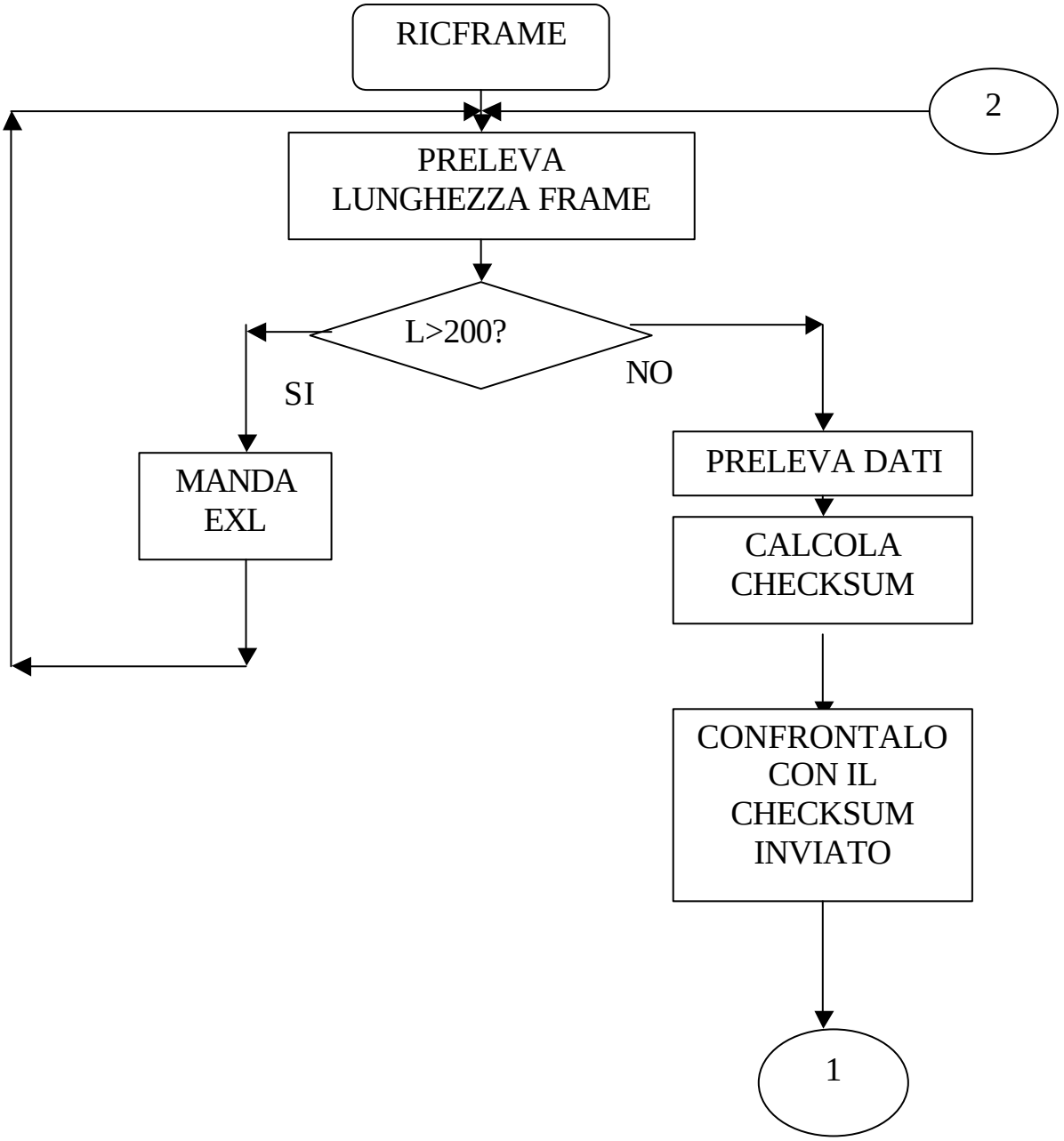
# Esercizio 1 bis

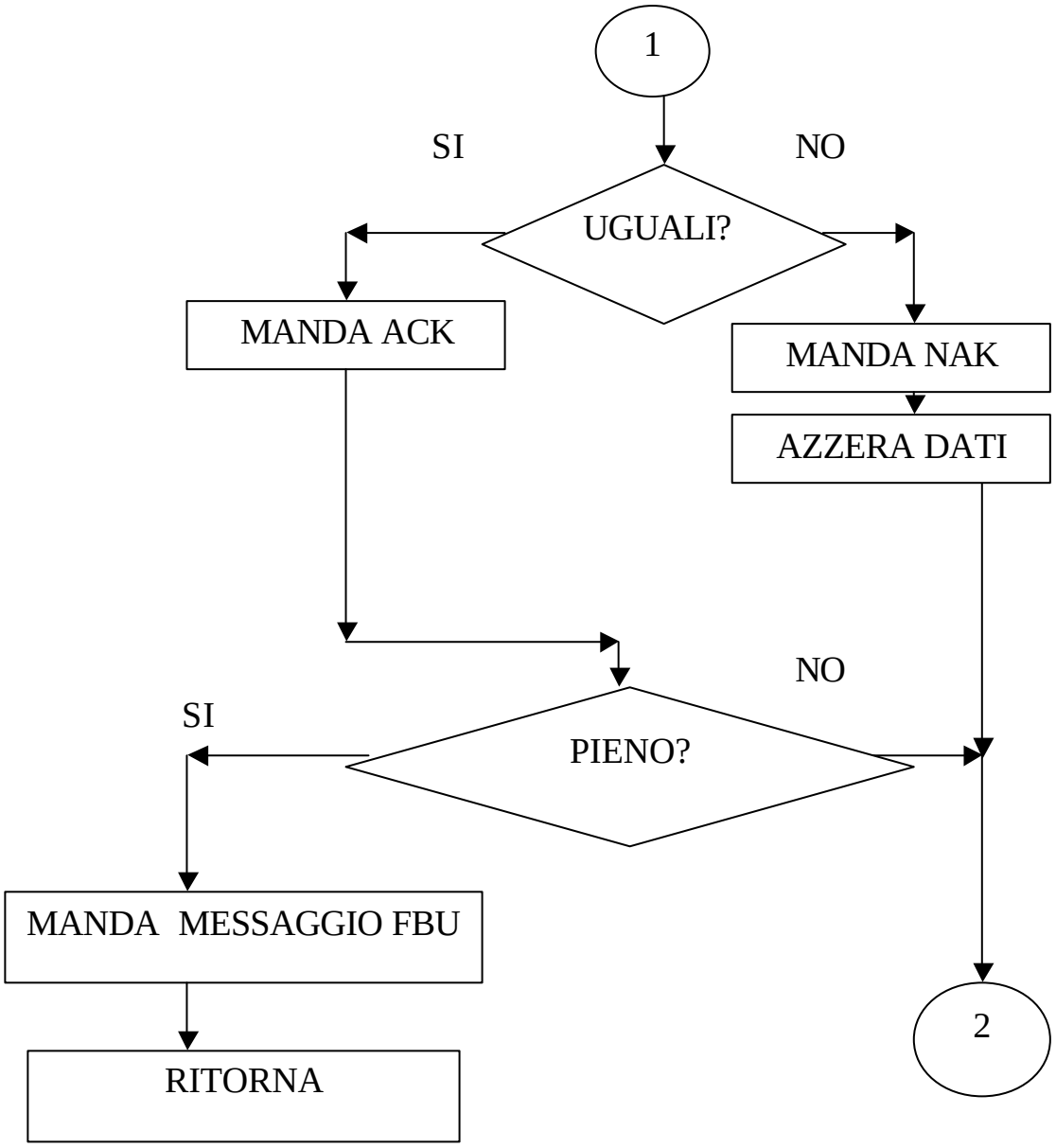
## *Traccia*

Vedi Esercizio 1

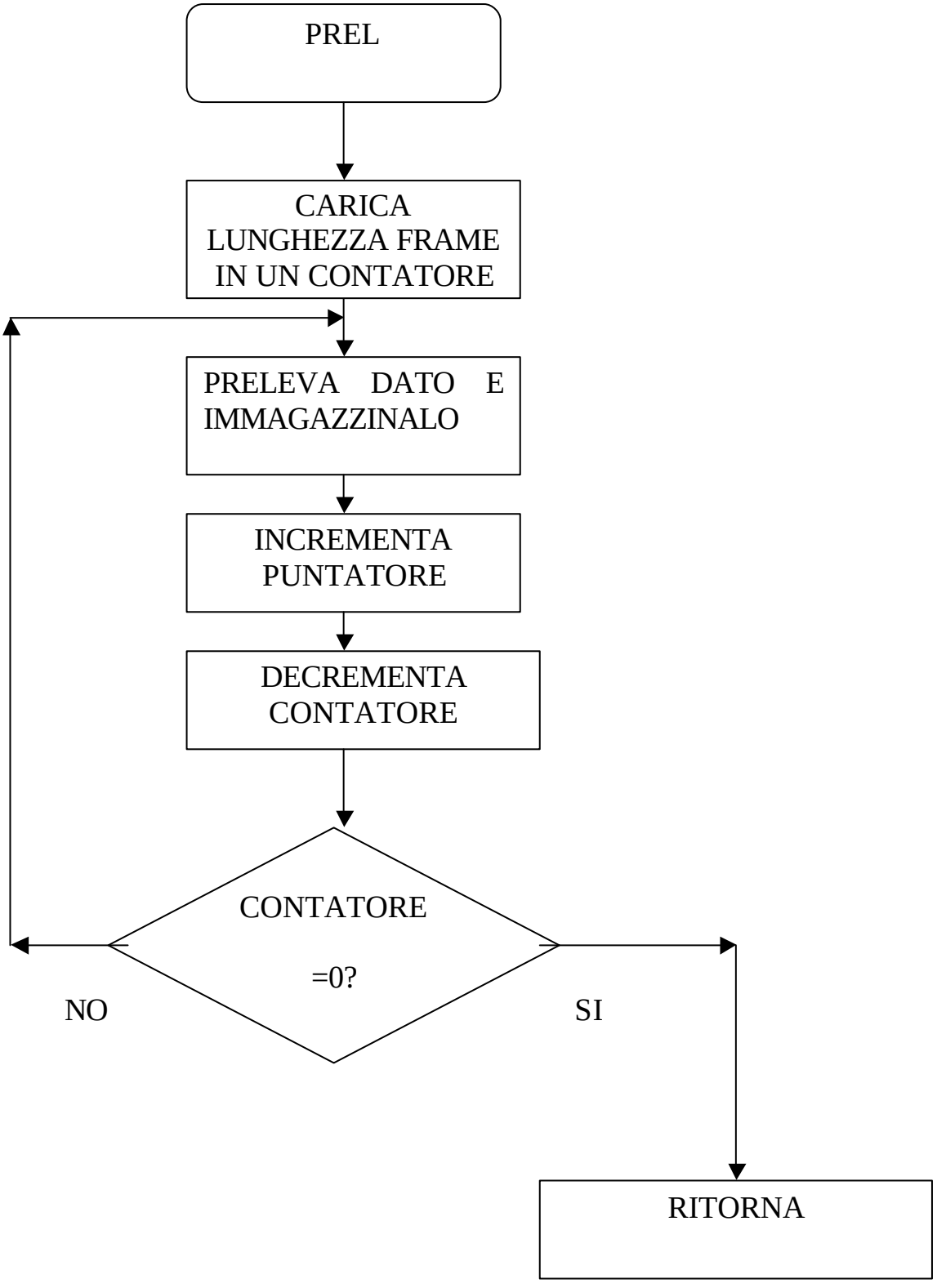
## *Analisi*

Si può migliorare il programma precedente se si nota, analizzando il flusso del programma principale, che, nel caso in cui il checksum calcolato dal microprocessore ricevitore sia diverso da quello del mittente, si svuota il buffer del pacchetto inviato. In tal caso è impossibile che abbiamo raggiunto la condizione di riempimento del buffer, per cui, in luogo di procedere anche in questo caso ad un inutile controllo sullo stato di riempimento del buffer,si ritorna all'inizio del programma per prelevare un altro buffer. Un'altra modifica utile consiste nel fatto che possiamo evitare di svuotare il buffer scrivendo nelle locazioni occupate dal pacchetto corrotto il dato 00H. Se riportiamo semplicemente il puntatore indietro, rendiamo automaticamente quelle locazioni disponibili e permettiamo che siano soprascritte dal nuovo pacchetto. Di seguito abbiamo i diagrammi di flusso e il listato corretto.

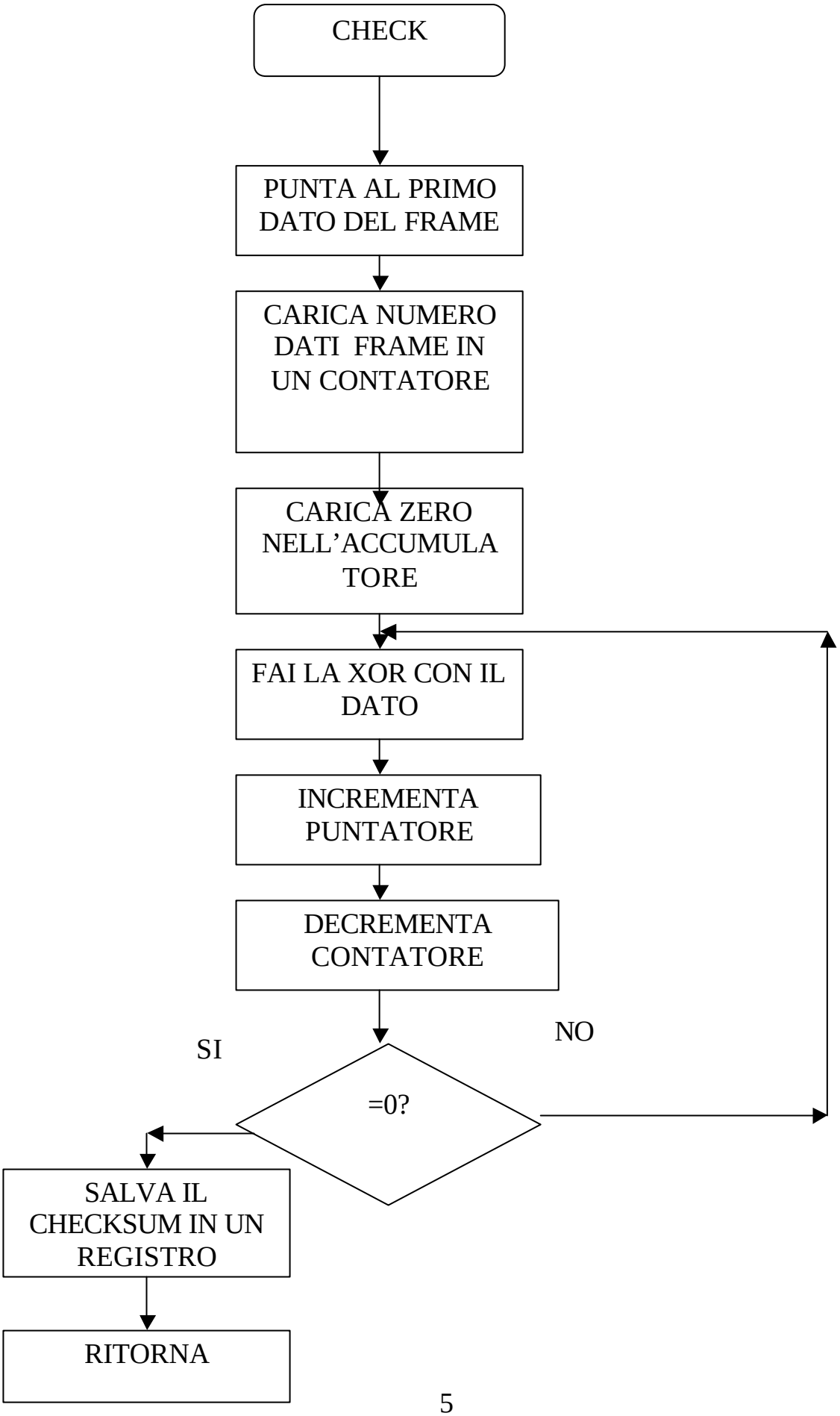

RICFRAME
2
PRELEVA
LUNGHEZZA FRAME
L>200?
SI
NO
MANDA
EXL
PRELEVA DATI
CALCOLA
CHECKSUM
CONFRONTALO
CON IL
CHECKSUM
INVIATO
1




Il blocco preleva dati può essere realizzato con un altro sottoprogramma PREL che può essere rappresentato dal seguente diagramma




Il blocco calcola checksum può essere realizzato con un altro sottoprogramma CHECK


CHECK
PUNTA AL PRIMO DATO DEL FRAME
CARICA NUMERO DATI FRAME IN UN CONTATORE
CARICA ZERO NELL'ACCUMULA TORE
FAI LA XOR CON IL DATO
INCREMENTA PUNTATORE
DECREMENTA CONTATORE
SI
NO
=0?
SALVA IL CHECKSUM IN UN REGISTRO
RITORNA

### *Codifica programma*

```
RICFRAME: LD HL,1A00H; punta all'inizio del buffer
ET:       IN A,(30h); leggi la lunghezza del frame contenuta nel primo byte
          CP C9h;controlla se la lunghezza è superiore a 200, pari a C8 hex
          JP N,ET1;se la lunghezza del frame è inferiore salta le istruzioni
                  ;che inviano il messaggio EXL
          LD A,A3h; carica in A il codice di EXL
          OUT (30h),A;invialo alla porta
          JP ET;torna all'inizio
ET1:      CALL PREL; se la lunghezza è giusta preleva i dati,
                   ;il sottoprogramma deve lasciare A inalterato poiché
                   ; la lunghezza del frame
          CALL CHECK;al termine del sottoprogramma HL punterà
                    ; alla locazione che contiene il checksum inviato,
                    ; imponiamo che lasci in D il checksum
                     ;calcolato
         LD E,A;salva A che contiene la lunghezza del frame
         LD A,D;poni in A il checksum calcolato
         CP (HL);confronta con il checksum calcolato
         JP Z,ET2;salta le istruzioni che mandano il messaggio NAK
          LD A,FEh; inserisci in A il valore di NAK
         OUT(30h),A
```

```
        LD A,E; recupera la lunghezza del frame
        CALL AZZERA
        JP ET;si ritorna all'inizio attendendo l'invio di un checksum corretto
ET2:    LD A,FFh ;carica in A il codice ACK
        OUT(30h),A
ET3:    AND A; questa istruzione non modifica l'accumulatore ma serve
                ;soltanto per azzerare il carry
        LD BC,410Fh;il buffer inizia con la locazione 1A00, e finisce alla
                        ; 410Fh poiché 10000 in esadecimale è 2710h da cui
                        ; la locazione finale è 1A00+2710-1=410Fh
                        ; il sottoprogramma CHECK dovrà fare in modo che HL
                        ; punti al checksum dell'ultimo frame inserito quindi , se
                        ; è l'ultimo frame inserito HL dovrà contenere 410F
        SBC HL,BC;questa è l'unica istruzione di sottrazione a 16 bit,
                        ;perciò abbiamo azzerato il carry
        JP Z,FINE;in questo caso vai alla fine del sottoprogramma
        ADD HL,BC;altrimenti ripristina HL
        INC HL;incrementalo in modo da puntare alla prima locazione vuota
        JP ET; e ritorna all'inizio per prelevare un nuovo frame
FINE:   LD A,FEh;invia il messaggio di buffer pieno
        OUT(30h),A
        RET
```

```
PREL:   LD B,A;salva nel contatore B la lunghezza del frame
            ; HL contiene già l'indirizzo della prima locazione del buffer
           : da riempire
        INIR
        RET
CHECK:  LD C,A
        LD B,00h;abbiamo inserito nel registro BC la lunghezza del frame
        AND A;azzera il carry
        SBC HL,BC;il sottoprogramma PREL ha portato HL più avanti di una
                  ; posizione per cui se sottraggo la lunghezza del frame
                  ;mi porto al primo byte del frame
        INC HL;mi devo portare al secondo byte che è il primo dei dati
        LD B,C;uso B come contatore
        DEC B;deve essere pari alla lunghezza del frame meno uno
            ;poiché non dovremo fare la XOR anche con il byte di checksum
            ;inviato dal trasmettitore
        LD E,A;salvo in E la lunghezza del frame
        LD A,00h;A conterrà il checksum
ET4:    XOR (HL)
        INC HL
        DJNZ ET4
        LD D,A ;salva il checksum calcolato
```

```
        LD A,E;salva di nuovo in A la lunghezza del frame
        RET
AZZERA:  LD C,A
;il sottoprogramma CHECK ha fatto in modo che HL punti
;all'ultimo byte del frame, ad esempio se si trattava del primo pacchetto e questo era
;composto di 100 dati il buffer era stato riempito da 1A00H a 1A63H, per cui dopo
;CHECK il registro HL contiene 1A63H
;per tornare indietro basta sottrarre ad HL la lunghezza del frame (64 in esadecimale)
meno 1
        DEC C; lunghezza del frame meno uno
          LD B,00h ; facciamo in modo che BC contenga la lunghezza del frame
                   ;con la parte alta a zero
        AND A ; azzeriamo il flag di carry poiché dobbiamo usare SBC
        SBC HL,BC
        RET
```