



UNIVERSITA' DEGLI STUDI DI  
NAPOLI FEDERICO II

Scuola Politecnica e delle Scienze di Base  
Corso di Laurea Magistrale in Ingegneria Informatica

Elaborato **Computer Systems Design**

## ***Simulazione prova scritta***

Anno Accademico 2020/21

Studente:

**Michele Maresca M63/1151**



# Indice

---

<b>INDICE</b>	<b>III</b>
<b>1. SPECIFICHE DI PROGETTO</b>	<b>4</b>
<b>2. ARCHITETTURA DEL SISTEMA</b>	<b>5</b>
<b>3. PROTOCOLLI</b>	<b>7</b>
<b>4. MAPPA DELLA MEMORIA</b>	<b>8</b>
4.1 MEMORIA NODO A	8
4.1.1 <i>Area Periferiche</i>	8
4.1.2 <i>Area Interruzioni</i>	8
4.1.3 <i>Area Dati</i>	8
4.2 MEMORIA NODO B	9
4.2.1 <i>Area Periferiche</i>	9
4.2.2 <i>Area Interruzioni</i>	9
4.2.3 <i>Area Dati</i>	9
4.3 MEMORIA NODO C	10
4.3.1 <i>Area Periferiche</i>	10
4.3.2 <i>Area Dati</i>	10
<b>5. DESCRIZIONE DI ALTO LIVELLO DEL PROGRAMMA IMPLEMENTATO</b>	<b>11</b>
5.1 MAIN	11
5.2 INTERRUZIONE RICEZIONE PIA VERSIONE 1	12
5.3 INTERRUZIONE RICEZIONE USART VERSIONE 1	13
5.4 INTERRUZIONE RICEZIONE PIA VERSIONE 2	14
5.5 INTERRUZIONE RICEZIONE USART VERSIONE 2	15
<b>6. IMPLEMENTAZIONE</b>	<b>18</b>
6.1 CODICE VERSIONE 1	18
6.2 CODICE VERSIONE 2	23

# 1. Specifiche di Progetto

---

Un sistema è composto da 3 unità A, B e C. B è collegato ad A mediante una periferica seriale, e a C mediante una periferica parallela. Il sistema opera come segue:

A invia fino ad un massimo di M messaggi di N byte a B. Per ogni messaggio ricevuto  $MSG_i$ , B verifica l'ultimo byte del messaggio  $MSG_i(N-1)$ :

- Se è diverso da 0, B continua con la ricezione;
- Se è uguale a 0, B interrompe la comunicazione con A e C.

Durante la ricezione dei messaggi (in qualsiasi momento), il sistema B può ricevere dei caratteri da C. In particolare, se riceve 2 caratteri (qualsiasi) successivi da C, B termina la ricezione del messaggio eventualmente in sospeso e poi interrompe la comunicazione con A e C.

## 2. Architettura del Sistema

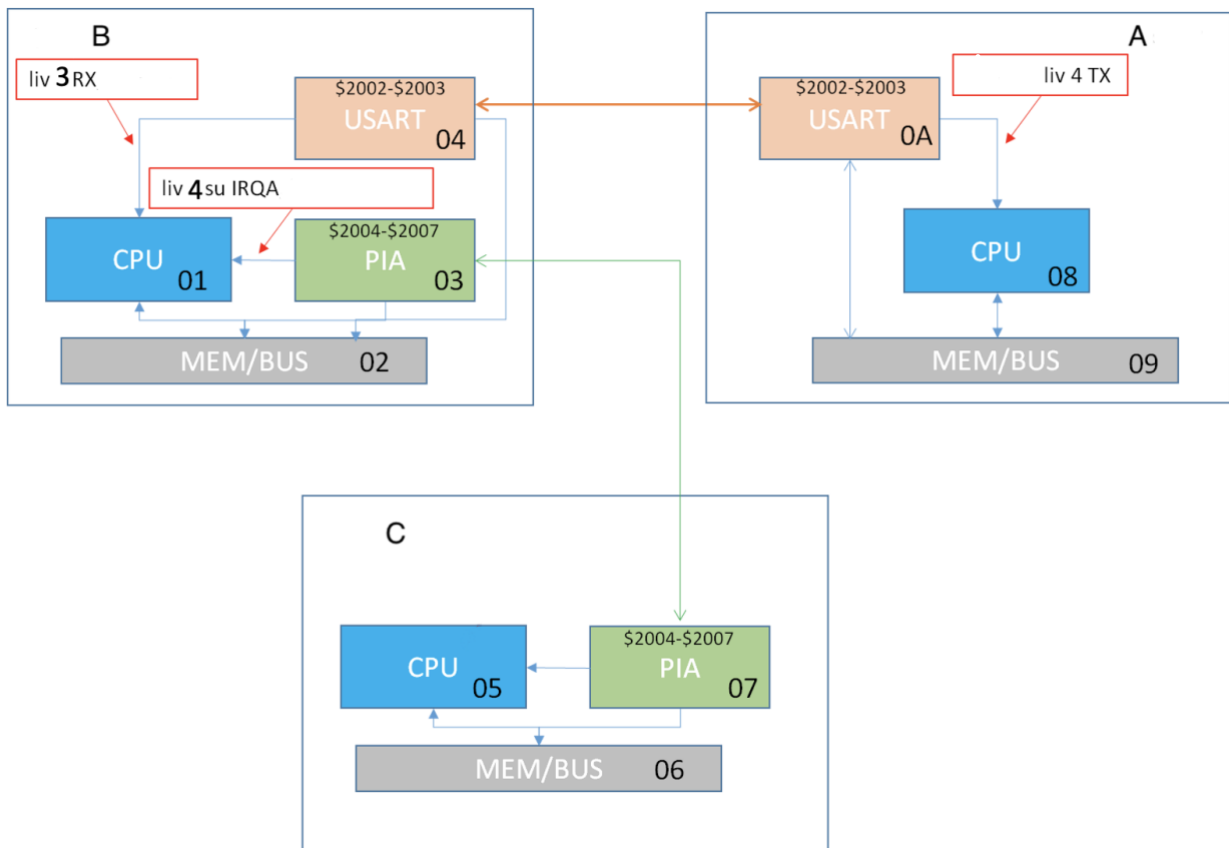


Figura 2.1: Architettura del Sistema Complessivo.

Nella Figura 2.1 è raffigurata l'architettura del sistema complessivo, il quale presenta i tre nodi A, B e C collegati tra loro.

In particolare, il nodo B è costituito da un processore M68000, una ROM di 8K (addr \$0-\$1FFF), una RAM di 10K (addr \$8000-\$A7FF), un device parallelo PIA mappato a \$2004-\$2007, un device seriale USART mappato a \$2002-\$2003.

La PIA è collegata alla linea di interruzione 4 dedicata alla ricezione, invece la USART è collegata alla linea di interruzione 3 dedicata alla ricezione.

Il nodo B è collegato al nodo A mediante una USART.

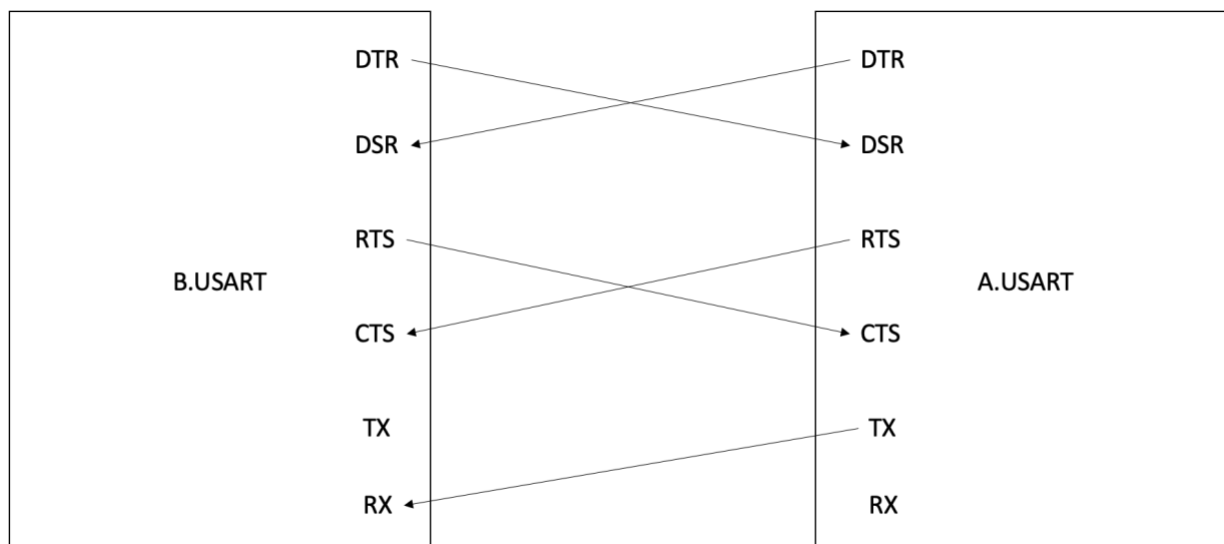
Il nodo A è costituito da un processore M68000, una ROM di 8K (addr \$0-\$1FFF), una RAM di 10K (addr \$8000-\$A7FF), e un device seriale USART mappato a \$2002-\$2003.

Il nodo B è collegato al nodo C mediante una PIA.

Il nodo C è costituito da un processore M68000, una ROM di 8K (addr \$0-\$1FFF), una RAM di 10K (addr \$8000-\$A7FF), e un device parallelo PIA mappato a \$2004-\$2007.

In Figura 2.2 è rappresentato il collegamento tra il nodo B e il nodo A.

In particolare, è rappresentato il collegamento tra le due USART, in configurazione "NULL MODEM", la quale prevede che i segnali di handshaking vengano direttamente scambiati tra i due terminali, senza la presenza di un modem. Il DSR (Data Set Ready) di un terminale è collegato con il DTR (Data Terminal Ready) dell'altro terminale e viceversa. Il RTS (Request To Send) di un terminale è collegato con il CTS (Clear To Send) dell'altro e viceversa. Il TX del nodo A è collegato a RX del nodo B.



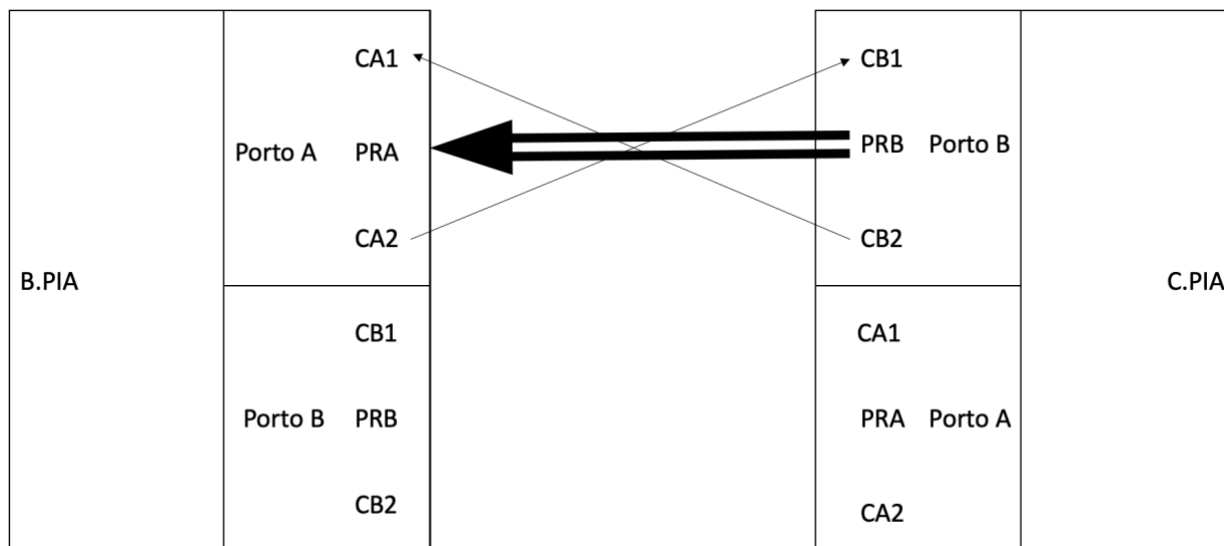
*Figura 2.2: Collegamento nodo B e nodo A attraverso USART*

In Figura 2.3 è rappresentato il collegamento tra il nodo B e il nodo C.

In particolare, la linea CB2 del nodo C è posta in uscita ed è connessa alla linea CA1 del nodo B, la quale è posta in ingresso.

La linea CA2 del nodo B è posta in uscita ed è connessa alla linea CB1 del nodo C, la quale è posta in ingresso.

La uscita dati PRB del nodo C è connessa all'ingresso dati PRA del nodo B.



*Figura 2.3: Collegamento nodo B e nodo C attraverso PIA*

### 3. Protocolli

Nella Figura 3.1 si riporta il protocollo di trasmissione della USART del nodo A.

Il nodo A asserisce DTR per comunicare al nodo B che è ON e pronto ad iniziare la comunicazione.

Il nodo B asserisce DSR per comunicare al nodo A che è ON ed è pronto a ricevere dati.

Il nodo A asserisce RTS quando vuole trasmettere dati.

Il nodo B asserisce CTS se è pronto a ricevere.

Nella configurazione usata i sistemi sono entrambi inizializzati ponendo DTR = 1 e RTS = 1.

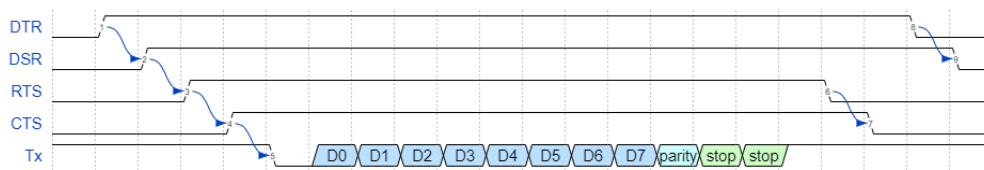


Figura 3.1: Protocollo di trasmissione della USART del nodo A.

Nella Figura 3.2 è rappresentata la comunicazione tra le due PIA: nel momento in cui il Porto B della PIA del nodo C comincia la trasmissione, scrivendo su PRB, abbassa CB2, il quale a sua volta è collegato a CA1 del Porto A del nodo B. Quando CA1 si abbassa, si alza CRA7, bit collegato all'interruzione del Porto A, scatenando di conseguenza l'alzarsi del segnale IRQA e viene generata l'interrupt. Si alza CA2 del nodo B quando IRQA = 1, in seguito a variazione di CA1. Inoltre, si alza CB1 del nodo C, essendo collegato a CA2.

Nel momento in cui l'ISR effettua la lettura da PRA del nodo B, si abbassa CA2 del nodo B, e di conseguenza si abbassa CB1 del nodo C, facendo alzare IRQB (IRQB = 1). In seguito a IRQB = 1, si alza CB2 del nodo C e quindi si alza anche CA1 del nodo B.

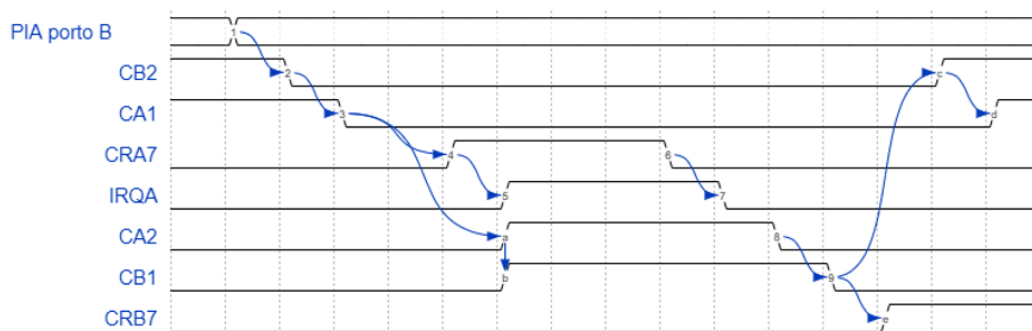


Figura 3.2: Protocollo di comunicazione tra le PIA.

## 4. Mappa della memoria

---

Di seguito viene riportata la mappa della memoria del tre nodi.

In particolare, sono state suddivisa in tre sezioni: Area Periferiche, Area Interruzioni ed Area Dati.

### 4.1 Memoria nodo A

#### 4.1.1 Area Periferiche

La USART è mappata su 2 indirizzi consecutivi, \$2002 e \$2003.

USARTP (pari)	\$2002
USARTD (dispari)	\$2003

#### 4.1.2 Area Interruzioni

La modalità di riconoscimento delle interruzioni utilizzata è la modalità autovettorizzata. Per calcolare l'indirizzo nella tabella delle interruzioni bisogna calcolare  $(24 + IPL) \times 4$ , dove IPL è il livello della linea dell'Interrupt Controller al quale il device che genera l'interruzione è collegato.

L'indirizzo \$8800, della ISR legata all'interruzione di livello 4 dovuta alla trasmissione dell'usart (su TxRDY) è in ROM all'indirizzo \$70.

ISR INTERRUZIONE LIVELLO 4	\$8800
----------------------------------	--------

Di seguito è rappresentato l'indirizzo in ROM nel quale è memorizzato l'indirizzo della ISR che gestisce l'interruzione.

ROM	Contenuto
\$70	\$8800

#### 4.1.3 Area Dati

L'area dati inizia dall'indirizzo \$8300 contiene gli N messaggi di lunghezza M da trasmettere. Supposto  $N = 4$  ed  $M = 3$ .

MEX1	\$8300
MEX2	\$8304
MEX3	\$8308



## 4.2 Memoria nodo B

### 4.2.1 Area Periferiche

Per quanto riguarda la mappa di memoria dei device, la USART è mappata su 2 indirizzi consecutivi, \$2002 e \$2003. La PIA è mappata su 4 indirizzi: 2 per il porto A e 2 per il porto B, rispettivamente \$2004-\$2005 e \$2006-\$2007.

USARTP (pari)	\$2002
USARTD (dispari)	\$2003
PIADA	\$2004
PIACA	\$2005
PIADB	\$2006
PIACB	\$2007

### 4.2.2 Area Interruzioni

La modalità di riconoscimento delle interruzioni utilizzata è la modalità autovettorizzata. Per calcolare l'indirizzo nella tabella delle interruzioni bisogna calcolare  $(24 + IPL) \times 4$ , dove IPL è il livello della linea dell'Interrupt Controller al quale il device che genera l'interruzione è collegato.

L'indirizzo \$8700, della ISR legata all'interruzione di livello 3 dovuta alla ricezione dell'usart (su RxRDY) è in ROM all'indirizzo \$6C.

Mentre, l'indirizzo \$8900, della ISR legata all'interruzione di livello 4 dovuta alla ricezione della pia (su IRQA) è in ROM all'indirizzo \$70.

ISR INTERRUZIONE LIVELLO 3	\$8700
ISR INTERRUZIONE LIVELLO 4	\$8900

Di seguito sono rappresentati gli indirizzi in ROM nei quali sono memorizzati gli indirizzi delle ISR che gestiscono le interruzioni.

ROM	Contenuto
\$6C	\$8700
\$70	\$8900

### 4.2.3 Area Dati

L'area dati è definita a partire dall'indirizzo \$8000: il primo byte definisce la variabile FLAG\_T, flag di terminazione, il quale è settato dalla routine di ricezione della PIA quando si ricevono due caratteri dal nodo C. La variabile FLAG\_T è controllata dalla routine di ricezione della USART, poiché se risulta pari ad 1 il nodo B termina la ricezione del messaggio eventualmente in sospeso e poi interrompe la comunicazione con A e C.

Il secondo byte definisce la variabile SEM, la quale definisce la variabile semaforo utilizzata per gestire la Mutua-esclusione.

Il terzo byte definisce la variabile NCRA, ovvero “Numero Caratteri Ricevuti dal nodo A”.

Il quarto byte definisce la variabile NCRC, ovvero “Numero Caratteri Ricevuti dal nodo C”.

Il quinto byte definisce la variabile ATTESA\_A, il quale è setta dalla routine di ricezione della USART, e viene posto ad 1 nel caso in cui debba mettersi in attesa. Viene posto a 0 dalla routine di ricezione della PIA nel caso in cui dovesse essere svegliato.

Il sesto byte definisce la variabile ATTESA\_C, il quale è setta dalla routine di ricezione della PIA, e viene posto ad 1 nel caso in cui debba mettersi in attesa. Viene posto a 0 dalla routine di ricezione della USART nel caso in cui dovesse essere svegliato.

Il settimo byte definisce la variabile NMRA, ovvero “Numero di Messaggi Ricevuti dal nodo A”.

FLAG_T	\$8000
SEM	\$8001
NCRA	\$8002
NCRC	\$8003
ATTESA_A	\$8004
ATTESA_C	\$8005
NMRA	\$8006

### 4.3 Memoria nodo C

#### 4.3.1 Area Periferiche

La PIA è mappata su 4 indirizzi: 2 per il porto A e 2 per il porto B, rispettivamente \$2004-\$2005 e \$2006-\$2007.

PIADA	\$2004
PIACA	\$2005
PIADB	\$2006
PIACB	\$2007

#### 4.3.2 Area Dati

L'area dati inizia dall'indirizzo \$8300 e contiene i caratteri da trasmettere

CAR	\$8300
-----	--------

## 5. Descrizione di alto livello del programma implementato

---

Di seguito è rappresentata la descrizione di alto livello esclusivamente del nodo B, mediante un diagramma di flusso. Nei paragrafi 5.2 e 5.3 vi sono i diagrammi di flusso che descrivono le ISR della versione consegnata allo scritto e nei paragrafi 5.4 e 5.5 ci sono i diagrammi di flusso della versione migliorata.

### 5.1 Main

Di seguito il diagramma di flusso che descrive il main.



*Figura 5.1: Diagramma di flusso main.*

## 5.2 Interruzione ricezione Pia versione 1

Di seguito il diagramma di flusso che descrive la ISR per la ricezione della PIA.

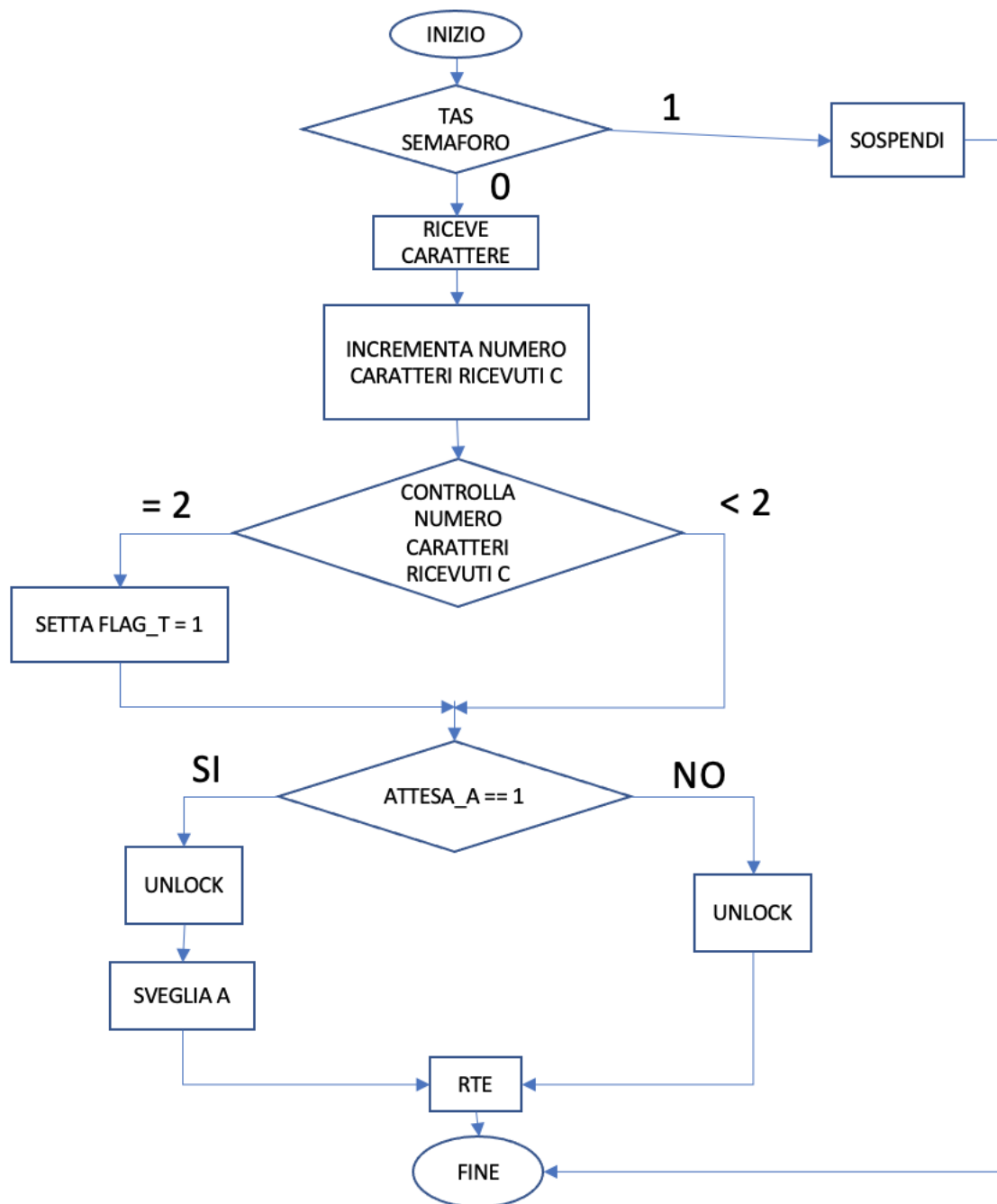


Figura 5.2: Diagramma di flusso ISR ricezione Pia.

### 5.3 Interruzione ricezione Usart versione 1

Di seguito il diagramma di flusso che descrive la ISR per la ricezione della USART.

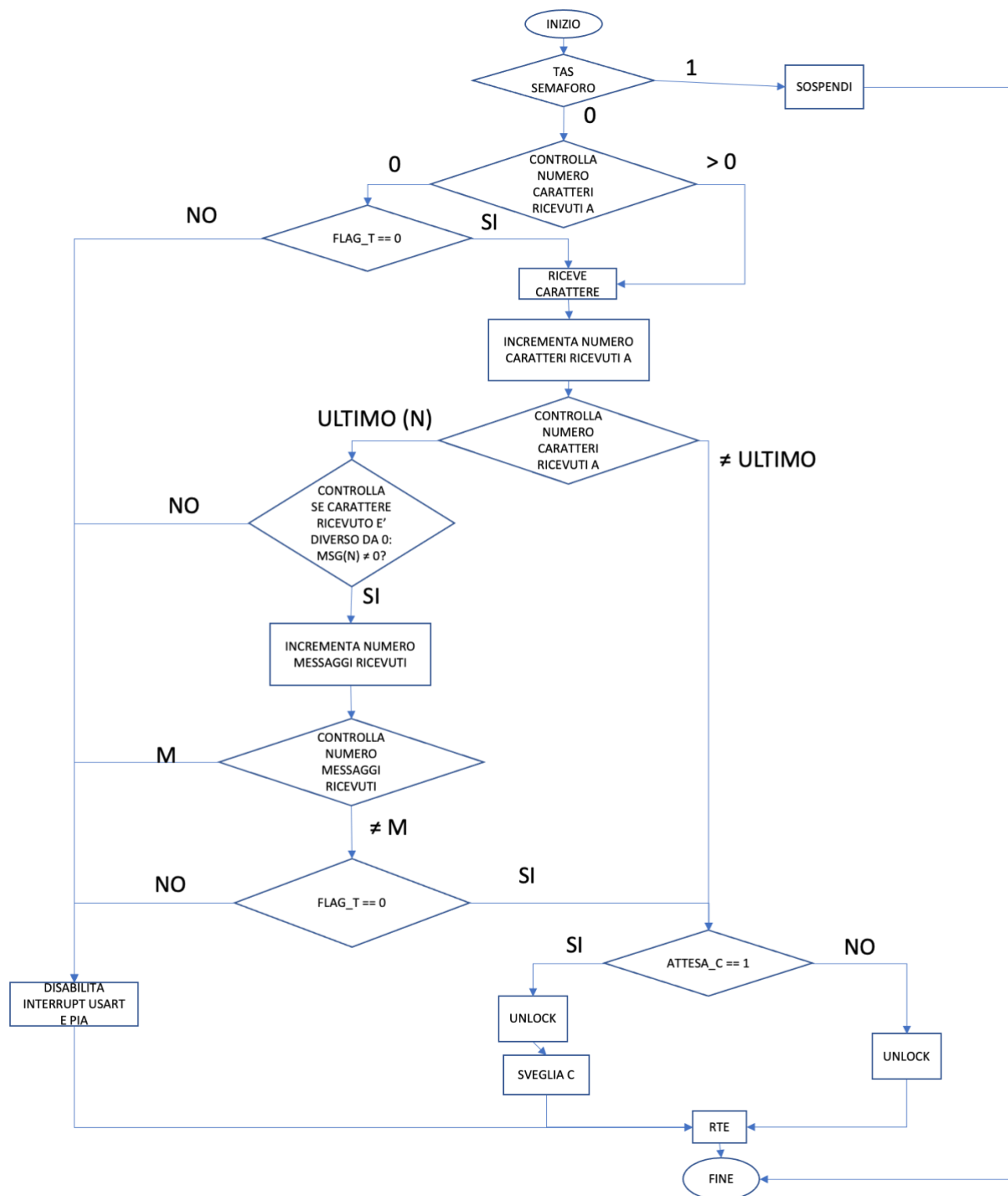


Figura 5.2: Diagramma di flusso ISR ricezione Usart.

## 5.4 Interruzione ricezione Pia versione 2

Nella versione 2 è stata fatta una suddivisione funzionale in subroutine, in particolare, in Figura 5.3 è rappresentata la ISR relativa all'interruzione di livello 3.

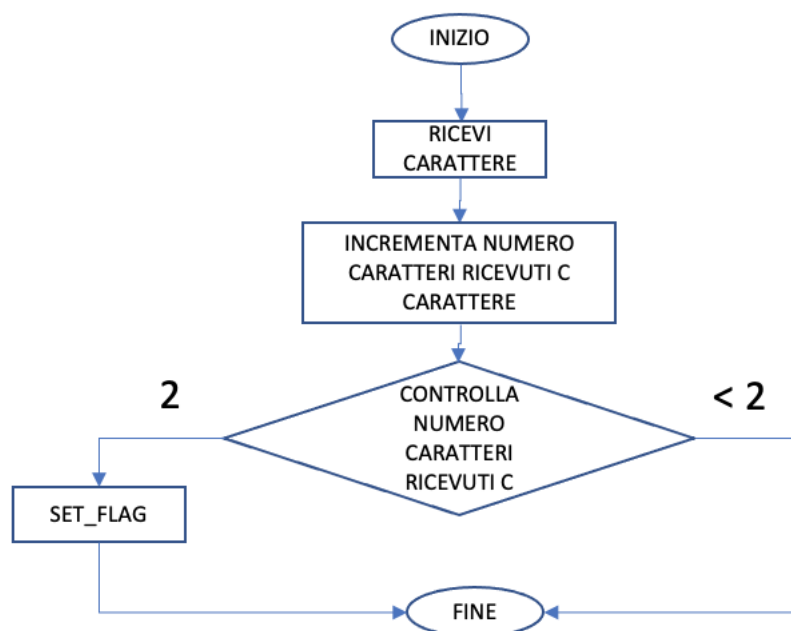


Figura 5.3: Diagramma di flusso ISR ricezione Pia.

In Figura 5.4 è rappresentato il diagramma di flusso relativo alla subroutine SET\_FLAG.

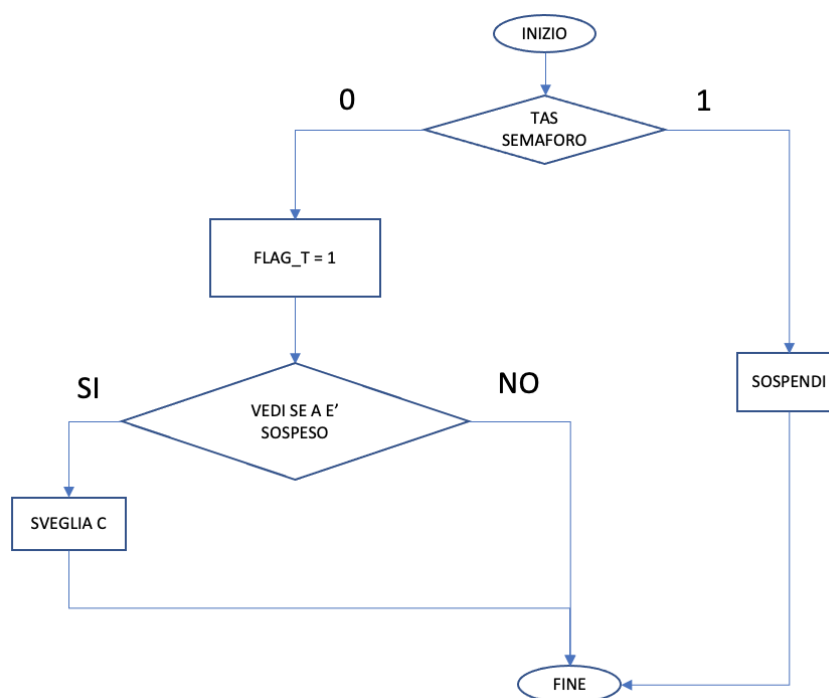
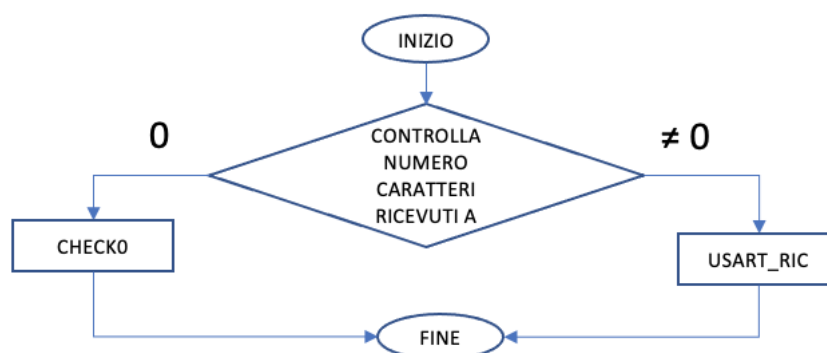


Figura 5.4: Diagramma di flusso subroutine SET\_FLAG.

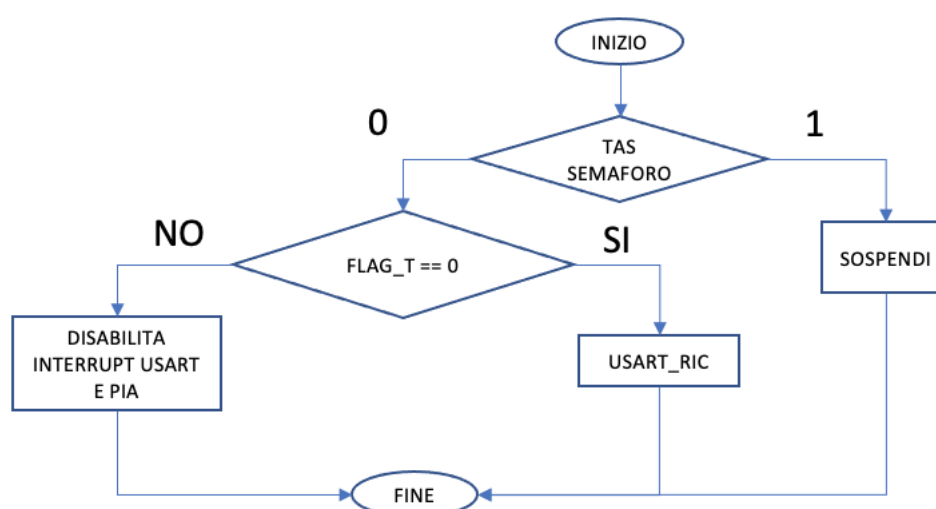
## 5.5 Interruzione ricezione Usart versione 2

Il diagramma in figura 5.5 rappresenta la ISR legata alla interrupt di livello 4. Essa a seconda dei casi chiama la subroutine CHECK0 o la subroutine USART\_RIC.



*Figura 5.5: Diagramma di flusso ISR ricezione Pia versione 2.*

In Figura 5.6 è rappresentato il diagramma di flusso che descrive la subroutine CHECK0.



*Figura 5.6: Diagramma di flusso subroutine CHECK0.*

In Figura 5.7 è rappresentato il diagramma di flusso che descrive la subroutine USART\_RIC.

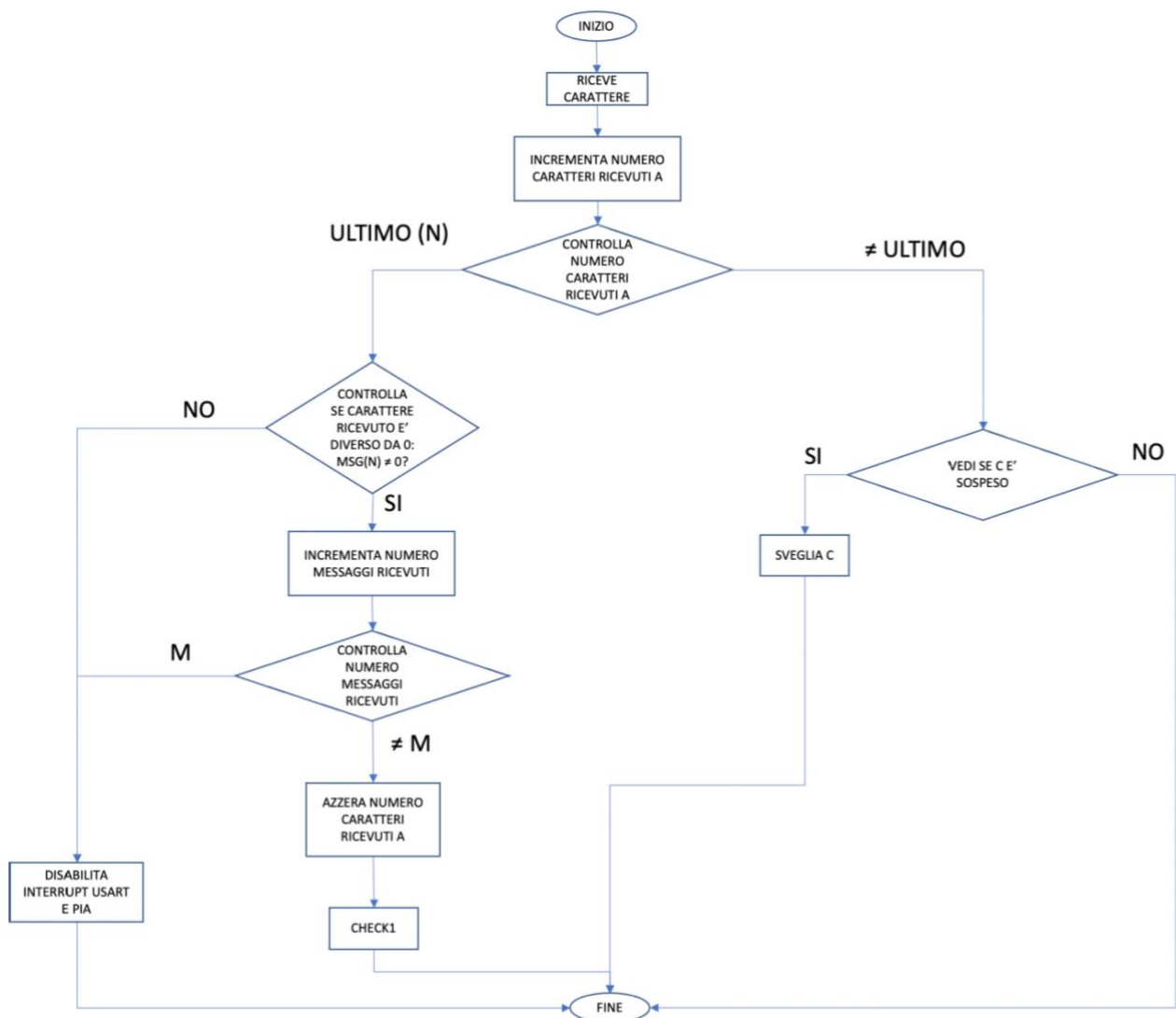
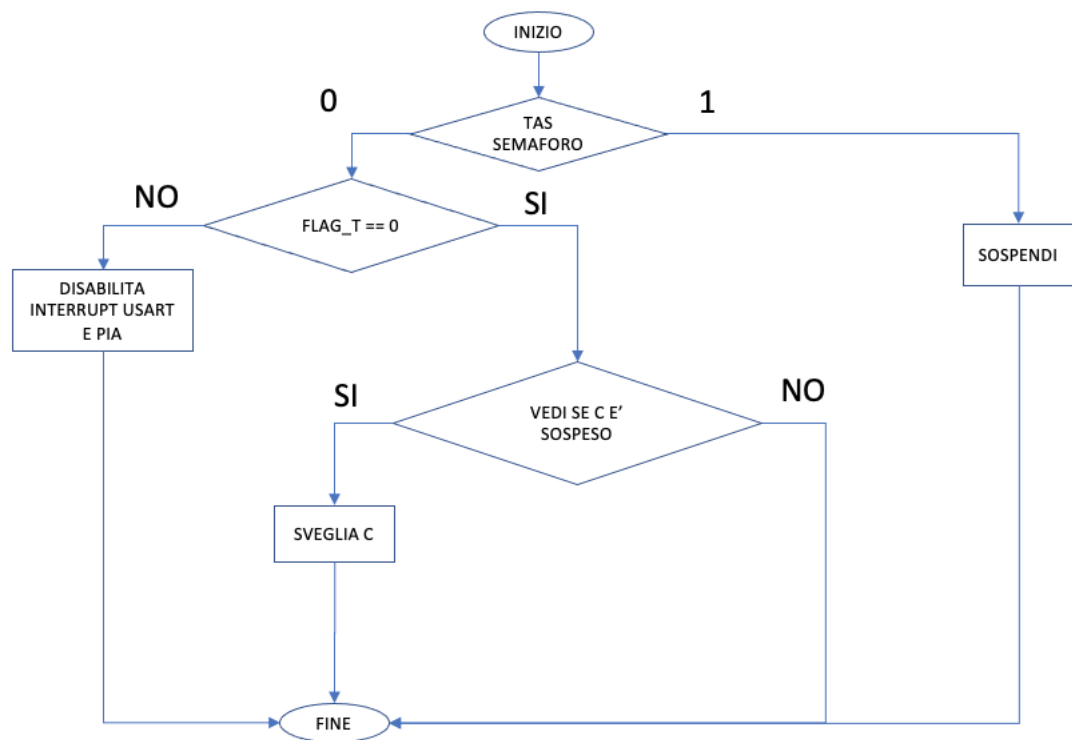


Figura 5.7: Diagramma di flusso subroutine USART\_RIC.



In Figura 5.8 è rappresentato il diagramma di flusso che descrive la subroutine CHECK1.



*Figura 5.8: Diagramma di flusso subroutine CHECK1.*

## 6. Implementazione

---

In questa sezione è presente il codice dell'implementazione. In particolare, nel paragrafo “Codice versione 1” vi è il codice del nodo B generato durante la prova scritta, mentre nel paragrafo “Codice versione 2” vi è il codice migliorato.

### 6.1 Codice versione 1

Di seguito vi è il file di configurazione custom\_3\_nodi\_pia\_usartV1.cfg per generare l'architettura in Asim.

```
CHIP Name: M68000
Type: CPU.          Identif: 01.          BUS: 0002.
Address 1: 00009000. Address 2: 00009200.
Com1: 0000. Com2: 0000. Com3: 0000. Com4: 0000.

CHIP Name: Memory
Type: MMU/BUS.      Identif: 02.          BUS: 0000.
Address 1: 00008000. Address 2: 00000000.
Com1: 0000. Com2: 0010. Com3: 0008. Com4: 0000.

CHIP Name: M6821PIA
Type: Device.       Identif: 03.          BUS: 0002.
Address 1: 00002004. Address 2: 00002007.
Com1: 0001. Com2: 0004. Com3: 0002. Com4: 0207.

CHIP Name: I8251USART
Type: Device.       Identif: 04.          BUS: 0002.
Address 1: 00002002. Address 2: 00002003.
Com1: 0001. Com2: 0003. Com3: 0001. Com4: 000A.

CHIP Name: M68000
Type: CPU.          Identif: 05.          BUS: 0006.
Address 1: 00009000. Address 2: 00009200.
Com1: 0000. Com2: 0000. Com3: 0000. Com4: 0000.

CHIP Name: Memory
Type: MMU/BUS.      Identif: 06.          BUS: 0000.
Address 1: 00008000. Address 2: 00000000.
Com1: 0000. Com2: 0010. Com3: 0008. Com4: 0000.

CHIP Name: M6821PIA
Type: Device.       Identif: 07.          BUS: 0006.
Address 1: 00002004. Address 2: 00002007.
Com1: 0005. Com2: 0004. Com3: 0003. Com4: 0203.

CHIP Name: M68000
Type: CPU.          Identif: 08.          BUS: 0009.
Address 1: 00009000. Address 2: 00009200.
Com1: 0000. Com2: 0000. Com3: 0000. Com4: 0000.

CHIP Name: Memory
Type: MMU/BUS.      Identif: 09.          BUS: 0000.
Address 1: 00008000. Address 2: 00000000.
Com1: 0000. Com2: 0010. Com3: 0008. Com4: 0000.

CHIP Name: I8251USART
Type: Device.       Identif: 0A.          BUS: 0009.
Address 1: 00002002. Address 2: 00002003.
Com1: 0008. Com2: 0003. Com3: 0004. Com4: 0004.
```

Di seguito è indicato il codice del nodo A implementato.

```

MAIN      ORG      $8000
USART     EQU      $2002

N         EQU      4          *LUNGHEZZA MESSAGGIO
M         EQU      3          *NUMERO MESSAGGI DA INVIARE

        JSR      INIT

        *ABILITA INTERRUPT (SMASCHERO 111 DELLE INTERRUZIONI DA SR)
        MOVE.W   SR,D0
        ANDI.W   #$F8FF,D0      *NON PASSA A STATO UTENTE MA RIMANGO IN
SUPERVISORE IN MODO DA INVIARE I MESSAGGI DAL MAIN
        MOVE.W   D0,SR

LOOP      JSR      INVIO
        BRA      LOOP

INVIO     MOVEM.L  D0-D5/A0-A5, -(A7)
        MOVEA.L  #USART,A0
        CLR.L    D2          *CONTATORE NUMERO CARATTERI INVIATI
        CLR.L    D3          *CONTATORE NUMERO MESSAGGI INVIATI
INPUT     MOVEA.L  #MEX1,A1
        MOVE.W   #1,D4
        MULU     #N,D4
        MULU     D3,D4
        ADDA.L   D4,A1
WAIT0     MOVE.B   1(A0),D0
        ANDI.B   #%10000000,D0      *ATTENDE CHE SI ATTIVA DSR
        BEQ      WAIT0
WAIT1     MOVE.B   1(A0),D0
        ANDI.B   #%00000001,D0      *ATTENDE CHE IL TRASFERIMENTO E' AVVENUTO
        (DATAOUT CARICATO TOTALMENTE NELLO SHIFT REGISTER)
        BEQ      WAIT1
        MOVE.B   (A1)+,(A0)
        ADDI.W   #1,D2          *INCREMENTA NUMERO CARATTERI INVIATI
        CMP.W    #N,D2          *CONTROLLA SE HA INVIATO TUTTI I CARATTERI DEL MESSAGGIO
        BNE      WAIT0
        MOVE.W   #0,D2          *AZZERA NUMERO CARATTERI INVIATI
        ADDI.W   #1,D3          *INCREMENTA NUMERO MESSAGGI INVIATI
        CMP.W    #M,D3          *CONTROLLA SE HA INVIATO TUTTI I MESSAGGI
        BNE      INPUT
        MOVEM.L  (A7)+,D0-D5/A0-A5
        RTS

INIT      MOVE.L   A0,-(A7)
        MOVEA.L  #USART,A0
        MOVE.B   #%01011101,1(A0) *ABILITO TRASMISSIONE ASINCORNA CON 8 BIT PER DATO E
        BIT DI PARITA' E DUE BIT DI STOP
        MOVE.B   #%00110011,1(A0) *NON ATTIVO RICEVITORE, ABILITO TRASMETTITORE,
        ATTIVO DTR E RTS E AZZERO STATUS
        MOVE.L   (A7)+,A0
        RTS

*INTERRUZIONE LIV.4 AUTOVET. 24+4 = 28 IN ROM A $70 PUNTA A $8800 LEGATO A TxRDY (QUANDO DA DATAOUT SERIALE
IL VALORE VIENE CARICATO COMPLETAMENTE NELLO SHIFT REGISTER)
INT4      ORG      $8800
        RTE

*AREA DATI
ORG      $8300          *MESSAGGI DA TRASMETTERE
MEX1     DC.B     1,2,3,4
MEX2     DC.B     5,6,7,8
MEX3     DC.B     9,$A,$B,$C
        END      MAIN

```

Di seguito è indicato il codice del nodo B implementato. In particolare, il seguente è il codice sviluppato durante la prova scritta.

```

                                ORG                                $8300                                *area main
MAIN
USARTP                        EQU                                $2002                                *usart pari
USARTD                        EQU                                $2003                                *usart dispari
PIADA                        EQU                                $2004
PIACA                        EQU                                $2005
PIADB                        EQU                                $2006
PIACB                        EQU                                $2007
                                JSR                                INIT
                                MOVE.W                            SR,D0
                                ANDI.W                            #$D8FF,D0                                *smaschera interrupt
                                MOVE.W                            D0,SR
LOOP
                                BRA                                LOOP
INIT
                                MOVE.B                            #0,PIACA
                                MOVE.B                            #$00,PIADA                                *inizializza PIA porto A
                                MOVE.B                            #%00100101,PIACA                                *inizializzazione usart
                                MOVE.B                            #%01011101,USARTD                                *modo usart
                                MOVE.B                            #%00110110,USARTD                                *controllo usart
                                RTS

*ISR PER INTERRUPT LIVELLO 3 LEGATA ALLA RICEZIONE SULLA USART, AUTOVETTORE 27 = ((24 + 3)*4).
*IN ROM A $6C, PUNTA AD INDIRIZZO $8700
                                ORG                                $8700
INT3
                                JSR                                ISR_USART_RIC
                                RTE

ISR_USART_RIC
                                TAS                                SEM
                                BEQ                                CONTIN
                                MOVE.B                            #1,ATTESA_A
                                RTS
CONTIN
                                TST.B                            NCRA                                *se non ha ricevuto
                                neanche un carattere controlla il FLAG_T, altrimenti
                                BNE                                PROS                                *è in ricezione e deve
                                terminare la ricezione dell'intero messaggio
                                TST.B                            FLAG_T
                                BEQ                                PROS
                                MOVE.B                            #%00100100,PIACA                                *nel caso in cui è diverso da
                                0 disabilita le interrupt
                                MOVE.B                            #$FF,USARTD                                *disabilita USART
                                RTS
PROS
                                MOVEM.L                            D0-D5/A0-A5,-(A7)                                *salva registri che "sporca"
                                MOVEA.L                            #USARTP,A0
                                MOVE.B                            1(A0),D3                                *controlla se si è verificato
                                qualche errore e in caso affermativo ripristina
                                ANDI.B                            #$38,D3                                *i flag di errore dal registro di stato
                                BEQ                                NOERR
                                MOVE.B                            #$37,1(A0)
                                NOERR
                                MOVE.B                            (A0),D0                                *il carattere è ricevuto ma non lo memorizza
                                MOVE.B                            NCRA,D1
                                ADDI.B                            #1,D1
                                MOVE.B                            D1,NCRA
                                CMP.B                            #N,D1
                                BNE                                ALTRIM
                                CMP.B                            #0,D0
                                BNE                                AVANTI2
                                MOVEM.L                            (A7)+,D0-D5/A0-A5
                                MOVE.B                            #%00100100,PIACA
                                MOVE.B                            #$FF,USARTD
                                RTS
AVANTI2
                                MOVE.B                            NMRA,D2
                                ADDI.B                            #1,D2
                                MOVE.B                            D2,NMRA
                                CMP.B                            #M,D2
                                BNE                                ALTR2
                                MOVEM.L                            (A7)+,D0-D5/A0-A5
                                MOVE.B                            #%00100100,PIACA
                                MOVE.B                            #$FF,USARTD

```

ALTR2	RTS		
	MOVE.B	#0,NCRA	*azzera il numero di caratteri ricevuti da A
	TST.B	FLAG_T	
	BEQ	PROS2	
	MOVEM.L	(A7)+,D0-D5/A0-A5	
	MOVE.B	##00100100,PIACA	
	MOVE.B	##\$FF,USARTD	
	RTS		
PROS2	TST.B	ATTESA_C	
	BEQ	TERMINA	
SVEGC			*C è in attesa è viene svegliato
	MOVEM.L	(A7)+,D0-D5/A0-A5	
	MOVE.B	#0,ATTESA_C	
	MOVE.B	#0,SEM	
	JSR	ISR_PIA_RIC	
	RTS		
TERMINA	MOVEM.L	(A7)+,D0-D5/A0-A5	
	MOVE.B	#0,SEM	
	RTS		
ALTRIM	TST.B	ATTESA_C	
	BEQ	TERMINA	
	BRA	SVEGC	
*ISR PER INTERRUPT LIVELLO 4 LEGATA ALLA RICEZIONE SULLA PIA, AUTOVETTORE 28 = ((24 + 4)*4).			
*IN ROM A \$70, PUNTA AD INDIRIZZO \$8900			
	ORG	\$8900	
INT4	JSR	ISR_PIA_RIC	
	RTE		
ISR_PIA_RIC	TAS	SEM	
	BEQ	CONT4	
	MOVE.B	#1,ATTESA_C	*C si mette in attesa
	RTS		
CONT4	MOVEM.L	D0-D5/A0-A5,-(A7)	
	MOVEA.L	#PIADA,A0	
	MOVE.B	(A0),D0	*riceve ma non memorizza
	MOVE.B	NCRC,D1	
	ADDI.B	#1,D1	
	MOVE.B	D1,NCRC	
	CMP.B	#2,D1	
	BNE	AT4	
	MOVE.B	#1,FLAG_T	*mette FLAG_T ad 1
	TST.B	ATTESA_A	
	BNE	WKUP	
AT4	MOVEM.L	(A7)+,D0-D5/A0-A5	
	MOVE.B	#0,SEM	*unlock
	RTS		
WKUP	MOVEM.L	(A7)+,D0-D5/A0-A5	
	MOVE.B	#0,ATTESA_A	
	MOVE.B	#0,SEM	
	JSR	ISR_USART_RIC	
	RTS		
	ORG	\$8000	*area dati
M	EQU	3	*numero messaggi
N	EQU	4	*lunghezza messaggio
FLAG_T	DC.B	0	*flag termina
SEM	DC.B	0	
NCRA	DC.B	0	*numero di caratteri ricevuti da A
NCRC	DC.B	0	*numeri di caratteri ricevuti da C
ATTESA_A	DC.B	0	
ATTESA_C	DC.B	0	
NMRA	DC.B	0	*numero di messaggi ricevuti da A
	END	MAIN	

Di seguito è indicato il codice del nodo C implementato.

	ORG	\$8000	
MAIN			
PIADA	EQU	\$2004	
PIACA	EQU	\$2005	
PIADB	EQU	\$2006	
PIACB	EQU	\$2007	
	JSR	INIZ	
	MOVE.W	SR,D0	
utente	ANDI.W	#\$F8FF,D0	*F anche al più significativo perchè non passo a stato
	MOVE.W	D0,SR	
	JSR	INVIO	
LOOP	BRA	LOOP	
INIZ	MOVE.B	#0,PIACB	
	MOVE.B	#\$FF,PIADB	
	MOVE.B	##00100100,PIACB	
	RTS		
INVIO	MOVEM.L	A0-A2/D0-D4, -(A7)	
	MOVEA.L	#PIADB,A1	
	MOVEA.L	#PIACB,A2	
	MOVEA.L	#CAR,A0	
	CLR.L	D1	
INPUT	CLR.L	D3	
*ATTESA INVIO CARATTERE			
CICLO	ADDI.W	#1,D3	
	CMP.W	#10,D3	
	BNE	CICLO	
	MOVE.B	(A1),D0	*lettura fittizia
	MOVE.B	\$0(A0,D1),D2	
	MOVE.B	D2,(A1)	
CICLO2	MOVE.B	(A2),D4	
	ANDI.B	#\$80,D4	
	BEQ	CICLO2	
	ADDI.W	#1,D1	
	CMP.W	#M,D1	
	BNE	INPUT	
	MOVEM.L	(A7)+,A0-A2/D0-D4	
	RTS		
	ORG	\$8300	
*AREA DATI			
M	EQU	2	
CAR	DC.B	1,2	
	END	MAIN	

## 6.2 Codice versione 2

Di seguito vi è il codice del nodo 2 migliorato rispetto al precedente. I principali cambiamenti sono: sono state create subroutine per gestire frammenti di codice che si ripetevano con maggiore frequenza, ad esempio la disabilitazione delle interrupt della pia e della usart. È stato fatto il controllo sugli errori in ricezione con la usart. Si è gestita la mutua-esclusione in modo più puntuale, ovvero non è stata utilizzata l'istruzione TAS per eseguire in mutua-esclusione le intere ISR di ricezione, ma è stata utilizzata solo per accedere all'effettiva zona critica, la quale presenta variabili che vanno gestite in mutua-esclusione, ovvero: FLAG\_T.

```

MAIN          ORG          $8300          *area main
USARTP        EQU          $2002          *usart pari
USARTD        EQU          $2003          *usart dispari
PIADA         EQU          $2004
PIACA         EQU          $2005
PIADB         EQU          $2006
PIACB         EQU          $2007
              JSR          INIT
              MOVE.W       SR,D0
              ANDI.W       #$D8FF,D0          *smaschera interrupt
              MOVE.W       D0,SR
LOOP          BRA          LOOP

INIT          MOVE.B       #0,PIACA
              MOVE.B       #$00,PIADA          *inizializza PIA porto A
              MOVE.B       #%00100101,PIACA
*inizializzazione usart
              MOVE.B       #%01011101,USARTD          *modo usart
              MOVE.B       #%00110110,USARTD          *controllo usart
              RTS

*ISR PER INTERRUPT LIVELLO 3 LEGATA ALLA RICEZIONE SULLA USART, AUTOVETTORE 27 = ((24 + 3)*4).
*IN ROM A $6C, PUNTA AD INDIRIZZO $8700
              ORG          $8700
INT3          TST.B        NCRA          *se non ha ricevuto neanche un
carattere controlla il FLAG_T, altrimenti
              BNE          PROS          *è in ricezione e deve terminare la
              ricezione dell'intero messaggio
              JSR          CHECK0
              RTE
PROS          JSR          USART_RIC
              RTE

CHECK0        TAS          SEM
              BEQ          CONTIN
              MOVE.B       #1,ATT_A_0
              RTS
CONTIN        TST.B        FLAG_T
              BEQ          PROS0
              JSR          DISABILITA
              MOVE.B       #0,SEM
              RTS
PROS0         MOVE.B       #0,SEM
              JSR          USART_RIC
              RTS

DISABILITA    MOVE.B       #%00100100,PIACA          *disabilita le interrupt pia
              MOVE.B       #$FF,USARTD          *disabilita USART
              RTS

USART_RIC     MOVEM.L       D0-D5/A0-A5,-(A7)          *salva registri che "sporca"
              MOVEA.L      #USARTP,A0

```

	MOVE.B	1(A0),D3	*controlla se si è verificato qualche
errore e in caso affermativo ripristina	ANDI.B	#\$38,D3	*i flag di errore dal registro di stato
	BEQ	NOERR	
	MOVE.B	#\$37,1(A0)	
NOERR	MOVE.B	(A0),D0	*il carattere è ricevuto ma non lo memorizza
	MOVE.B	NCRA,D1	
	ADDI.B	#1,D1	
	MOVE.B	D1,NCRA	
	CMP.B	#N,D1	
	BNE	ALTRIM	
	CMP.B	#0,D0	
	BNE	AVANTI2	
	MOVEM.L	(A7)+,D0-D5/A0-A5	
	JSR	DISABILITA	
	RTS		
AVANTI2	MOVE.B	NMRA,D2	
	ADDI.B	#1,D2	
	MOVE.B	D2,NMRA	
	CMP.B	#M,D2	
	BNE	ALTR2	
	MOVEM.L	(A7)+,D0-D5/A0-A5	
	JSR	DISABILITA	
	RTS		
ALTR2	MOVE.B	#0,NCRA	*azzerà il numero di caratteri ricevuti da A
	MOVEM.L	(A7)+,D0-D5/A0-A5	
	JSR	CHECK1	
	RTS		
ALTRIM	MOVEM.L	(A7)+,D0-D5/A0-A5	
	TAS	SEM	
	BEQ	GOON	
	RTS		
GOON	JSR	CHECKC	
	RTS		
	TAS	SEM	
CHECK1	BEQ	CNTN	
	MOVE.B	#0,ATT_A_1	
	RTS		
CNTN	TST.B	FLAG_T	
	BEQ	PROS2	
	JSR	DISABILITA	
	RTS		
PROS2	JSR	CHECKC	
	RTS		
CHECKC	TST.B	ATT_C	
	BEQ	TERMINA	
SVEGC			*C è in attesa è viene svegliato
	MOVE.B	#0,ATT_C	
	MOVE.B	#0,SEM	
	JSR	SET_FLAG	
	RTS		
TERMINA	MOVE.B	#0,SEM	
	RTS		
*ISR PER INTERRUPT LIVELLO 4 LEGATA ALLA RICEZIONE SULLA PIA, AUTOVETTORE 28 = ((24 + 4)*4).			
*IN ROM A \$70, PUNTA AD INDIRIZZO \$8900			
	ORG	\$8900	
INT4	MOVEM.L	D0-D5/A0-A5, -(A7)	
	MOVEA.L	#PIADA,A0	
	MOVE.B	(A0),D0	*riceve ma non memorizza
	MOVE.B	NCRC,D1	
	ADDI.B	#1,D1	
	MOVE.B	D1,NCRC	
	CMP.B	#2,D1	
	BNE	AT4	
	JSR	SET_FLAG	
AT4	MOVEM.L	(A7)+,D0-D5/A0-A5	
	RTE		
	TAS	SEM	
SET_FLAG	BEQ	CONT4	



	MOVE.B	#1,ATT_C	*C si mette in attesa
	RTS		
CONT4	MOVE.B	#1,FLAG_T	*mette FLAG_T ad 1
	TST.B	ATT_A_0	
	BNE	WKUP0	
	TST.B	ATT_A_1	
	BNE	WKUP1	
	MOVE.B	#0,SEM	
	RTS		
WKUP0	MOVE.B	#0,ATT_A_0	
	MOVE.B	#0,SEM	
	JSR	CHECK0	
	RTS		
WKUP1	MOVE.B	#0,ATT_A_1	
	MOVE.B	#0,SEM	
	JSR	CHECK1	
	RTS		
	ORG	\$8000	*area dati
M	EQU	3	*numero messaggi
N	EQU	4	*lunghezza messaggio
FLAG_T	DC.B	0	*flag termina
SEM	DC.B	0	
NCRA	DC.B	0	*numero di caratteri ricevuti da A
NCRC	DC.B	0	*numeri di caratteri ricevuti da C
ATT_A_0	DC.B	0	
ATT_A_1	DC.B	0	
ATT_C	DC.B	0	
NMRA	DC.B	0	*numero di messaggi ricevuti da A
	END	MAIN	