Z})$

$(J_N + J_Z + \bar{N} + Z)$

$(J_N + J_Z + \bar{N} + \bar{Z})$

$(J_N + \bar{J}_Z + N + Z)$

$(J_N + \bar{J}_Z + \bar{N} + Z)$

$(\bar{J}_N + J_Z + N + Z)$

$(\bar{J}_N + J_Z + \bar{N} + Z)$

$(J\bar{N} + \bar{J}Z + N + Z)$

FND: $\bar{J}_A N \; J_A Z \; \bar{N} \bar{Z} +$

$\bar{J}_A N \; J_A Z \; N \bar{Z}$

$J_A N \; \bar{J}_A Z \; N Z +$

$J_A N \; \bar{J}_A Z \; N Z +$

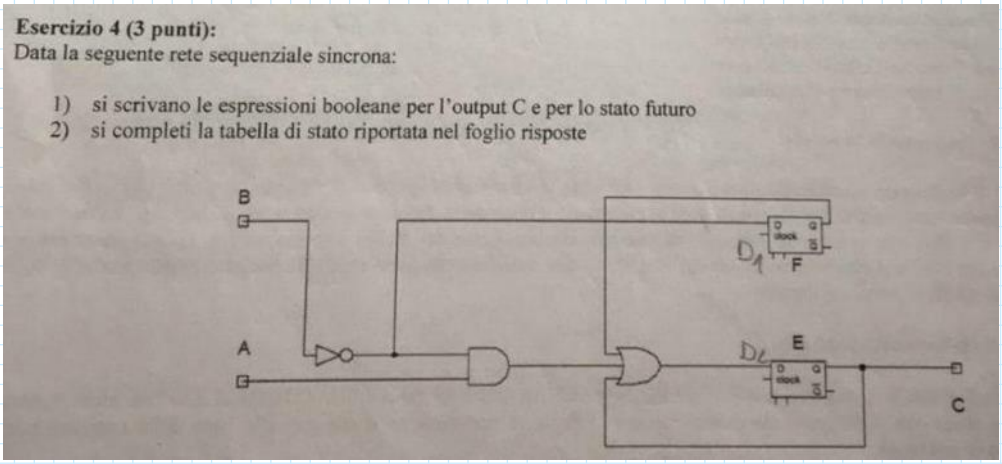
$J_A N \; J_A Z \; \bar{N} Z +$

$J_A N \; J_A Z \; N \bar{Z}$

$J_A N \; J_A Z \; N Z$

**Esercizio 4 (3 punti):**
Data la seguente rete sequenziale sincrona:

1) si scrivano le espressioni booleane per l'output C e per lo stato futuro
2) si completi la tabella di stato riportata nel foglio risposte




Punto 2?

Ritaglio schermata acquisito: 31/05/2019 18:44

$Q_2 = C = A\bar{B} + Q_1 + Q_2$

$Q_1 = \bar{B}$

**Esercizio 9 - laboratorio (4 punti)**
Utilizzando il linguaggio assemblativo nel formato JAS visto in laboratorio, scrivere un metodo di nome MINFACTOR con 2 parametri formali (chiamateli X e Y), entrambi interi positivi, che restituisca al chiamante il più piccolo intero K tale che K·X > Y. Scrivere anche il main contenente il codice che realizzi la chiamata di tale metodo con parametri attuali rispettivamente 5 e 10 (in notazione decimale) e che scriva il risultato restituito dal metodo in una variabile locale chiamata value (coi valori di esempio, il risultato è 3). Si limiti al minimo l'introduzione di variabili inutili.

```
.constant
objref 0xCAFE
.end-constant
.main
.var
.end-var
Ldc_w objref
BIPUSH 5
BIPUSH 10
INVOKEVIRTUAL minfactor
halt
.end-main
.method minfactor(x,y)
.var
k
var
factor
.end-var
L1:
ILOAD var
IFEQ L2
ILOAD x
ILOAD factor
IADD
ISTORE factor
IINC var -1
goto L1
L2:
// controlla se il risultato > y
ILOAD y
ILOAD factor
ISUB
IINC k 1
ILOAD k
ISTORE var
IFLT L3
goto L1
L3:
ILOAD k
IRETURN
.end-method
```

**Esercizio 1 (3 punti):**

Dati i numeri:

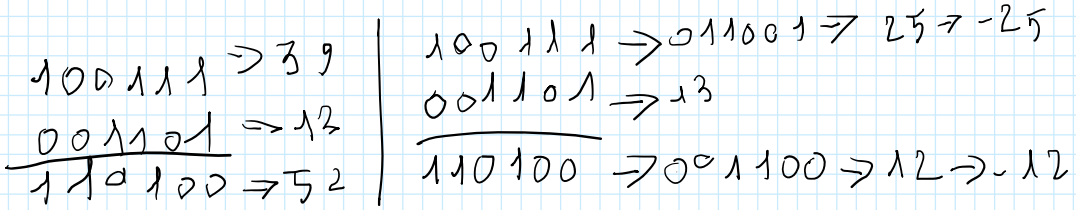
a.100111

e

a.001101

effettuarne la somma binaria usando la rappresentazione in base 2 per numeri senza segno ed usando la codifica in complemento a 2 per i numeri relativi. In entrambi i casi, specificare il numero decimale corrispondente agli addendi ed al risultato.

Ritaglio schermata acquisito: 31/05/2019 22:56




**Esercizio 2 (3 punti):**
Dato il numero -3.125 ricavare la sua codifica secondo lo standard IEEE 754 in precisione singola.

Ritaglio schermata acquisito: 31/05/2019 22:59

3 = 11

0,125 = 0,001

11,001

$1,1001 \cdot 2^1$

0,125 | 0
25 | 0
5 | 1
0

Segno: - (1)

E+P: 2 → 127+1: 128 → 1000 0000

Mantissa: 1000 1

Num

1100 0000 0100 0000

**Esercizio 3 (3 punti):**
Dimostrare la verità o la falsità della seguente identità :

(NOT(A) NOR NOT(B)) AND C = (A AND B) OR C

usando la tabella di verità presente nel modulo risposte o altri metodi.

| A | B | C | ¬A NOR ¬B | $f_1$ | $\overline{f_2}$ |
|---|---|---|---|---|---|
| 0 | 0 | 0 | 0 | 0 | 0 |
| 0 | 0 | 1 | 0 | 0 | 1 |
| 0 | 1 | 0 | 0 | 0 | 0 |
| 0 | 1 | 1 | 0 | 0 | 1 |
| 1 | 0 | 0 | 0 | 0 | 0 |
| 1 | 0 | 1 | 0 | 0 | 1 |
| 1 | 1 | 0 | 0 | 0 | 1 |
| 1 | 1 | 1 | 1 | 1 | 1 |

Le due espressioni sono diverse, come si nota dalla tabella di verità

**Esercizio 4 (3 punti):**
Disegnare la rete logica che ne realizza il circuito sommatore completo (full adder) a 4 bit utilizzando come componente elementare il sommatore completo a 1 bit.

**Esercizio 10 - laboratorio (4 punti)**
Scrivere il codice MIC1 di una nuova istruzione SWAPLOC12 che scambi tra loro i valori contenuti nelle prime due variabili locali.
Descrivere brevemente le modifiche da apportare alla configurazione dell'emulatore per aggiungere la suddetta istruzione.

MAR = LV +1; rd
MAR = LV+2; rd
H = MDR
OPC = MDR

MDR = H; wr
MAR = LV+1
MDR = OPC; wr; goto Main1

**Esercizio 2 (3 punti):**
Se esistesse il formato IEEE 754 in precisione *pessima* con 1 bit di segno, 3 bit di esponente e 3 bit di mantissa quale sarebbe il più grande numero normalizzato rappresentabile?

Il numero più grande rappresentabile sarebbe 1,000 * 2^3 = 8

**Esercizio 3 (3 punti):**
Si dimostri che

$$\overline{A+BC} = (\overline{A}\,\overline{B}) + (\overline{A}\,\overline{C})$$

utilizzando l'algebra di Boole e mostrando tutti i passaggi.

**COMMUTATIVA**
A+B=B+A; AB=BA;
**ASSOCIATIVA**
(A+B)+C=A+(B+C)=A+B+C;(AB)C = A(BC)=ABC
**DISTRIBUTIVA**
A(B+C)=(AB)+(AC); A+BC= (A+B)(A+C)

| AND | AND | Nome | OR |
|---|---|---|---|
| Annullamento | A0=0 | Identità | A+0=A |
| Identità | A1=A | Annullamento | A+1=1 |
| Idempotenza | AA=A | Idempotenza | A+A=A |
| Inverso | A!A=0 | Inverso | A+!A=1 |
| Assorbimento | A(A+B)=A | Assorbimento | A+AB=A |
| DeMorgan | !(AB)=!A+!B | DeMorgan | !(A+B)=!A!B |

IMPARARE A MEMORIA

$\overline{A} + \overline{BC} = (\overline{A}\,\overline{B}) + (\overline{A}\,\overline{C})$ De Morgan

$\overline{A} \cdot \overline{BC} = (\overline{A}\,\overline{B}) + (\overline{A}\,\overline{C})$ De Morgan

$\overline{A} \cdot (\overline{B}+\overline{C}) = (\overline{A}\,\overline{B}) + (\overline{A}\,\overline{C})$ Distributiva

$(\overline{A}\,\overline{B}) + (A\,C) = (\overline{A}\,\overline{B}) + (\overline{A}\,\overline{C})$

**Esercizio 1 (3 punti):**
Data la sequenza 111111 quale codifica per i numeri relativi tra modulo e segno, complemento a 1, complemento a 2 rappresenta il numero decimale più piccolo?

Ritaglio schermata acquisito: 01/06/2019 11:51

MODULO SEGNO:
Intervallo: $-2^5-1 ; 2^5-1$
$\Downarrow$
$-31$

CMP A 1
Intervallo
$-2^5-1 ; 2^5-1$
$\Downarrow$
$-31$

CMP A 2
Intervallo
$-2^5 ; 2^5-1$
$\Downarrow$
$-32$

Essendo che 11111 rappresenta l'ultimo valore negativo degli interi nella notazione Comp1, ho 1, neg, basta dire il mio numero appartiene in quell'intervallo

**Esercizio 10 - laboratorio (4 punti)**

Scrivere il microcodice MIC1 dell'istruzione **ADDOP byte**, che sostituisce la parola in cima alla stack con la somma tra la stessa e il valore intero con segno rappresentato in **byte**, assumendo che tale microcodice vada a modificare il microinterprete. Si descrivano quindi anche quali modifiche devono essere fatte al file di configurazione dell'emulatore Mic1MMV e al codice del microinterprete stesso affinché l'emulatore possa eseguire un programma IJVM (.jas) contenente l'istruzione **ADDOP**.

PC = PC + 1; fetch
H = MBR
MAR = SP
MDR = TOS + H ; wr ; goto Main1

**Esercizio 9 - laboratorio (4 punti)**
Si supponga che, all'interno di un programma scritto nel linguaggio assemblativo JAS visto in laboratorio (per l'architettura MIC1), sia definita la costante intera positiva N e una funzione F(X) che, ricevuto un valore X compreso tra 0 e N (estremi inclusi), restituisce un valore intero anch'esso compreso tra 0 e N (estremi inclusi).
Scrivere un metodo "invmin" che riceve come parametro un intero Y, e che trova il più piccolo intero X, compreso tra 0 e N (estremi inclusi), tale che F(X) = Y. Se tale valore intero non viene trovato, la funzione deve ritornare il valore -1. Scrivere anche la definizione della costante N e il main contenente il codice che realizzi la chiamata del metodo invmin, usando dei valori a scelta per N e per Y. Come esempio (particolare!) di funzione F, assumere che nel programma sia già presente il codice seguente:

```
.method F(X)
        LDC_W N
        ILOAD X
        ISUB
        IRETURN
.end-method
```

**Esercizio 2 (3 punti):**
Se esistesse il formato IEEE 754 in precisione *infima* con 1 bit per il segno, 4 bit per l'esponente e 2 bit per la mantissa quale numero sarebbe rappresentato dalla sequenza 1 1101 10?

Motivare la risposta con spiegazioni, passaggi e calcoli. Il solo risultato finale non sarà considerato sufficiente in fase di valutazione.

1 1101 10
Segno: -(1) Decodifica

Esp: $1101 \Rightarrow 13_{10} \rightarrow 13 - 2^{4-1} - 1 = 13 - 7 = +6$

Mantissa: 1,10

Num: $1{,}10 \cdot 2^{6} = 1100000 = 96 = -96$

**Esercizio 1 (3 punti):**
Dimostrare usando le tabelle di verità se la seguente uguaglianza è verificata

X XOR (Y NAND Z) = (X NAND Y) XOR (X NOR Z)

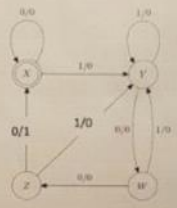
Scrivete la tabella di verità disponendo in ordine crescente i numeri X Y Z, come esemplificato nel modulo risposte.

$F_1 \neq F_2$

| X | Y | Z | Y NAND Z | $F_1$ | X NAND Y | X NOR Z | $F_2$ |
|---|---|---|---|---|---|---|---|
| 0 | 0 | 0 | 1 | 1 | 1 | 1 | 0 |
| 0 | 0 | 1 | 1 | 1 | 1 | 0 | 1 |
| 0 | 1 | 0 | 1 | 1 | 1 | 1 | 0 |
| 0 | 1 | 1 | 1 | 1 | 1 | 0 | 1 |
| 1 | 0 | 0 | 1 | 0 | 1 | 0 | 1 |
| 1 | 0 | 1 | 1 | 0 | 1 | 0 | 1 |
| 1 | 1 | 0 | 1 | 0 | 0 | 0 | 0 |
| 1 | 1 | 1 | 0 | 1 | 0 | 0 | 0 |

**Esercizio 2 (3 punti):**
Dato il grafo di transizione dell'automa a stati finiti che riconosce le sequenze 1000

1) si completi la tabella di transizione riportata nel foglio delle risposte;
2) si codifichino i valori dello stato X, Y, Z, W (quanti bit sono necessari?);
3) si scrivano le espressioni booleane per Out e Prossimo Stato in qualunque forma si desideri
4) Cosa si dovrebbe modificare e come per far riconoscere all'automa la sequenza 1001?



| A | B | M | A+B | OUT |
|---|---|---|---|---|
| 0 | 0 | 0 | 0 0 | 0 |
| 0 | 1 | 1 | 0 1 | 0 |
| 1 | 0 | 0 | 0 1 | 0 |
| 1 | 1 | 1 | 1 0 | 0 |
| 1 | 0 | 0 | 0 0 | 1 |
| 1 | 0 | 1 | 0 1 | 0 |
| 1 | 1 | 0 | 1 0 | 0 |
| 1 | 1 | 1 | 0 1 | 0 |

X = 00
Y = 01
Z = 10
W = 11
} 2 bit

**Esercizio 3 (3 punti):**
Sia A il numero relativo la cui rappresentazione binaria in complemento a 2 vale 1001 e B il numero relativo la cui rappresentazione binaria in modulo e segno vale 1101.

a) Qual è il minore tra A e B?

b) Quanto vale, in complemento a 2, la somma A+A su 4 bit?

Motivare la risposta con spiegazioni, passaggi e calcoli. Il solo risultato finale non sarà considerato sufficiente in fase di valutazione.

1001 → 0111 → -7

1101 → 0011 → -3

B è il valore minore tra A

1001+
1001
0010 → 0010 → 2 → si tratta di un overflow

**Esercizio 1 (3 punti).**
Descrivere la rappresentazione dei numeri relativi in complemento a due su n bit, specificando:

a) come si ottiene la codifica dei numeri positivi e negativi;
b) qual è l'intervallo di rappresentazione;
c) quali sono i principali vantaggi di questa rappresentazione dei numeri relativi.

Per ottenere la codifica in base 2, se un numero è positivo basta ricopiarlo così com'è.
Se un numero è negativo, bisogna complementarlo a 1 (negarlo) e sommargli 1.
L'intervallo di rappresentazione su n bit è [-2^(n-1); +2^(n-1)-1]
Il principale vantaggio di questa rappresentazione è che si ha una sola rappresentazione dello zero.

**Esercizio 2 (3 punti).**
Si consideri una rappresentazione binaria di numeri razionali (con segno) in virgola mobile su 8 bit, organizzata con 1 bit di segno, 4 bit di esponente e 3 bit di mantissa (nell'ordine). L' esponente è codificato in eccesso 8. La mantissa $m$ è normalizzata in modo che: $0{,}5 \leq m < 1$ (il bit più significativo del campo della mantissa corrisponde alla potenza $2^{-1}$). La configurazione 000 della mantissa è riservata per la codifica del numero zero.

**a)** Qual è il massimo numero rappresentabile esattamente?
**b)** Qual è il minimo numero positivo (quindi, maggiore di zero) rappresentabile esattamente?
**c)** Quante diverse rappresentazioni ci sono per il numero 0?

$+1{,}0 \cdot 2^{7}$ ⇒ Massimo numero rappresentabile

$+1{,}111 \cdot 2^{-8}$ ⇒ Minimo rappresentabile

2 : +0 -0

**Esercizio 4 (3 punti).**
Si consideri il seguente diagramma di stato relativo ad un automa a stati finiti. Si ricavi la tabella di stato corrispondente e si disegni il circuito sequenziale che la realizza, utilizzando flip-flop di tipo D.


| A | INPUT | A |
|---|---|---|
| 0 | 1 | |
| 0 | 1 | |
| 1 | 0 | |
| 1 | 1 | |

**Esercizio 6 (3 punti).**
Dire, motivando la risposta, quali delle seguenti sequenze di 36 bit sono microistruzioni Mic-1 che:

- accedono alla memoria , ma non all'area dei metodi,
- prevedono il controllo dei bit di stato dell'ALU,
- operano scritture multiple sui registri.


a) 011100101 100 10010111 011000101 010 0010
b) 000000000 100 00000000 011111111 001 0011
c) 010000000 001 00111111 000000000 100 1000

Accedono alla memoria ma non all'area dei metodi:
A,C

Predevono il controllo dei bit di stato dell'ALU:
C

Operano scritture multiple su registri:
A,B

**Esercizio 7 (3 punti)**
Quali delle seguenti affermazioni relative all'architettura Mic-2 sono vere:

a) non è necessario che almeno uno degli operandi dell'ALU debba provenire dal registro H;
b) il microprogramma del Mic-2 è identico al microprogramma del Mc-1;
c) l'IFU recupera dall'area dei metodi 4 byte alla volta;
d) l'offset di 2 byte presente nell'istruzione IJVM "*GOTO offset*" viene prelevato direttamente dal registro MBR2;
e) utilizza la tecnica del pipelining.

**Esercizio 9 (4 punti)**
Utilizzando il linguaggio assemblativo IJVM, scrivere un metodo `prova` che prenda in input tre parametri formali i, j e k (numeri interi) e che restituisca il valore (i+k+j) se j =k ; i+k altrimenti. Scrivere inoltre il segmento di codice che traduce la chiamata di tale metodo con parametri attuali 2,3 e 4.

```
.constant
Objref 0xCAFE
.end-constant
.main
.var
.end-var
LDC_W Objref
BIPUSH 2
BIPUSH 3
BIPUSH 4
INVOKEVIRTUAL PROVA

.end-main
```

```
.method PROVA(i,j,k)
.var
.end-var
ILOAD j
ILOAD k
IF_ICMPEQ L1:
ILOAD i
ILOAD k
ISUB
IRETURN
L1:
ILOAD i
ILOAD k
ILOAD j
IADD
IRETURN
.end-method
```

**Esercizio 10 (4 punti)**
La sequenza di istruzioni di livello Mic-1 sotto riportata realizza una nuova istruzione `bish8pu x` (x è un offset a 8 bit in binario puro) . Qual è il significato di tale istruzione?

```
bish8pu1    MAR=SP
bish8pu2    H=TOS << 8
bish8pu3    TOS=MDR=MBRU OR H;wr
bish8pu4    PC=PC+1;fetch
bish8pu5    goto Main1
```

Questo metodo combina due byte in modo che i primi 8 bit presi da TOS siano i più significativi, mentre gli altri 8 presi da MBRU siano messi nelle 8 posizioni meno significative. Inoltre aggiorna il TOS e la memoria con questo valore. C'è bisogno di questo metodo perché l'operazione di fetch può prendere in input solo 8 bit alla volta.