Corso di Laurea Magistrale in **Ingegneria Informatica**

Elaborato finale per il Corso di Calcolatori Elettronici 2

Prof. Valentina Casola

A.A. 2019/2020

Studente:

Mario Pace M63000988

Sommario

1. Specifiche di progetto

Un sistema X è dotato di una periferica seriale e di una periferica parallela. Il sistema trasmette un messaggio di 32 caratteri (byte) sulla periferica seriale e ottiene l'eco del messaggio sulla periferica parallela.
Lo studente può scegliere diverse ipotesi di funzionamento e dovrà opportunamente motivarle nel progetto. A titolo esemplificativo si potrebbe optare per una delle seguenti modalità di funzionamento:

1) **Il messaggio è trasmesso interamente dalla seriale e successivamente è ricevuto mediante il meccanismo delle interruzioni sulla periferica parallela. Non può essere inviato un altro messaggio sulla seriale se non è stato ricevuto prima l'eco sulla parallela.**
2) Non viene fatta alcuna ipotesi di correlazione tra le attività della periferica seriale e la parallela.

Si illustrino:

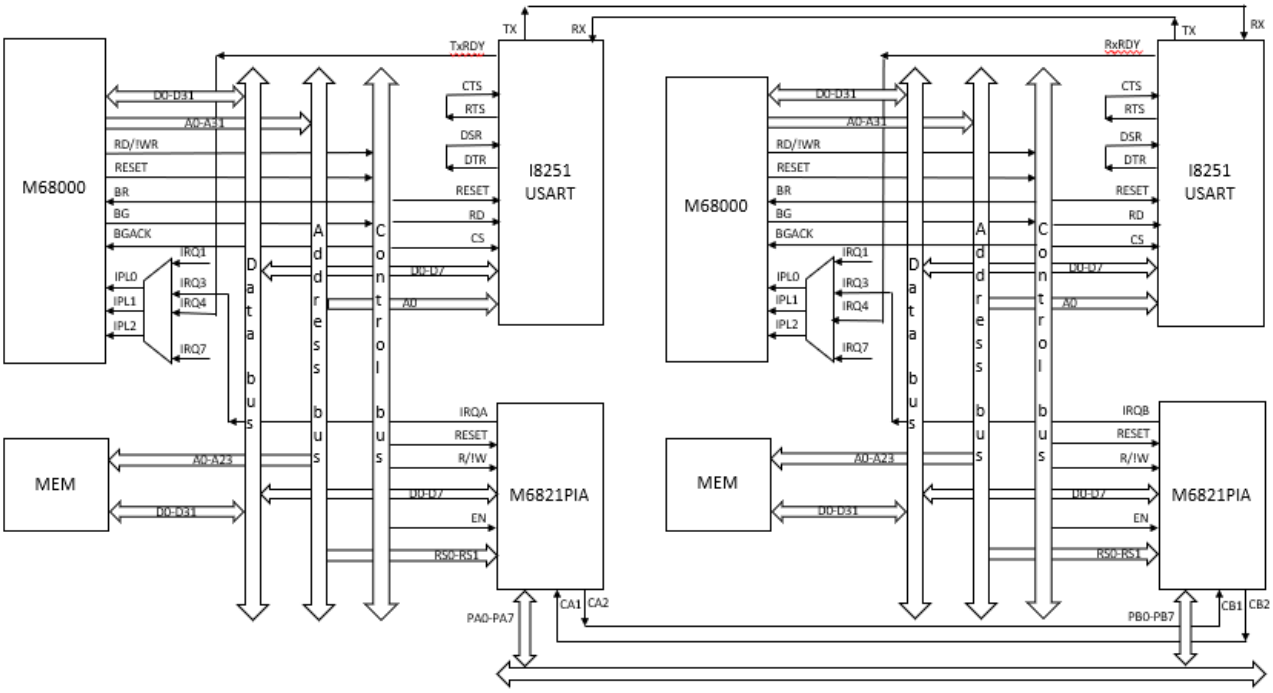
- l'architettura complessiva del sistema;
- il collegamento tra i dispostivi;
- i protocolli;
- il software e la memoria con riferimento a dati e programmi in essa allocati

Nel progetto si è scelto di utilizzare l'ipotesi di funzionamento 1 (in grassetto nella traccia).

Si è inoltre scelto di utilizzare il meccanismo delle interruzioni sia in trasmissione sia in ricezione per entrambe le periferiche di entrambi i sistemi X e Y.

2. Soluzione

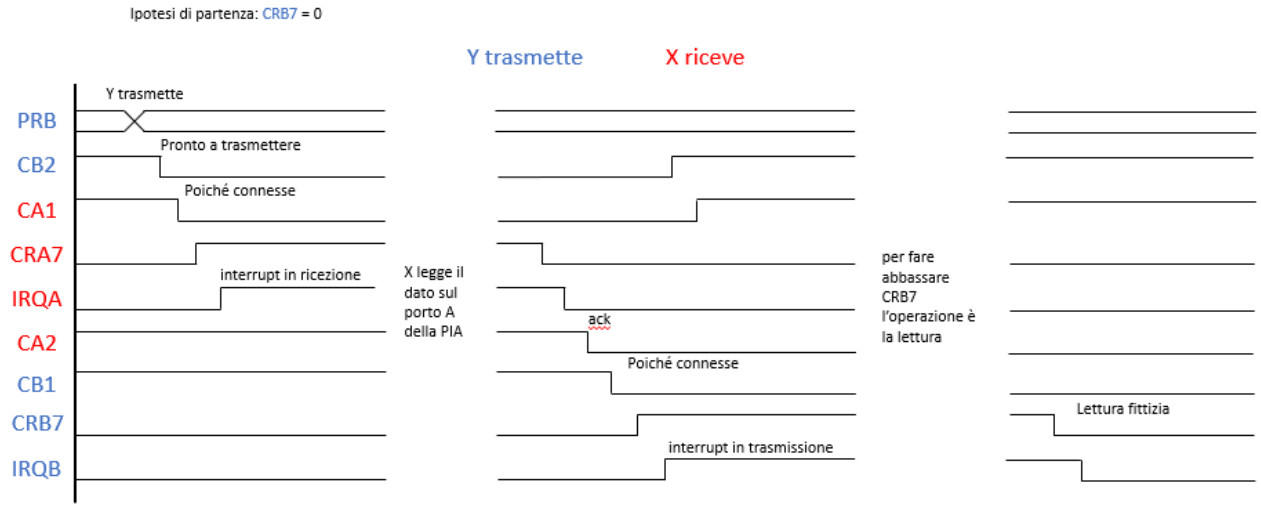
2.1. Architettura del sistema



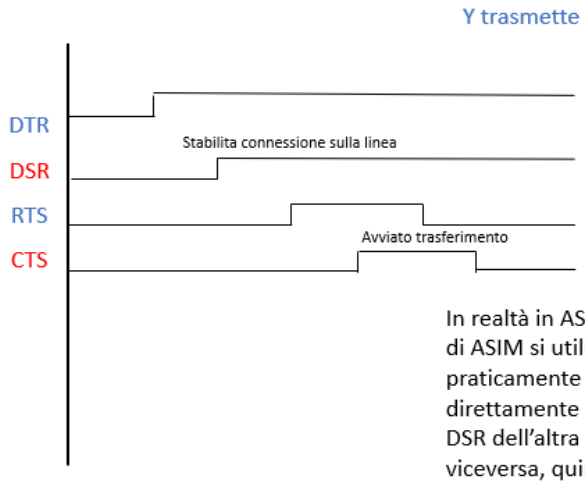
Sulla sinistra è rappresentato il sistema X, sulla destra il sistema Y (non specificato nella traccia ma necessario per la ricezione dei dati seriali e la trasmissione dell'eco parallelo).

2.2. Protocolli

Protocollo di comunicazione PIA-PIA

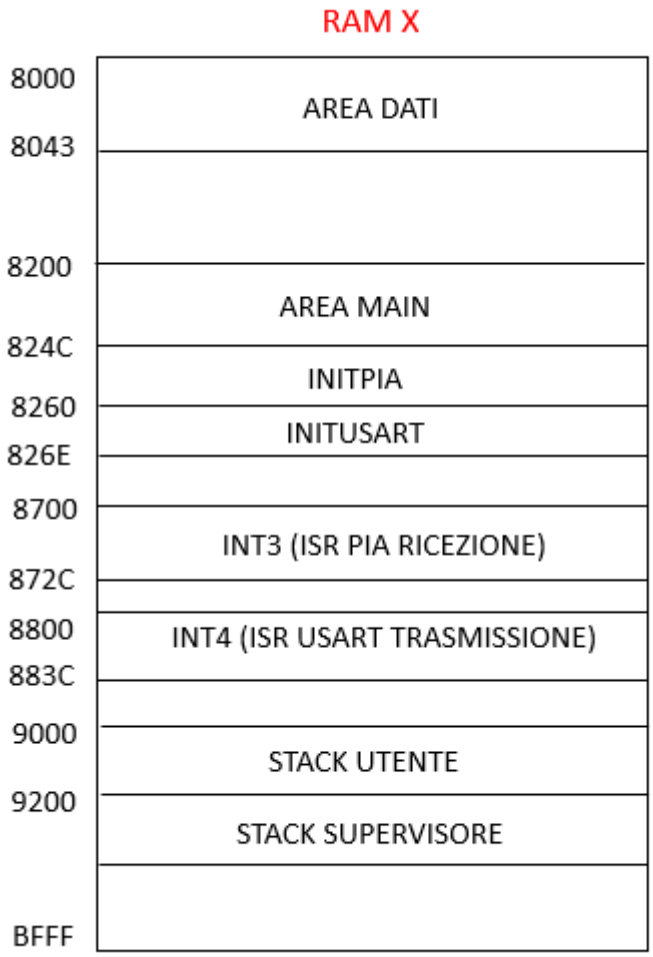


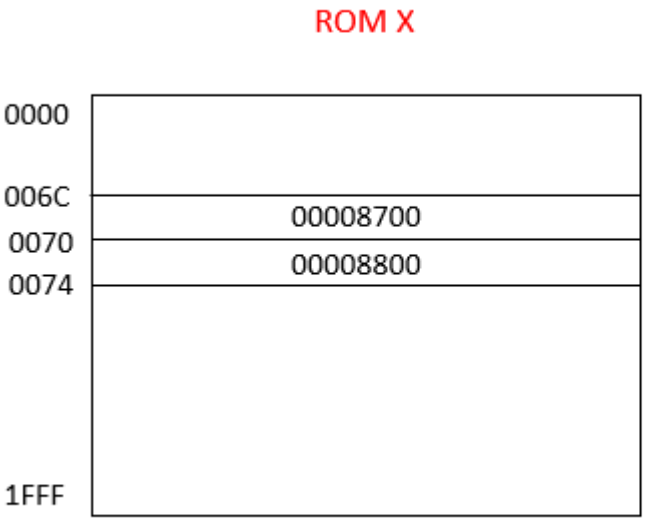
Protocollo di comunicazione USART-USART



In realtà in ASIM non è presente questo protocollo di handshaking : Nel caso di ASIM si utilizza una configurazione particolare detta NULLMODEM, dove praticamente il modem non viene utilizzato, le due periferiche seriali sono direttamente connesse tra di loro e in particolare il DTR dell'una finisce nel DSR dell'altra e viceversa, e il bit RTS dell'una va a finire nel CTS dell'altra e viceversa, quindi sono collegate in maniera incrociata tra di loro. All'interno del registro di controllo io devo porre direttamente a 1 il bit DTR e direttamente a 1 il bit RTS, quindi di fatto stabilisco in maniera fissa questo handshake, cioè le due periferiche sono pronte a comunicare e disponibili a scambiarsi i messaggi.

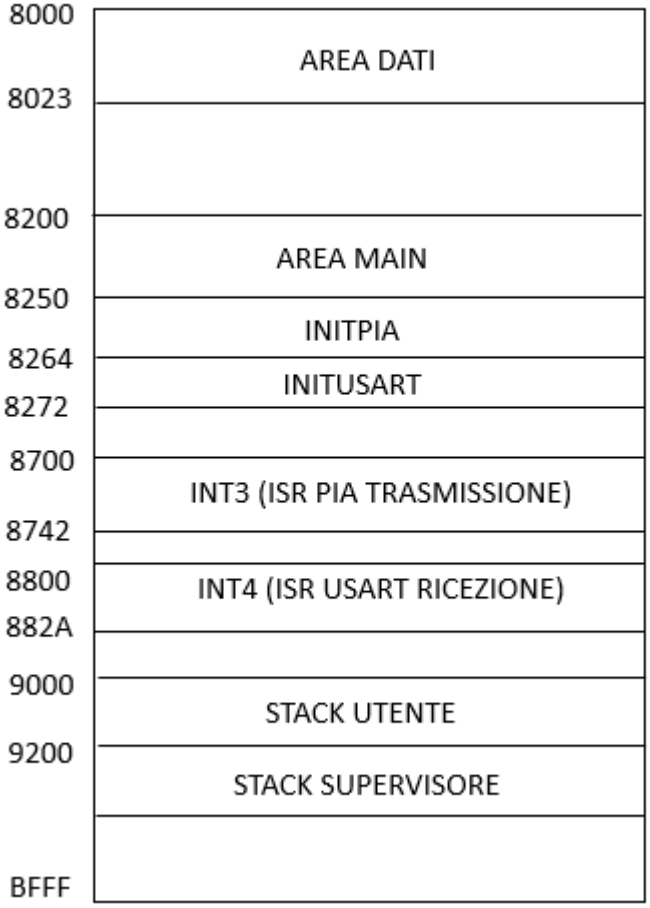
2.3. Mappa della memoria



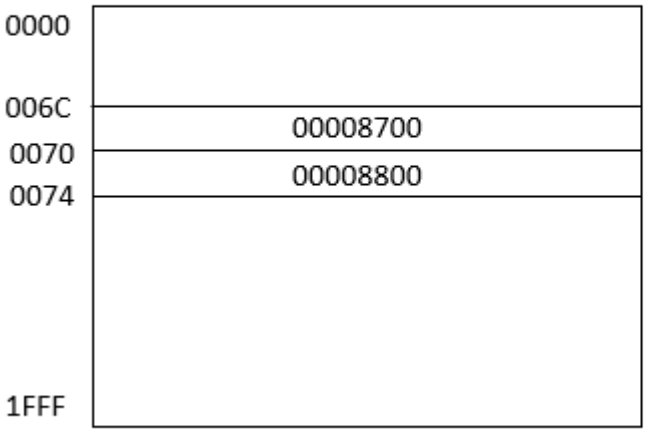


Modalità autovettorizzata: utilizzo terzo e quarto autovettore, quindi locazioni 27*4=108 -> 6C e 28*4=112 -> 70

RAM Y



ROM Y

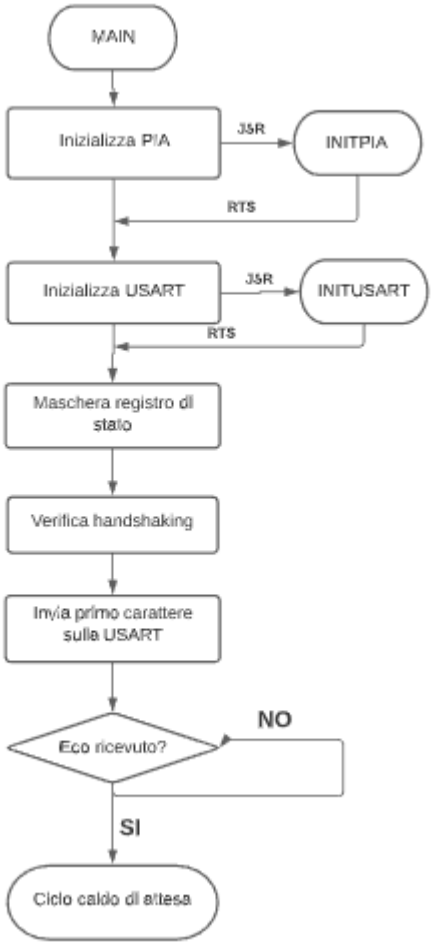


Modalità autovettorizzata: utilizzo terzo e quarto autovettore, quindi locazioni 27*4=108 -> 6C e 28*4=112 -> 70

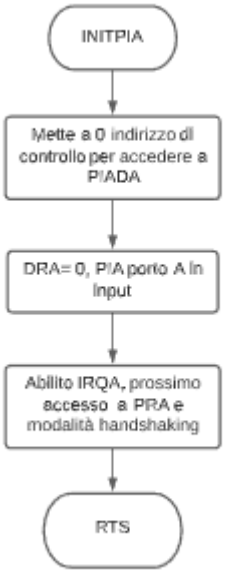
2.4. Implementazione

2.4.1. Documentazione del codice

Sistema X MAIN :



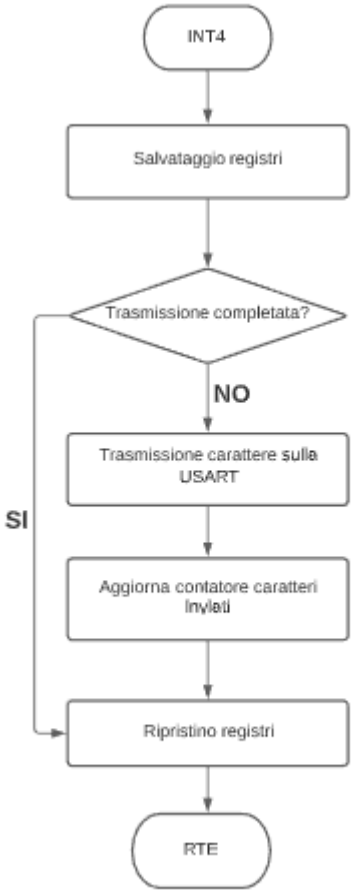
Sistema X INITPIA:



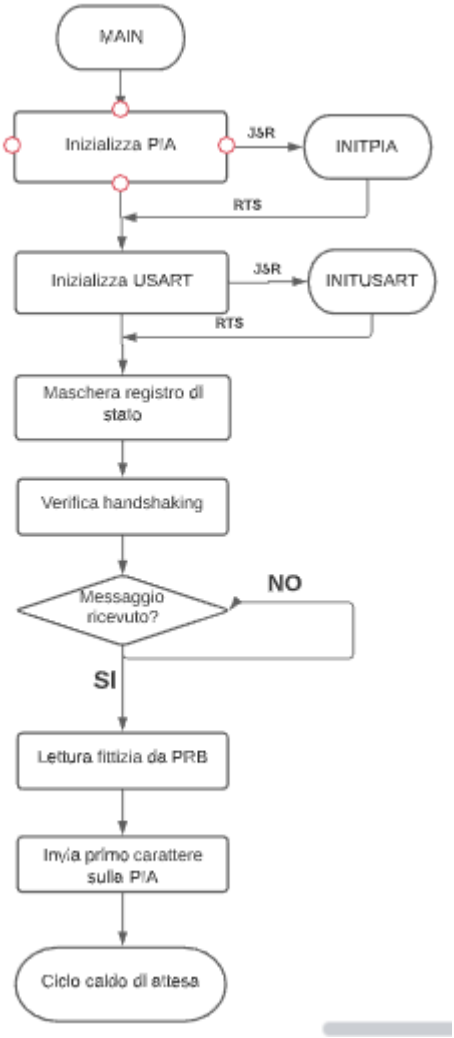
Sistema X INITUSART:

Sistema X INT3 (interruzione PIA ricezione carattere):

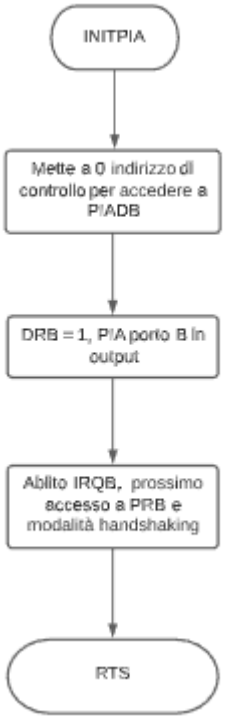
Sistema X INT4 (interruzione USART trasmissione carattere) :



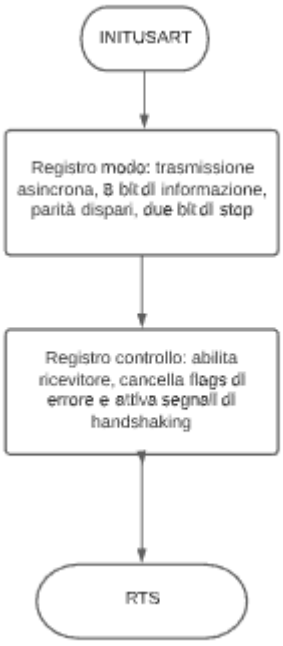
Sistema Y MAIN:



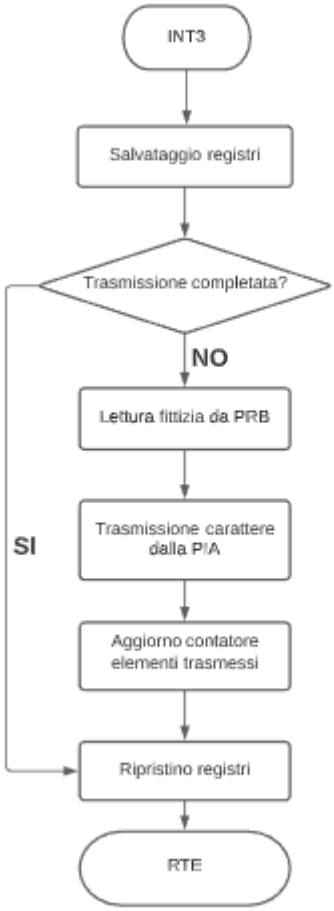
Sistema Y INITPIA:



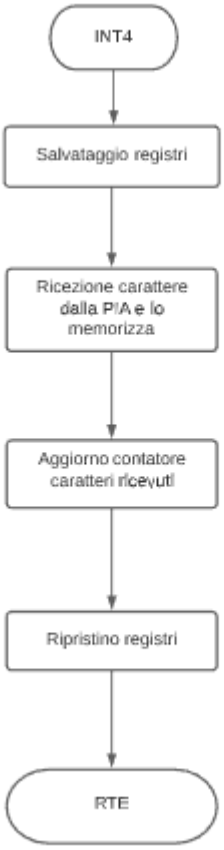
Sistema Y INITUSART:



Sistema Y INT3 (interruzione PIA trasmissione carattere) :



Sistema Y INT4 (interruzione USART ricezione carattere) :



2.4.2. Codice Assembly

	SISTEMA X
AREA DATI	ORG $8000 MSG DC.B 1,2,3,1,2,3,1,2,3,1,2,3,1,2,3,1,2,3,1,2,3,1,2,3,1,2,3,4,5 ;messaggio da inviare ECO DS.B 32 ;spazio per memorizzare l'eco del messaggio DIM DC.B 32 ;dimensione del messaggio sia da inviare che da ricevere COUNTI DC.B 1 ;contatore caratteri inviati, inizializzato a 1 perchè il primo carattere lo invio nel main COUNTR DC.B 0 ;contatore caratteri ricevuti
MAIN	ORG $8200 PIADA EQU $2004 ;indirizzo della PIA porto A dato, usato in input per ricevere l'eco PIACA EQU $2005 ;indirizzo della PIA porto A stato/controllo USARTD EQU $2008 ;registro dato della USART USARTC EQU $2009 ;registro di controllo della USART MAIN JSR INITPIA ; subroutine che inizializza PIA porto A in input JSR INITUSART ;subroutine che inizializza USART in output MOVE.W SR,D0 ;legge il registro di stato ANDI.W #$D8FF,D0 ;maschera per reg stato (imposta stato utente e abilita tutte le interruzioni) MOVE.W D0,SR ; copia valore nel registro di stato MOVEA.L #USARTD,A1 ;indirizzo registro dato in A1 MOVEA.L #USARTC,A2 ;indirizzo registro controllo/stato in A2 MOVEA.L #MSG,A0 ;indirizzo area messaggio in A0 CLR D1 ;pulisco il registro di appoggio CLR D2 ;pulisco il contatore elementi trasmessi CHECKDSR MOVE.B (A2),D3 ;Verifica che DSR = 1, inutile per la configurazione con DTR=1 in questo caso ANDI.B #$80,D3 BEQ CHECKDSR PRIMO MOVE.B (A0)+,D1 MOVE.B D1,(A1) ;invio primo carattere sulla seriale LOOP MOVE.B COUNTR,D4 ; attendo che l'eco venga ricevuto prima di inviare un nuovo messaggio MOVE.B DIM,D0 CMP D0,D4 BNE LOOP NEW JMP NEW ; non è previsto l'invio di ulteriori messaggi, quindi mi fermo in un loop
SUBR	INITPIA MOVE.B #0,PIACA ;mette 0 nel registro controllo così al prossimo accesso sarà a PIADA MOVE.B #$00,PIADA ;accede a DRA e pone DRA=0 : le linee di A sono linee di input MOVE.B #%00100101,PIACA ;abilito IRQA (ricezione), prossimo accesso a PRA e modalità handshaking RTS
SUBR2	INITUSART MOVE.B #$5D,USARTC ;tr.asincrona, 8 bit informazione, parità dispari e 2 bit di stop MOVE.B #$23,USARTC ;abilita trasmettitore e attiva i segnali di handshaking. RTS
INT3	ORG $8700 ; interruzione della PIA per leggere il carattere INT3 MOVE.L A1,-(A7) ;salvataggio registri

```
        MOVE.L  A0,-(A7)
        MOVE.L  D0,-(A7)

        MOVEA.L         #PIADA,A1
        MOVEA.L         #ECO,A0  ;indirizzo area di salvataggio
        MOVE.B COUNTR,D0 ;contatore corrente degli elementi ricevuti
        MOVE.B (A1),(A0,D0)     ;acquisisce il carattere e lo trasferisce in memoria
        ADD.B   #1,D0  ; incremento il contatore dei caratteri ricevuti
        MOVE.B D0,COUNTR  ; e lo memorizzo

        MOVE.L  (A7)+,D0        ;ripristino registri
        MOVE.L  (A7)+,A0
        MOVE.L  (A7)+,A1

        RTE
```

```
INT4    ORG     $8800           ; interruzione della USART per inviare il carattere

INT4    MOVE.L  A1,-(A7)        ;salvataggio registri
        MOVE.L  A0,-(A7)
        MOVE.L  D0,-(A7)
        MOVE.L  D1,-(A7)

        MOVEA.L         #USARTD,A1
        MOVEA.L         #MSG,A0          ;indirizzo area del messaggio da inviare

        MOVE.B COUNTI,D0        ;contatore corrente degli elementi inviati
        MOVE.B DIM,D1
        CMP.B   D0,D1                   ;controlla se devo trasmettere altri caratteri
        BEQ     FINE

INVIO   ADDA.L  D0,A0   ; mi posiziono all'indirizzo del dato da inviare
        MOVE.B (A0),D1
        MOVE.B D1,(A1)          ;trasmette il carattere sulla seriale

        ADD.B   #1,D0
        MOVE.B D0,COUNTI        ;aggiorna contatore caratteri inviati

FINE    MOVE.L  (A7)+,D1        ;ripristino registri
        MOVE.L  (A7)+,D0
        MOVE.L  (A7)+,A0
        MOVE.L  (A7)+,A1

        RTE
```

SISTEMA Y

```
AREA    ORG     $8000
DATI

MSG             DS.B    32      ; spazio per memorizzare il messaggio ricevuto e poi da reinviare
DIM             DC.B    32      ; dimensione del messaggio
COUNTI          DC.B    1       ; contatore dei caratteri inviati, inizializzato a 1 perchè il primo lo invio dal main
COUNTR          DC.B    0       ; contatore dei caratteri ricevuti
```

```
MAIN    ORG     $8200

PIADB   EQU     $2006   ;indirizzo di PIA porto B dato, usato in output
PIACB   EQU     $2007   ;indirizzo di PIA porto B stato/controllo
USARTD  EQU     $2008           ;registro dato della USART
USARTC  EQU     $2009           ;registro di controllo della USART
```

```
MAIN    JSR     INITPIA     ; subroutine che inizializza PIA porto B in output
        JSR     INITUSART       ; subroutine che inizializza la USART in input


        MOVE.W SR,D0    ;legge il registro di stato
        ANDI.W  #$D8FF,D0 ;maschera per reg stato (imposta stato utente e abilita tutte le interruzioni)
        MOVE.W D0,SR    ; copia valore nel registro di stato

        MOVEA.L         #USARTC,A2      ;indirizzo registro controllo/stato

CHECKDSR    MOVE.B (A2),D3  ;Verifica che DSR = 1, inutile per la configurazione con DTR=1 in questo caso
            ANDI.B          #$80,D3
            BEQ             CHECKDSR

LOOP    MOVE.B COUNTR,D4   ;attende che l'intero messaggio venga ricevuto sulla USART prima di inviare l'eco
        MOVE.B DIM,D0
        CMP    D0,D4
        BNE    LOOP

        MOVEA.L         #PIACB,A1       ;indirizzo registro di controllo CRB
        MOVEA.L         #PIADB,A2       ;indirizzo registro PRB
        MOVEA.L         #MSG,A0         ;indirizzo area messaggio


INVIO1  MOVE.B (A2),D1          ;lettura fittizia da PRB => serve per azzerare CRB7
        MOVE.B (A0),D1              ;carattere corrente da trasferire in D1;
        MOVE.B D1,(A2)              ;dato su bus di PIA porto B

LOOP2   JMP LOOP2       ;ciclo caldo dove il processore attende interrupt
```

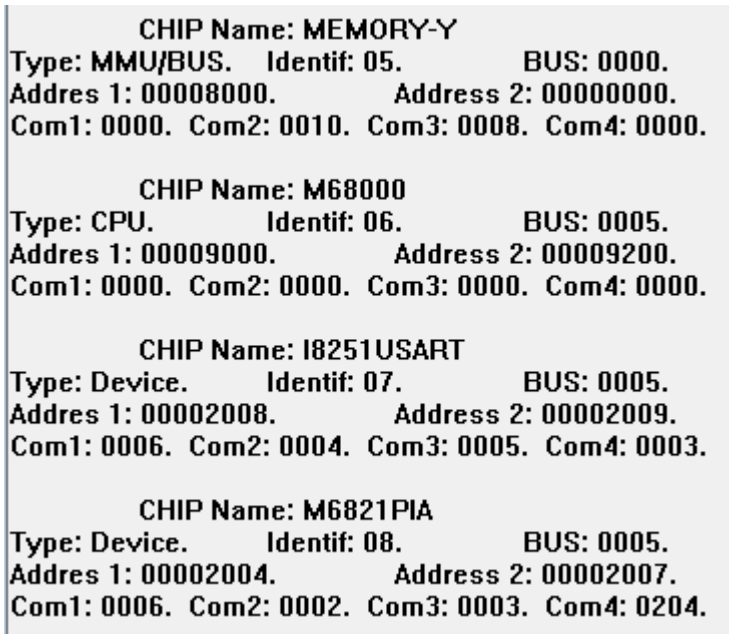
```
SUBR    INITPIA         MOVE.B #0,PIACB         ;seleziona il registro direzione di PIA porto B
                        MOVE.B #$FF,PIADB           ;accede a DRB e pone DRB=1 : le linee di B sono linee di output
                        MOVE.B #%00100101,PIACB   ;abilito IRQB, prossimo accesso a PRB e modalità handshaking
                        RTS


SUBR2   INITUSART       MOVE.B #$5D,USARTC      trasmissione asincrona, 8 bit di informazione
                        MOVE.B #$36,USARTC      ;abilita ricevitore, cancella flags di errore, attiva segnali handshaking.
                        RTS
```

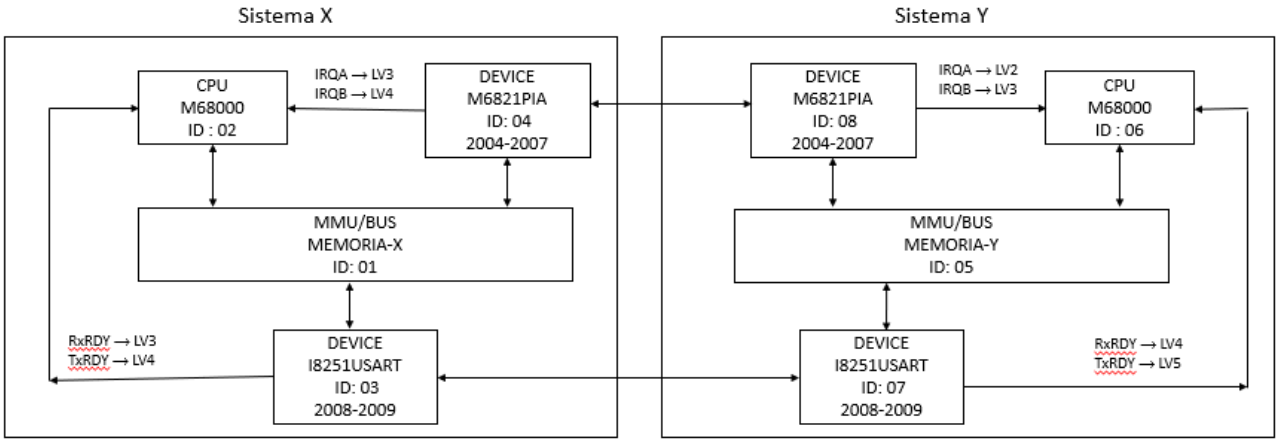
```
INT3            ORG $8700               ; interruzione della PIA per trasmettere il carattere

INT3    MOVE.L  A1,-(A7)            ;salvataggio registri
        MOVE.L  A0,-(A7)
        MOVE.L  D0,-(A7)
        MOVE.L  D1,-(A7)
        MOVE.L  D2,-(A7)

        MOVEA.L         #PIADB,A1
        MOVEA.L         #MSG,A0         ;indirizzo area del carattere da inviare
        MOVE.B  DIM,D0  ;dimensione del messaggio
        MOVE.B  COUNTI,D1       ;contatore corrente dei caratteri inviati

        CMP.B   D1,D0
        BEQ     FINE

INVIO   MOVE.B (A1),D2          ;lettura fittizia da PRB => serve per azzerare CRB7 dopo il primo carattere
        MOVE.B (A0,D1),(A1)     ;carattere corrente da trasferire in D2;
        ADD.B   #1,D1                   ;aggiorno il contatore degli elementi trasmessi
        MOVE.B  D1,COUNTI
```

```
        FINE    MOVE.L  (A7)+,D2          ;ripristino registri
                MOVE.L  (A7)+,D1
                MOVE.L  (A7)+,D0
                MOVE.L  (A7)+,A0
                MOVE.L  (A7)+,A1

                RTE
```

```
INT4            ORG             $8800           ; interruzione per ricevere caratteri sulla USART

        INT4    MOVE.L  A1,-(A7)        ;salvataggio registri
                MOVE.L  A0,-(A7)
                MOVE.L  D0,-(A7)

                MOVEA.L         #USARTD,A1
                MOVEA.L         #MSG,A0         ;indirizzo area di salvataggio
                MOVE.B  COUNTR,D0       ;contatore corrente dei caratteri ricevuti

                MOVE.B  (A1),(A0,D0)    ;riceve un carattere e lo memorizza

                ADD.B   #1,D0
                MOVE.B  D0,COUNTR       ;aggiorna contatore caratteri ricevuti

                MOVE.L  (A7)+,D0 ;ripristino registri
                MOVE.L  (A7)+,A0
                MOVE.L  (A7)+,A1

                RTE
```

2.4.3. Simulazione in ASIM




Sistema X
CPU
M68000
ID : 02
IRQA → LV3
IRQB → LV4
DEVICE
M6821PIA
ID: 04
2004-2007
MMU/BUS
MEMORIA-X
ID: 01
RxRDY → LV3
TxRDY → LV4
DEVICE
I8251USART
ID: 03
2008-2009
Sistema Y
DEVICE
M6821PIA
ID: 08
2004-2007
IRQA → LV2
IRQB → LV3
CPU
M68000
ID : 06
MMU/BUS
MEMORIA-Y
ID: 05
DEVICE
I8251USART
ID: 07
2008-2009
RxRDY → LV4
TxRDY → LV5