Sistemi di Elaborazione/Architettura degli Elaboratori
6 Giugno 2001

ASSEMBLER

Una subroutine Assembler (Motorola 68000) accetta in ingresso una matrice *M* di dimensione *nxm* di numeri interi con segno formato byte mediante il registro A0, i numeri interi *m* e *n* nelle word meno significative di D0 e D1, e gli estremi di un intervallo numerico [*a*,*b*] nei byte meno significativi di D2 e D3. Detta subroutine restituisce il valore 1 long in D0 se le *m* somme degli elementi di colonna di *M* sono tutte comprese nell'intervallo [*a*,*b*], altrimenti il valore 0 long.

Ad esempio per la matrice $\begin{pmatrix} 1 & 2 & 3 & 4 & 5 \\ 9 & 7 & 5 & 3 & 1 \\ 1 & 3 & 5 & 5 & 5 \end{pmatrix}$ e l'intervallo [11,13], la routine ritorna 1 in quanto le somme 11,12,13,12 e 11 sono comprese nell'intervallo [11,13].

SOLUZIONE

```
            lea       matrice,a0        esempio di chiamata alla subroutine
            move.w    #5,d0             nel caso della matrice 3x5 dichiarata
            move.w    #3,d1             al termine e intervallo [11,13]
            move.b    #11,d2
            move.b    #13,d3
            bsr       check
            break

check       movem.l   d1 / d4-d7,-(sp)  salvataggio registri
            move.w    d0,d4             d4 manterrà il valore m
            subq.w    #1,d0             d0 = contatore ciclo DBcc per scandire le colonne
            subq.w    #1,d1             d1 = valore iniziale del contatore del ciclo per scandire le
                                        righe
row         move.w    d1,d5             d5 = contatore del ciclo DBcc per scandire le righe
            move.w    d0,d6             inizializzo (l'indice) d6 con d0
            clr.b     d7                inizializzo un accumulatore per la somma di colonna
columsum    add.b     (a0,d6.w),d7      utilizzo l'indice d6 per accumulare in d7 l'elemento
                                        corrente della matrice
            add.w     d4,d6             sposto l'indice d6 sul prossimo elemento da sommare
            dbra      d5,columsum       continuo la colonna in esame
            cmp.b     d2,d7             controllo se la somma è inferiore ad a
            blt.s     fail              nel caso la subroutine esce con fallimento
            cmp.b     d3,d7             controllo se la somma è superiore a b
            bgt.s     fail              nel caso la subroutine esce con fallimento
            dbra      d0,row            passo alla prossima riga
success     moveq.l   #1,d0             assegno l'output per indicare il successo
            bra.s     exit              e vado a terminare
fail        moveq.l   #0,d0             assegno l'output per indicare il fallimento
exit        movem.l   (sp)+,d1 / d4-d7  ripsistino i registri
            rts


matrice     dc.b      1,2,3,4,5
            dc.b      9,7,5,3,1
            dc.b      1,2,3,4,5
```

Sistemi di Elaborazione dell'informazione/Architettura degli Elaboratori
29 Giugno 2001

Esercizio 1.
a) Si converta in binario puro il seguente numero razionale in modo da garantire un errore di conversione inferiore a 0.01 (decimale) :

30.1328125 = 11110.0010001

b) Si determinino le rappresentazioni con il minor numero di bit dei numeri 549 e -549 in modulo e segno, complemento a 1 e complemento a 2 :

549 = 01000100101 (M&S, compl. a 1, compl. a 2) : 11 bits

-549 = 11000100101 (M&S) = 10111011010 (compl. a 1) = 10111011011 (compl. a 2) : 11 bits.

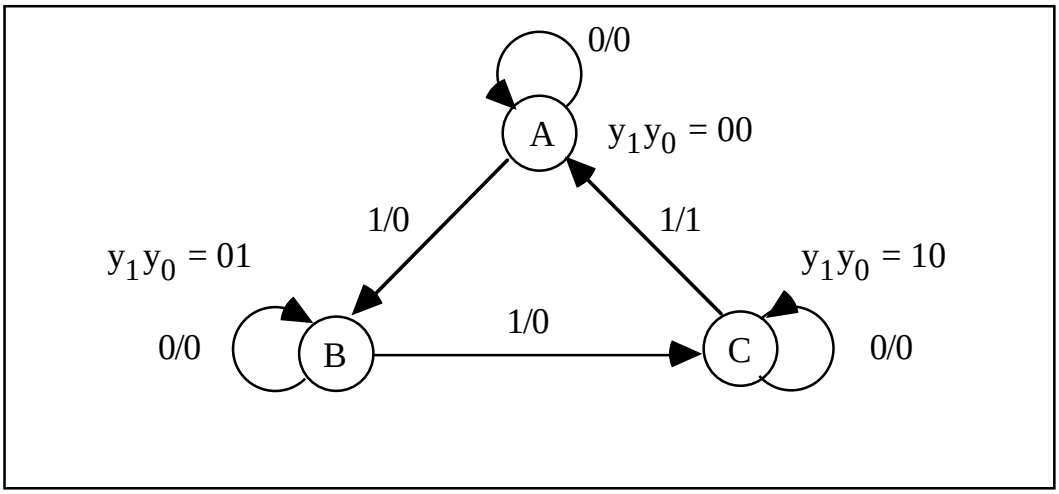
Esercizio 2.
Si progetti un dispositivo sequenziale sincrono che riceve in ingresso una linea seriale x sulla quale viene trasmessa una sequenza di bit. Il dispositivo dispone di un'uscita u seriale che normalmente vale 0 e vale 1 ogni tre occorrenze del valore 1 in ingresso. E. g. se l'ingresso x riceve la sequenza di valori

x = 001000101011010... ,

l'uscita corrispondente sarà

u = 000000001000010... .

a) Si disegni il diagramma degli stati del dispositivo sul retro di questo foglio.
b) Si sintetizzino le espressioni booleane di stato futuro e di uscita utilizzando le seguenti mappe di Karnaugh supponendo:
1) di utilizzare dei flip-flop di tipo JK ;
2) che i 4 valori delle variabili y_1 y_0 : 00, 01, 10, 11 codifichino i 4 (possibili) stati A, B, C, D del dispositivo.



y_1y_0 \ x	0	1
00		
01		1
11	–	–
10	–	–

J_1

y_1y_0 \ x	0	1
00	–	–
01	–	–
11	–	–
10		1

K_1

y_1y_0 \ x	0	1
00		1
01	–	–
11	–	–
10		

J_0

y_1y_0 \ x	0	1
00		
01		
11	–	–
10		1

u

y_1y_0 \ x	0	1
00	–	–
01		1
11	–	–
10	–	–

K_0

$u = y_1x$, $J_1 = y_0x$, $J_0 = \overline{y_1}\,x$, $K_0 = K_1 = x$.

Lo stato D è inutilizzato e pertanto nelle 5 mappe di Karnaugh le caselle corrispondenti ai valori di $y_1\,y_0$: 11 sono indeterminate.

Si risponda alle seguenti domande, su questo foglio, contrassegnando le opportune caselle **[]**:

1) Il numero di transistor contenuti nei più recenti microprocessori è nell'ordine di:
[] decine di migliaia **[]** un paio di milioni **[]** decine di milioni **[]** 200-300 milioni **[]** 1-2 miliardi

2) Il decode nei processori RISC è semplificato da questi aspetti dell'instruction set:
[] campo address nell'ordine del paio di bit **[]** numero di istruzioni nell'ordine della decina
[] formato fisso **[]** formato variabile **[]** byte ordering semplificato **[]** nessuno di questi aspetti

3) Considerando il flip-flop di tipo T:

se il comando di ingresso è 0, lo stato del flip-flop sarà _____
se il comando di ingresso è 1, lo stato del flip-flop sarà _____

4) Le tecniche più efficaci per risolvere i control hazard in una pipeline sono basate su:
[] una memoria cache dedicata **[]** una ALU dedicata **[]** una control unit microprogrammata
[] snoopy cache controller **[]** separazione della cache di primo livello in cache istruzioni e cache dati

5) Il *programmed I/O* è una tecnica:
[] tipica di microprocessori a basso costo
[] tipica dell'uso domestico dei PC
[] tipica del calcolo scientifico
[] che spreca inutilmente tempo di CPU
[] che ottimizza l'uso del tempo di CPU
[] che non influisce sull'uso del tempo CPU

6) Un utente cambia la configurazione del suo PC, spostando la scheda audio dal bus ISA al PCI. La gestione delle informazioni audio diventa:
più affidabile? **[]** si **[]** no **[]** solo se la scheda supporta la connessione PCI
più efficiente? **[]** si **[]** no **[]** solo se la scheda supporta la connessione PCI

7) La prima passata di un programma assemblatore:
[] costruisce la symbol table
[] stabilisce associazioni fra costanti ed i loro valori
[] stabilisce associazioni fra label e loro posizioni nel codice
[] nessuna delle tre risposte precedenti

8) Quale di queste tecniche di mapping è più efficiente per la gestione della memoria cache?
[] LRU **[]** direct **[]** 2-way set-associative **[]** 8-way set-associative

9) Data un'architettura a stack su cui calcolare l'espressione (L+M)/Z, si indichi (mediante un numero all'interno delle caselle) l'ordine in cui vanno eseguite le seguenti istruzioni (1=la prima, ..., 5=l'ultima):

[ ] PUSH L **[ ]** PUSH M **[ ]** ADD **[ ]** PUSH Z **[ ]** DIV

Si indichino il proprio cognome e nome nel sottostante riquadro:

COGNOME:_____ NOME:_____

ASSEMBLER

Una subroutine Assembler (Motorola 68000) accetta in ingresso due vettori *V0* e *V1* di numeri interi distinti ordinati in senso crescente e produce in uscita un vettore *V2* contenente tutti e soli gli elementi presenti sia in *V1* che in *V2*, ordinati in senso crescente. I registri A0, A1 e A2 contengono gli indirizzi base dei vettori *V0*, *V1* e *V2* mentre i registri D0 e D1 contengono nella loro <u>word</u> meno significativa, rispettivamente, la dimensione di *V0* e *V1*. La subroutine restituisce nel registro D2 la dimensione del vettore risultante *V2*. Si supponga che tutti i vettori contengano interi <u>con segno</u> in formato <u>byte.</u>

Il listato (che comprende un esempio di chiamata) è già delineato qui di seguito, con opportuni commenti. Si completino direttamente su questo foglio **<u>tutte</u> e <u>sole</u>** le parti indicate da sottolineatura (N.B.: tutti i punti sono già esplicitamente forniti).

```
          lea       vett1,a0                esempio di chiamata alla subroutine
          move.w    #7,d0                   nel caso dei vettori vett1, di dimensione 7;
          lea       vett2,a1                vett2, di dimensione 10, e vett
          move.w    #10,d1
          lea       v,a2
          bsr       intersect
          break

intersect movem.l d0-d1/d3/a0-a2,-(sp)      salvataggio registri
          subq.w    #1,d0                   d0=contatore ciclo per scandire V0
          subq.w    #1,d1                   d1=contatore ciclo per scandire V1
          clr.w     d2                      inizializzo l'output d2
loop      move.b    (a0)+,d3                sposto in d3 l'elemento di V0 in esame…
test      cmp.b     (a1)+,d3                …e lo confronto con l'elemento x di V1 in esame
          dble      d1,test                 se d3 è maggiore di x, continuo a confrontarlo con
                                            gli elementi successivi di V1
          bgt.s     finish                  se ho confrontato tutti gli elementi di V1 termino
          blt.s     lessthan                se d3 è minore di x, vado a correggere a1
          move.b    d3,(a2)+                inserisco d3 in V2 (dato che d3 è uguale a x)
          addq.w    #1,d2                   aggiorno l'output d2
          dbra      d1,continue             se ho ancora elementi da confrontare in V1 continuo
          bra.s     finish                  altrimenti termino
lessthan  suba.w    #1,a1                   correggo a1
continue  dbra      d0,loop                 se ho ancora elementi da confrontare in V0 proseguo

finish    movem.l (sp)+,d0-d1/d3/a0-a2      ripristino i registri
          rts                               e termino

vett1     dc.b      1,3,5,6,7,10,13
vett2     dc.b      2,3,4,5,7,8,9,10,11,12
vett      ds.b      10
```

Sistemi di Elaborazione dell'informazione/Architettura degli Elaboratori
11 Luglio 2001

Esercizio 1.
a) Si converta in binario puro il seguente numero razionale in modo da garantire un errore di conversione inferiore a 0.01 (decimale) :

3.14159 = 11.0010010

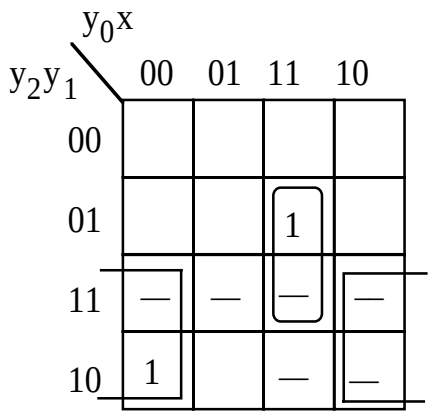
b) Siano A= 11101100 e B = 10010000 le rappresentazioni in complemento a 2 a 8 bit di due numeri interi. Si effettuino le operazioni A+B e A-B , indicando se si è verificato overflow.

A= 11101100 = -20 B = 10010000 = -112

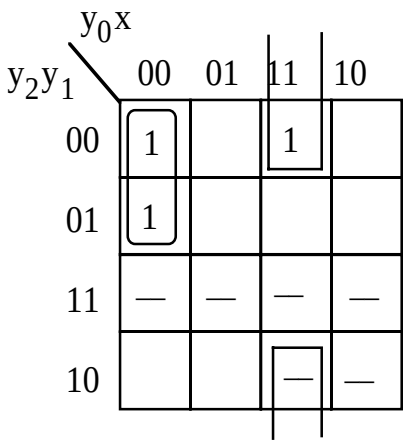
A+B = 01111100 Overflow = Si

A-B = 01011100 = 92 Overflow = No

Esercizio 2.
Si progetti un dispositivo sequenziale sincrono che riceve in ingresso una linea seriale x sulla quale viene trasmessa una sequenza di bit a multipli di 3. Il dispositivo legge ogni tripla e genera un'uscita u che normalmente vale 0 e vale 1 in corrispondenza del 3º bit di ogni tripla che ha un numero pari di occorrenze della cifra binaria 1 .
a) Si disegni il diagramma degli stati del dispositivo sul retro di questo foglio.
b) Si sintetizzino le espressioni booleane di stato futuro e di uscita utilizzando le seguenti mappe di Karnaugh supponendo:
1) di utilizzare dei flip-flop di tipo D ;
2) che gli 8 valori delle variabili $y_2\ y_1\ y_0$: 000, 001, 010, 011, 100, 101, 110, 111 codifichino gli 8 (possibili) stati A, B, C, D, E, F, G, H del dispositivo.



$U = y_1y_0x + y_2\overline{x}$



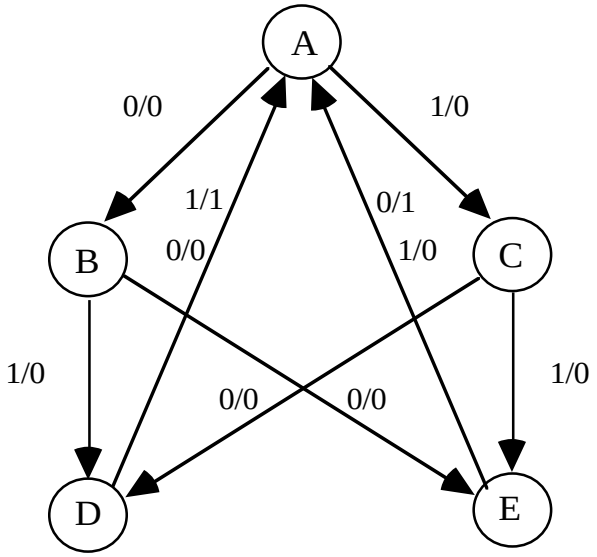
$D_0 = x\overline{y_1}y_0 + \overline{y_2}\,\overline{y_0}\,\overline{x}$

y_2y_1 \ y_0x	00	01	11	10
00		1	1	
01	1			
11	—	—	—	—
10			—	—

$$D_1 = \overline{y_2}\,\overline{y_1}\,x + y_1\,\overline{y_0}\,\overline{x}$$

y_2y_1 \ y_0x	00	01	11	10
00				1
01		1		
11	—	—	—	—
10			—	—

$$D_2 = \overline{y_0}\,y_1\,x + \overline{y_1}\,y_0\,\overline{x}$$



Gli stati F, G, H sono inutilizzati e pertanto nelle 4 mappe di Karnaugh le caselle corrispondenti ai valori di $y_2\,y_1\,y_0$: 101, 110, 111 sono indeterminate.

Si risponda alle seguenti domande, su questo foglio, contrassegnando le opportune caselle **[]**:

1) Il numero di transistor contenuti nei calcolatori di prima generazione era nell'ordine di:
[] migliaia **[]** decine di migliaia **[]** decine di milioni **[]** nessuna delle tre

2) Il decode nei processori CISC puo' essere reso complesso da questi aspetti dell'instruction set:
[] numerosi utilizzi possibili del campo address **[]** numerosi cicli di clock per eseguire un'istruzione
[] formato fisso **[]** formato variabile **[]** byte ordering **[]** nessuno di questi aspetti

3) Considerando il flip-flop di tipo SR:

se il comando di ingresso e' 00, lo stato del flip-flop sara' _____
se il comando di ingresso e' 11, lo stato del flip-flop sara' _____

4) Le tecniche piu' efficaci per risolvere i data hazard in una pipeline sono basate su:
[] memoria cache per TLB **[]** branch prediction **[]** control unit microprogrammata
[] snoopy cache controller **[]** invio anticipato di operandi fra segmenti della pipeline

5) Il programmed I/O e' una tecnica:
[] tipica di microprocessori a basso costo
[] tipica dell'uso domestico dei PC
[] tipica del calcolo scientifico
[] che spreca inutilmente tempo di CPU
[] che ottimizza l'uso del tempo di CPU
[] che non influisce sull'uso del tempo CPU

6) Un utente cambia la configurazione del suo PC, spostando l'I/O scheda video da porta AGP a bus PCI. La gestione delle informazioni video diventa:
piu' affidabile?**[]** si **[]** no **[]** solo se la scheda supporta la connessione PCI
piu' efficiente?**[]** si **[]** no **[]** solo se la scheda supporta la connessione PCI

7) La seconda passata di un programma assemblatore:
[] costruisce la symbol table
[] memorizza associazioni fra label simboliche e loro posizione nel codice
[] risolve le associazioni fra istruzioni di salto e label simboliche
[] genera il codice in linguaggio macchina
[] nessuna delle quattro risposte precedenti

8) Quale tecnica di rimpiazzamento da' i migliori risultati nella gestione della memoria cache?
[] LRU **[]** direct **[]** 4-way set-associative **[]** FIFO **[]** LIFO

9) Se sul processore 68000 si verifica l'eccezione 5, l'indirizzo del gestore d'eccezione viene reperito:
[] all'indirizzo di RAM 0 **[]** sullo stack **[]** sul bus **[]** all'indirizzo di RAM 5
[] all'indirizzo di RAM 10 **[]** all'indirizzo di RAM 20 **[]** nessuna delle 6 risposte

Si indichino il proprio cognome e nome nel sottostante riquadro:

COGNOME:_____ NOME:_____

Sistemi di Elaborazione/Architettura degli Elaboratori
11 Luglio 2001

ASSEMBLER

Una matrice quadrata *M* di ordine *n* si dice CONVERGENTE con raggio *r* se il modulo della somma degli elementi di ogni riga, escluso quello sulla diagonale, è inferiore a *r*.
Una subroutine Assembler (Motorola 68000) accetta in ingresso una matrice *M* di numeri interi con segno a 8 bit (mediante il registro A0), un numero intero con segno *n* nella word meno significativa del registro D0 ed un numero intero con segno *r* nel byte meno significativo del registro D1. La subroutine restituisce il valore -1 in D3 se *M* è convergente con raggio *r*, altrimenti restituisce un valore non negativo in D3.

Il listato (che comprende un esempio di chiamata) è già delineato qui di seguito, con opportuni commenti. Si completino direttamente su questo foglio **<u>tutte</u> e <u>sole</u>** le parti indicate da sottolineatura (N.B.: tutti i punti e virgole sono già esplicitamente forniti).

```
        lea       Mat1,A0                esempio di chiamata della subroutine nel....
        move.w    #3,D0                  caso della matrice dichiarata al termine del listato
        move.b    #7,D1                  ...e con n = 3 e r = 7
        bsr       conv
        break

conv:   movem.l   A0/D0/D2/D4, -(SP)     salvataggio registri
        subq.w    #1,D0                  prepara D0 come contatore
        move.w    D0,D3                  D3=contatore per il ciclo che scandirà le righe
nextrow move.w    D0,D2                  D2=contatore ciclo che scandirà gli elementi di una riga
        clr.b     D4                     D4=contatore somma di una riga
loop:   add.b     (A0)+,D4               aggiungi alla somma l'elemento in esame
        dbra      D2, loop               passiamo a sommare il prossimo elemento
        sub.w     d3,d2                  D2=distanza dell'elemento sulla diagonale dal primo…
                                         …elemento della riga seguente (-1-D3)
        sub.b     (A0,D2.w), D4          sottrai l'elemento sulla diagonale alla somma totale
        tst.b     D4                     confronta la somma della riga con 0
        bge.s     magg                   se è maggiore o uguale a 0 confrontala con r ...
        neg.b     D4                     altrimenti complementala
magg    cmp.b     D4,D1                  confronta il modulo della somma con r
        dble      D3, nextrow            se è il caso si va ad esaminare la riga seguente
noneg:  movem.l   (SP)+,A0/D0/D2/D4      ripristino registri
        rts

Mat1    dc.b      1,0,-4
        dc.b      0,3,5
        dc.b      2,4,0
```

Sistemi di Elaborazione/Architettura degli Elaboratori
5 Settembre 2001

ASSEMBLER

Un insieme di *m* stringhe di caratteri (a lunghezza variabile ma limitata superiormente dal numero *n-1*) viene memorizzato in una matrice *M* di ordine *m*×*n*. Una stringa è una sequenza di caratteri del codice ASCII (8 bit) terminata dal valore intero 0. Una subroutine Assembler (Motorola 68000) accetta in ingresso l'indirizzo base di tale matrice *M* mediante il registro A0, i numeri interi con segno *m* e *n* nelle word meno significative dei registri D0 e D1 rispettivamente. La subroutine restituisce il valore 1 (long) in D0 se le stringhe di *M* sono ordinate lessicograficamente, 0 (long) altrimenti. Una stringa è lessicograficamente minore di un'altra se, scandendo i caratteri da sinistra a destra, dopo una eventuale sequenza di caratteri uguali c'è un carattere della prima alfabeticamente minore del corrispettivo dell'altra, oppure la prima è terminata.

```
        lea       matrice,a0          esempio di chiamata alla subroutine
        move.w    #5,d0               nel caso dell'insieme dichiarato in calce
        move.w    #6,d1
        bsr.s     check
        break

check   movem.l   a0/a1/d1/d2,-(sp)   salvataggio registri
        move.l    a0,a1               in coppia con la prossima istruzione…
        adda.w    d1,a1               …A1 punta alla seconda stringa
        subq.w    #2,d0               imposto D0 come contatore di ciclo esterno
        subq.w    #1,d1               predispongo D1 come limite del ciclo interno
loop    move.w    d1,d2               predispongo D2 come contatore del ciclo interno
cmpstrs cmpm.b    (a0)+,(a1)+         confronto le due stringhe
        dbne      d2,cmpstrs          fino a che la prima è <= lessicograficamente
        bcs.s     fail                se la seconda è > fallisco
        beq.s     loopend             se è = continuo
        adda.w    d2,a0               sposto A0 sulla prossima stringa
        adda.w    d2,a1               sposto A1 sulla prossima stringa
loopend dbra      d0,loop             verifico le prossime stringhe
        moveq.l   #1,d0               imposto il successo
        bra       exit                e vado a terminare
fail    clr.l     d0                  imposto il fallimento
exit    movem.l   (sp)+,a0/a1/d1/d2   e termino
        rts

matrice dc.b      'm',000,001,002,003,004
        dc.b      'm','a',000,001,002,003
        dc.b      'm','a','r',000,001,002
        dc.b      'm','a','r','c',000,001
        dc.b      'm','a','r','c','o',000
```

Sistemi di Elaborazione/Architettura degli Elaboratori
20 Settembre 2001

ASSEMBLER

Una subroutine Assembler (Motorola 68000) calcola media (intera) e varianza (intera) di un vettore *v* di *n* numeri interi senza segno a 16 bit. Media μ e varianza ν vengono calcolate secondo le (note) formule

$$\mu = \frac{\sum_{i=1}^{n} v_i}{n} \quad \text{e} \quad \nu = \frac{\sum_{i=1}^{n} v_i^2}{n} - \mu^2 .$$

Si noti che le operazioni di somma, moltiplicazione e divisione utilizzate nelle formule sono da assumersi in aritmetica intera senza segno.

La subroutine accetta in ingresso l'indirizzo base *v* mediante il registro A0 e il numero *n* nella word meno significativa del registro D0, mentre restituisce nelle word meno significative dei registri D0 e D1 rispettivamente μ e ν.

```
        lea       vett,a0          esempio di chiamata alla subroutine
        move.w    #5,d0            nel caso dell'insieme dichiarato in calce
        bsr.s     varianza
        break

varianza movem.l  a0/d2-d4,-(sp)   salvataggio registri
        move.w    d0,d2            D2 manterrà il valore n
        move.w    d0,d3            in coppia con la prossima istruzione…
        subq.w    #1,d3            …predispongo D3 come contatore ciclo
        clr.l     d0               inizializza D0 come accumulatore per la somma v[i]
        clr.l     d1               inizializza D1 come accumulatore somma v[i]^2
loop    clr.l     d4               azzero D4
        move.w    (a0)+,d4         carico v[i] in D4
        add.l     d4,d0            aggiorno D0
        mulu.w    d4,d4            calcolo v[i]^2
        add.l     d4,d1            aggiorno D1
        dbra      d3,loop          continuo col prossimo elemento del vettore
        divu.w    d2,d0            D0 = media
        divu.w    d2,d1            D1 = (somma v[i]^2)/n
        move.w    d0,d3            in coppia con la prossima istruzione…
        mulu.w    d0,d3            …assegna a D3 il quadrato della media
        sub.w     d3,d1            D1 = varianza

exit    movem.l   (sp)+,a0/d2-d4   ripristino registri
        rts

vett    dc.w      50,51,52,53,54
        end
```

Sistemi di Elaborazione dell'informazione/Architettura degli Elaboratori
20 Settembre 2001

Esercizio 1.
a) Si converta in binario puro il seguente numero razionale in modo da garantire un errore di conversione inferiore a 0.01 (decimale) :

41.1875 = 101001.0011

b) Si determinino le rappresentazioni con il minor numero di bit dei numeri 1024 e -1024 in modulo e segno, complemento a 1 e complemento a 2 :

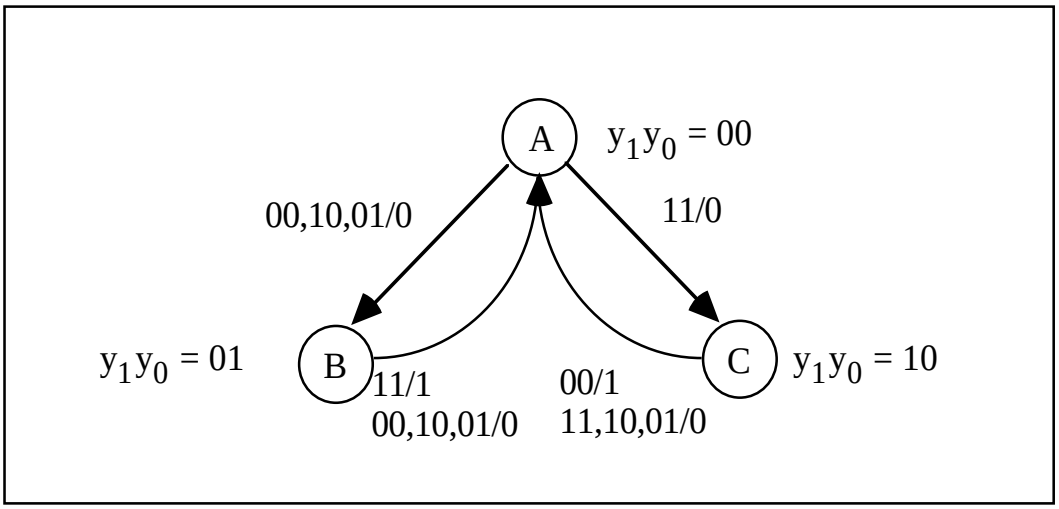
1024 = 010000000000 (M&S, compl. a 1, compl. a 2) : 12 bits

-1024 = 110000000000 (M&S) : 12 bits = 101111111111 (compl. a 1) : 12 bits = 10000000000 (compl. a 2 : 11 bits.

Esercizio 2.
Si progetti un dispositivo sequenziale sincrono che riceve in ingresso due linee seriali a, b sulle quali vengono trasmesse sequenze di bit a multipli di due. Il dispositivo interpreta ogni coppia come un numero in complemento a 2 (il 1º bit è il meno significativo) e genera un'uscita che normalmente vale 0 , mentre vale 1 in corrispondenza del secondo bit delle due coppie se la somma delle due coppie genera overflow. E. g.

a = 01 10 …
b = 11 11 …
u = 01 00 …

a) Si disegni il diagramma degli stati del dispositivo sul retro di questo foglio.
b) Si sintetizzino le espressioni booleane di stato futuro e di uscita utilizzando le seguenti mappe di Karnaugh supponendo:
1) di utilizzare dei flip-flop di tipo JK ;
2) che i 4 valori delle variabili $y_1\ y_0$: 00, 01, 10, 11 codifichino i 4 (possibili) stati A, B, C, D del dispositivo.



Lo stato D è inutilizzato e pertanto nelle 5 mappe di Karnaugh le caselle corrispondenti ai valori di $y_1\ y_0$: 11 sono indeterminate.

ab \ y_1y_0	00	01	11	10
00	1	–	–	
01	1	–	–	
11		–	–	
10	1	–	–	

$$J_0 = \overline{y_1}\,\overline{b} + \overline{y_1}\,\overline{a}$$

ab \ y_1y_0	00	01	11	10
00	–	1	–	–
01	–	1	–	–
11	–	1	–	–
10	–	1	–	–

$$K_0 = 1$$

a b \ y_1y_0	00	01	11	10
00			—	—
01			—	—
11	1		—	—
10			—	—

$J_1 = \overline{y_0}\,ab$

a b \ y_1y_0	00	01	11	10
00	—	—	—	1
01	—	—	—	1
11	—	—	—	1
10	—	—	—	1

$K_1 = 1$

a b \ y_1y_0	00	01	11	10
00			—	1
01			—	
11		1	—	
10			—	

$u = y_1\,\overline{a}\,\overline{b} + y_0\,ab$