Prova 22/05/2025

Rocco Lo Russo
roc.lorusso@studenti.unina.it

Università di Napoli Federico II - DIETI — 16/07/2025

Introduzione

In questo documento verrà sviluppata la prova intercorso del 22/05/2025, esponendo analiticamente i seguenti punti: **mappa della memoria**, **pseudocodice** e implementazione in **ASIM**.

1 Traccia

Un sistema è composto da 3 unità, A, B e C, tra loro collegate mediante due periferiche parallele che interconnettono A con B e A con C rispettivamente. I messaggi hanno un primo carattere identificativo che può essere pari a 0 a un valore diverso da 0. Il sistema opera in due fasi successive come descritto di seguito:

- Fase 1: A riceve K messaggi di N caratteri da B e da C in modo alternato. In ordine non prefissato, quindi si parte da B o da C, e **non ci sono sovrapposizioni tra i messaggi ricevuti da B o da C**;
- Fase 2: Al termine della fase 1, il nodo A continua nella stessa modalità alternata e termina la ricezione dei messaggi se due messaggi ricevuti (dalle due diverse periferiche) hanno il carattere identificativo del messaggio pari a 0.

1.1 Analisi della traccia

Dalla traccia emerge che la comunicazione debba essere gestita in modo tale da garantire la *non sovrapposizione* dei messaggi ricevuti da B e da C. Questo implica che verrà utilizzato un unico buffer di ricezione in cui verranno conservati i messaggi ricevuti, e che una periferica non potrà mandare un nuovo carattere se l'altra non avrà finito la trasmissione di un intero messaggio. La ricezione dei messaggi in modo alternato pone il vincolo, nella codifica della ISR, di considerare casuale la provenienza del primo messaggio. Ci sono due modi di interpretare questo passo della traccia:

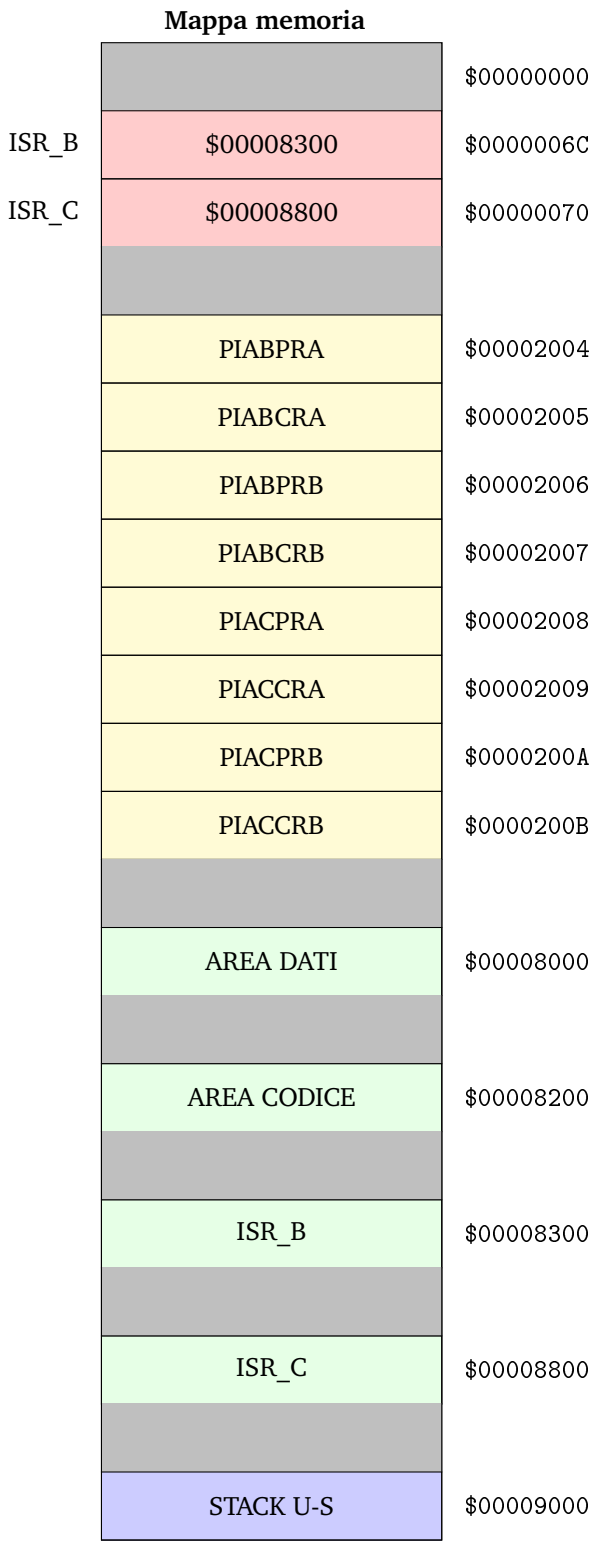
- Il primo messaggio può arrivare da qualsiasi periferica, e una volta arrivato questo stabilisce l'ordine di arrivo di tutti gli altri K-1 messaggi;
- I messaggi vengono ricevuti in modo alternato ma a coppie, ovvero **per ogni coppia** il primo messaggio può provenire da qualsiasi periferica, ma il secondo deve provenire necessariamente dall'altra.

La soluzione presentata più avanti si basa sulla seconda interpretazione. Sotto questi vincoli, i conflitti da gestire sono:

- Accesso in mutua esclusione in scrittura alla risorsa rappresentata dal nodo A, in modo da garantire la non sovrapposizione dei messaggi;
- Regolare l'accesso alla risorsa in modo che sia alternato, curando i casi in cui le ISR pongano una periferica in stato di attesa.

2 Mappa della memoria

In questa sezione verrà presentata una mappa della memoria del nodo A in accordo a quanto specificato nel file .cfg utilizzato nella simulazione e in accordo alla memoria caricata (*file rom.mem*).



3 Implementazione

In questa sezione verrà presentato il codice assembly per Motorola68000 e lo pseudocodice usato come riferimento per l'implementazione.

3.1 Variabili

Descrizione delle variabili utilizzate:

`fine`	Intero che può assumere i valori 0 (il nodo A è in ricezione) o 1 (il nodo A ha terminato la ricezione).
`lock`	Intero che può assumere i valori 0 o 1, viene testato dall'istruzione atomica TAS per garantire l'accesso mutualmente esclusivo alla sezione critica.
`possesso`	Intero che può assumere i valori -1 (`is_free`), 0 (`is_reading_b`) o 1 (`is_reading_c`).
`buff`	Puntatore alla prima locazione di un vettore di dimensione K*N di caratteri; serve per accedere alla memoria del nodo A.
`curr`	Intero da 0 a N che tiene conto dei caratteri ricevuti.
`tot`	Intero da 0 a K*N per l'accesso indirizzato al vettore dei caratteri.
`msg`	Intero da 0 a K che conta i messaggi ricevuti.
`end_b,(end_c)`	Intero: 0 se il messaggio da b (c) non è terminato, 1 altrimenti.
`fase2`	Intero: 0 (fase 1), 1 (fase 2).
`cond_b,(cond_c)`	Intero: 0 se il primo carattere ricevuto da b (c) non è 0, 1 altrimenti.
`b_sus,(c_sus)`	Intero: 0 se b (c) non è bloccato, 1 se è in attesa che venga letto il carattere.
`idx`	Variabile temporanea per memorizzare un valore della variabile `tot`.

3.2 Pseudocodice

Assumiamo che ISR_B e ISR_C siano speculari, e che ISR_B sia più prioritaria di ISR_C : se il nodo A riceve un messaggio da C durante l'esecuzione di ISR_B, ISR_C prelaziona ISR_B. Come anticipato nella sezione 1, l'accesso alla sezione critica in cui si controlla ed eventualmente modifica la variabile *possesso* avverrà in mutua esclusione.

```
#define is_reading_b 0
#define is_reading_c 1
#define is_free -1

isr_b(){
    if(!fine){
        if(TAS(lock)){
            if(possesso !=c and !end_b){
                possesso = is_reading_b;
            }
            lock = 0;
        }else{
            RTE;
        }

        switch (possesso){
            case is_reading_b:
                buff[tot] = PIABPRA;
                if(curr == 0 && fase2 && buff[tot] == 0){
                    cond_b = 1;
                }
                tot++;
                curr++;
                if(curr == N){
                    curr = 0;
                    msg++;
                    end_b = 1;
                    possesso = is_free;
                    if(end_c){
                        end_b = 0;
                        end_c = 0;
                        if(msg==k){
                            fase2=1;
                        }
                    }
                    if(fase2 && (cond_b && cond_c)){
                        fine = 1;
                    }
                    if(c_sus && !end_c){
                        tot++;
                        curr++;
                        buff[tot-1] = PIACPRA;
                        if(buff[tot-1] == 0 && fase2){
                            cond_c=1;
                        }
                        possesso = is_reading_c;
                    }
                }
             case is_free:
                if(c_sus && !end_c){
                    tot++;
                    curr++;
                    buff[tot-1] = PIACPRA;
```

```
54                      if(buff[tot-1] == 0 && fase2){
55                          cond_c=1;
56                      }
57                      possesso = is_reading_c;
58                  }
59              case is_reading_c{
60                  if(c_sus){
61                      idx = tot;
62                      tot++;
63                      curr++;
64                      possesso = is_reading_c;
65                      if(curr == N){
66                          curr = 0;
67                          msg++;
68                          end_c = 1;
69                          possesso = is_free;
70                          if (end_b){
71                              end_b = 0;
72                              end_c = 0;
73                              msg++;
74                              if(msg == k){
75                                  fase2=1;
76                              }
77                          }
78                          if(fase2 && (cond_b && cond_c)){
79                              fine = 1;
80                          }
81                          if(!end_b){
82                              buff[tot]=PIABPRA;
83                              tot++;
84                              curr++;
85                              if(fase2 && buff[tot-1]==0){
86                                  cond_b = 1;
87                              }
88                              possesso = is_reading_b;
89                          }
90                      }
91                      buff[idx] = PIACPRA;
92                  }
93              }
94          }
95          RTE;
96      }else{
97          RTE;
98      }
99  }
```

3.3 Codice assembly MC68000

Di seguito viene esposta la codifica in assembly, basata sullo pseudocodice esposto nel paragrafo precedente, del programma eseguito dal nodo A e delle ISR relative alla ricezioni di caratteri sulla parallela proveniente da B e C. I programmi eseguiti dai nodi B e C consistono in semplici cicli di invio di messaggi tramite parallela.

```
1  *** AREA DATI ***
2           ORG     $8000
3  FINE     DC.B    0
4  LOCK     DC.B    0
5  POSS     DC.B    -1
6  CURR     DC.B    0
7  TOT      DC.B    0
8  MSG      DC.B    0
9  END_B    DC.B    0
10 END_C    DC.B    0
11 FASE2    DC.B    0
12 COND_B   DC.B    0
13 COND_C   DC.B    0
14 BUFF     DS.B    32
15
16 *** AREA CODICE ***
17          ORG     $8200
18 PIABPRA  EQU     $2004
19 PIABCRA  EQU     $2005
20 PIACPRA  EQU     $2008
21 PIACCRA  EQU     $2009
22 N        EQU     3
23 K        EQU     6
24
25 MAIN     JSR     PIAINIT
26          MOVE.W  SR,D0
27          ANDI.W  #$D8FF,D0
28          MOVE.W  D0,SR
29 LOOP     JMP     LOOP
30
31
32
33 PIAINIT  MOVE.B  #0,PIABCRA
34          MOVE.B  #$00,PIABPRA
35          MOVE.B  #%00100101,PIABCRA
36          MOVE.B  #$00,PIACCRA
37          MOVE.B  #$00,PIACPRA
38          MOVE.B  #%00100101,PIACCRA
39          RTS
40
41          ORG     $8300
42 ISR_B    MOVEM.L D0-D7/A0-A2,-(SP)
43          MOVE.B  FINE,D0
44          CMP.B   #$01,D0
45          BEQ     RETURN
46          TAS     LOCK
47          BNE     RETURN
48          MOVE.B  POSS,D0
49          CMP.B   #1,D0
50          BEQ     CONT0
51          MOVE.B  END_B,D0
52          CMP     #1,D0
53          BEQ     CONT0
```

```
54          MOVE.B   #0,POSS
55 CONT0    MOVE.B   #0,LOCK
56          MOVE.B   POSS,D0
57          CMPI.B   #1,D0
58          BEQ      CASE1
59          MOVE.B   POSS,D0
60          CMPI.B   #-1,D0
61          BEQ      CASE2
62          MOVEA.L  #PIABPRA,A0      * is_reading_b
63          MOVEA.L  #BUFF,A1
64          MOVE.B   TOT,D0
65          MOVE.B   CURR,D1
66          MOVE.B   (A0),(A1,D0)
67          MOVE.B   (A1,D0),D2
68          MOVE.B   FASE2,D3
69          CMPI.B   #0,D2
70          BNE      CONT1
71          CMPI.B   #1,D3
72          BNE      CONT1
73          CMPI.B   #0,D1
74          BNE      CONT1
75          MOVE.B   #1,COND_B
76 CONT1    ADDQ     #1,D0
77          ADDQ     #1,D1
78          MOVE.B   D0,TOT
79          MOVE.B   D1,CURR
80          CMP.B    #N,D1
81          BNE      RETURN
82          MOVE.B   #0,CURR
83          MOVE.B   MSG,D0
84          ADDQ     #1,D0
85          MOVE.B   D0,MSG
86          MOVE.B   #1,END_B
87          MOVE.B   #-1,POSS
88          MOVE.B   END_C,D1
89          CMPI.B   #1,D1
90          BNE      CONT2
91          MOVE.B   #0,END_B
92          MOVE.B   #0,END_C
93          CMP.B    #K,D0
94          BNE      CONT2
95          MOVE.B   #1,FASE2
96 CONT2    MOVE.B   FASE2,D0
97          CMPI.B   #1,D0
98          BNE      CONT3
99          MOVE.B   COND_B,D0
100         CMPI.B   #1,D0
101         BNE      CONT3
102         MOVE.B   COND_C,D0
103         CMPI.B   #1,D0
104         BNE      CONT3
105         MOVE.B   #1,FINE
106 CONT3   MOVE.B   PIACCRA,D0       * controllo c_sus
107         ANDI     #%10000000,D0
108         BEQ      RETURN
109         MOVE.B   END_C,D0
110         CMPI.B   #1,D0
111         BEQ      RETURN
112         MOVEA.L  #PIACPRA,A0
113         MOVEA.L  #BUFF,A1
114         MOVE.B   TOT,D0
115         MOVE.B   TOT,D7
```

```
116          MOVE.B   CURR,D1
117          ADDQ     #1,D0
118          ADDQ     #1,D1
119          MOVE.B   D0,TOT
120          MOVE.B   D1,CURR
121          MOVE.B   (A0),(A1,D7)
122          MOVE.B   (A1,D7),D0
123          MOVE.B   FASE2,D1
124          CMPI.B   #1,D1
125          BNE      CONT4
126          CMPI.B   #0,D0
127          BNE      CONT4
128          MOVE.B   #1,COND_C
129 CONT4    MOVE.B   #1,POSS
130          JMP      RETURN
131 CASE1    MOVE.B   PIACCRA,D0       * is_reading_c
132          ANDI     #%10000000,D0
133          BEQ      RETURN
134          MOVE.B   TOT,D7
135          MOVE.B   TOT,D0
136          MOVE.B   CURR,D1
137          ADDQ     #1,D0
138          ADDQ     #1,D1
139          MOVE.B   D0,TOT
140          MOVE.B   D1,CURR
141          MOVE.B   #1,POSS
142          CMPI.B   #N,D1
143          BNE      CONT5
144          MOVE.B   #0,CURR
145          MOVE.B   MSG,D0
146          ADDQ     #1,D0
147          MOVE.B   D0,MSG
148          MOVE.B   #1,END_C
149          MOVE.B   #-1,POSS
150          MOVE.B   END_B,D0
151          CMPI.B   #1,D0
152          BNE      CONT6
153          MOVE.B   #0,END_B
154          MOVE.B   #0,END_C
155          MOVE.B   MSG,D0
156          CMPI.B   #K,D0
157          BNE      CONT6
158          MOVE.B   #1,FASE2
159 CONT6    MOVE.B   FASE2,D0
160          CMPI.B   #1,D0
161          BNE      CONT7
162          MOVE.B   COND_B,D0
163          CMPI.B   #1,D0
164          BNE      CONT7
165          MOVE.B   COND_C,D0
166          CMPI.B   #1,D0
167          BNE      CONT7
168          MOVE.B   #1,FINE
169 CONT7    MOVE.B   END_B,D0
170          CMPI.B   #1,D0
171          BEQ      CONT5
172          MOVE.B   TOT,D0
173          MOVE.B   CURR,D1
174          MOVEA.L  #PIABPRA,A0
175          MOVEA.L  #BUFF,A1
176          MOVE.B   (A0),(A1,D0)
177          MOVE.B   (A1,D0),D2
```

```
178           ADDQ      #1,D0
179           ADDQ      #1,D1
180           MOVE.B    D0,TOT
181           MOVE.B    D1,CURR
182           MOVE.B    FASE2,D3
183           CMPI.B    #1,D3
184           BNE       CONT8
185           CMPI.B    #0,D2
186           BNE       CONT8
187           MOVE.B    #1,COND_B
188 CONT8     MOVE.B    #0,POSS
189 CONT5     MOVEA.L   #PIACPRA,A0
190           MOVEA.L   #BUFF,A1
191           MOVE.B    (A0),(A1,D7)
192           JMP       RETURN
193 CASE2     MOVE.B    PIACCRA,D0        * is_free
194           ANDI      #%10000000,D0
195           BEQ       RETURN
196           MOVE.B    END_C,D0
197           CMPI.B    #1,D0
198           BEQ       RETURN
199           MOVE.B    TOT,D0
200           MOVE.B    CURR,D1
201           MOVEA.L   #PIACPRA,A0
202           MOVEA.L   #BUFF,A1
203           MOVE.B    D0,D7
204           ADDQ      #1,D0
205           ADDQ      #1,D1
206           MOVE.B    D0,TOT
207           MOVE.B    D1,CURR
208           MOVE.B    (A0),(A1,D7)
209           MOVE.B    (A1,D7),D2
210           MOVE.B    FASE2,D3
211           CMPI.B    #1,D3
212           BNE       CONT9
213           CMPI.B    #0,D2
214           BNE       CONT9
215           MOVE.B    #1,COND_C
216 CONT9     MOVE.B    #1,POSS
217           JMP       RETURN
218 RETURN    MOVEM.L   (SP)+,D0-D7/A0-A2
219           RTE
```

La ISR_C è stata omessa per motivi di spazio, essendo perfettamente speculare alla ISR_B.