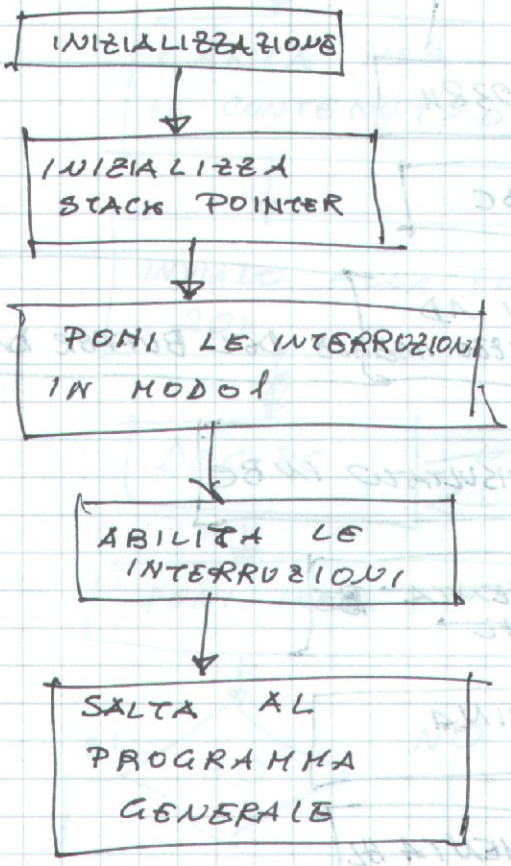
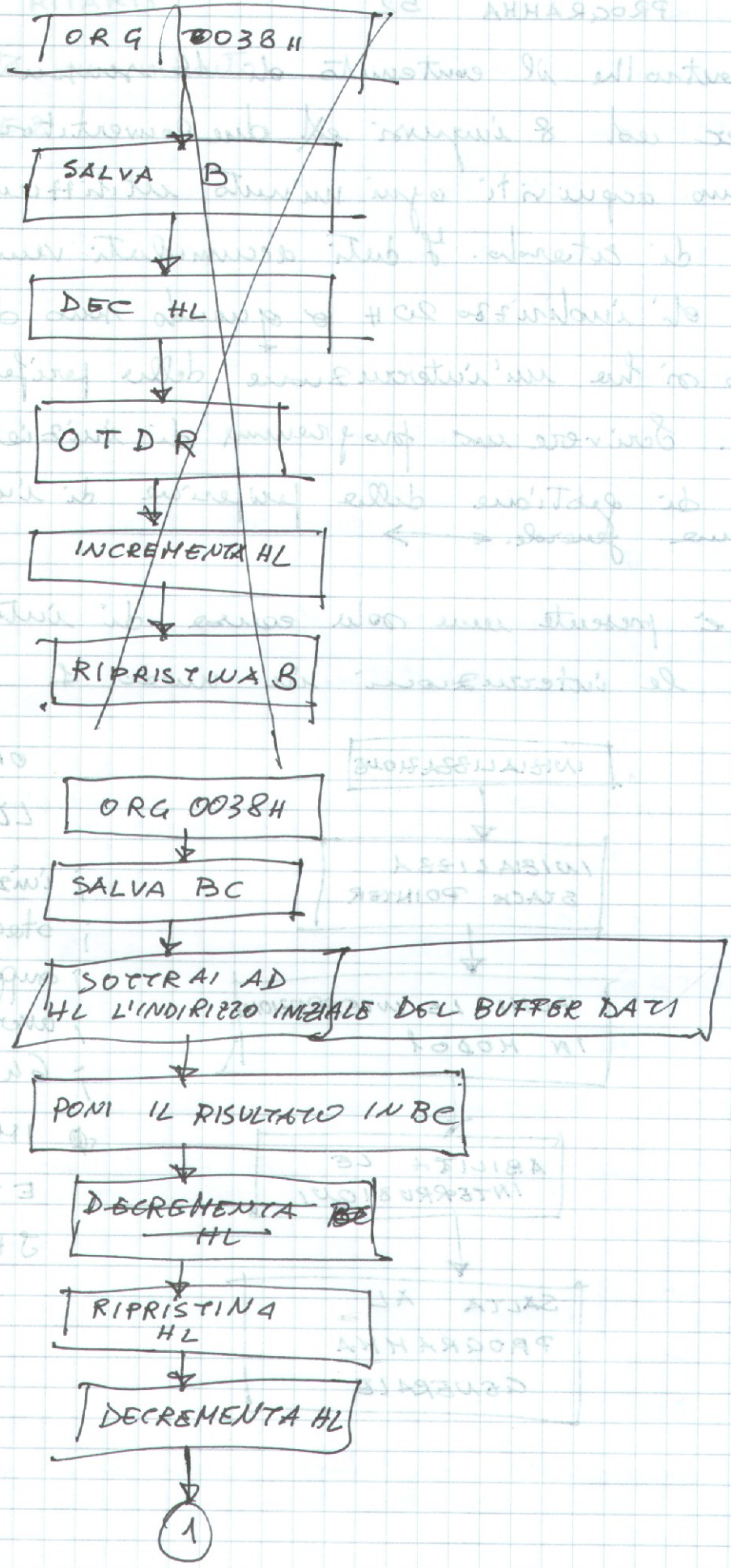
PROGRAMMA 52

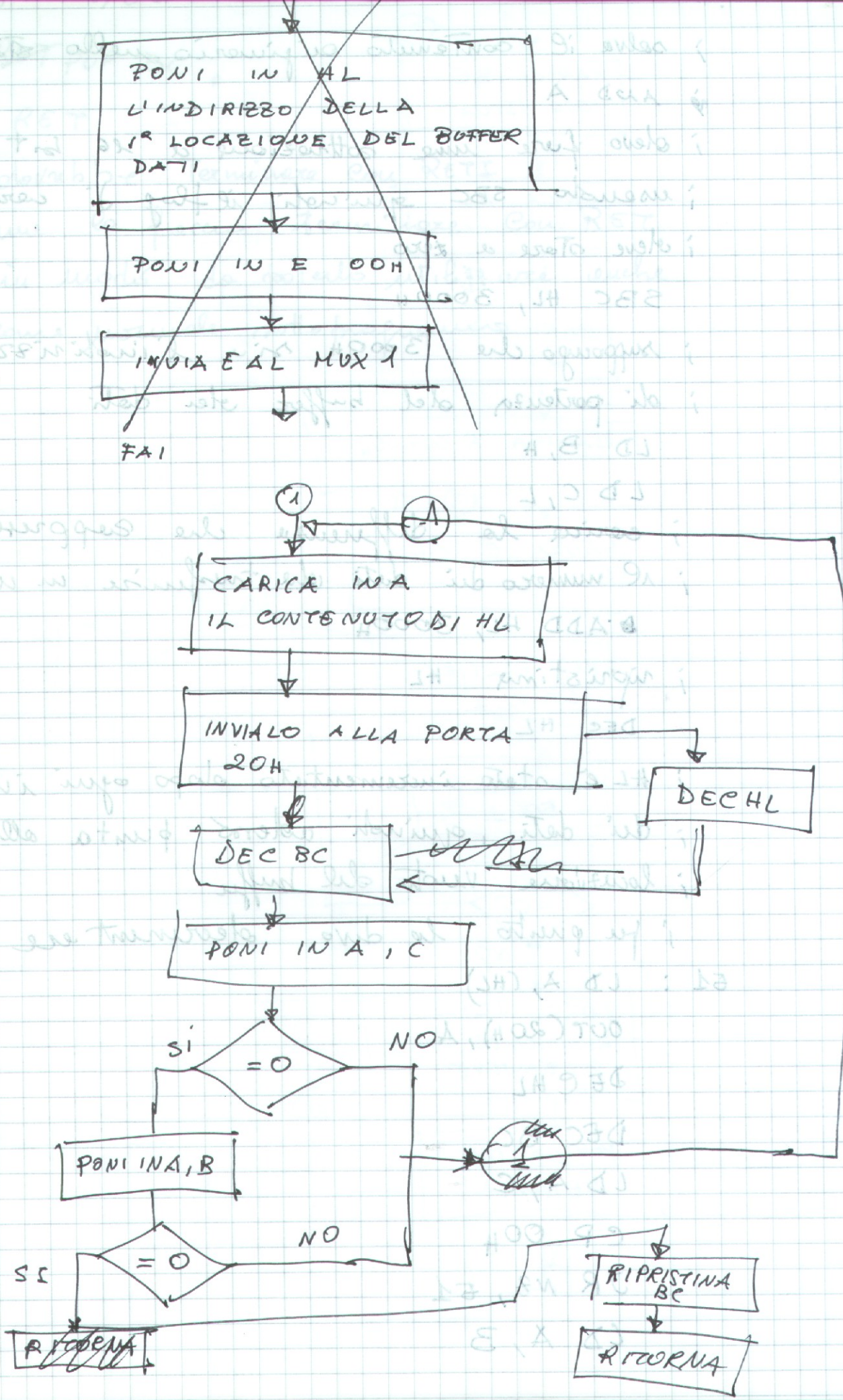
Un µP controlla il contenuto di 16 sensori attraverso due multiplexer ad 8 ingressi ed due convertitori A/D. I dati vengono acquisiti ogni minuto utilizzando un sottoprogramma di ritardo. I dati accumulati vengono spediti alla porta di indirizzo 20H e quando sono diventati 1600 e quando si ha un'interruzione della periferica di indirizzo 30H. Scrivere un programma di inizializzazione e programma di gestione della periferica di indirizzo 30H. e programma generale. ←→

Poiché è presente una sola causa di interruzione usiamo le interruzioni in modo 1




```
ORG  0000H
LD SP, 0FFFFH
; inizializzo lo
; stack pointer
; supponendo di
; avere tutti i
; 64 K di memoria
IM 1
EI
JP SENSORI
```

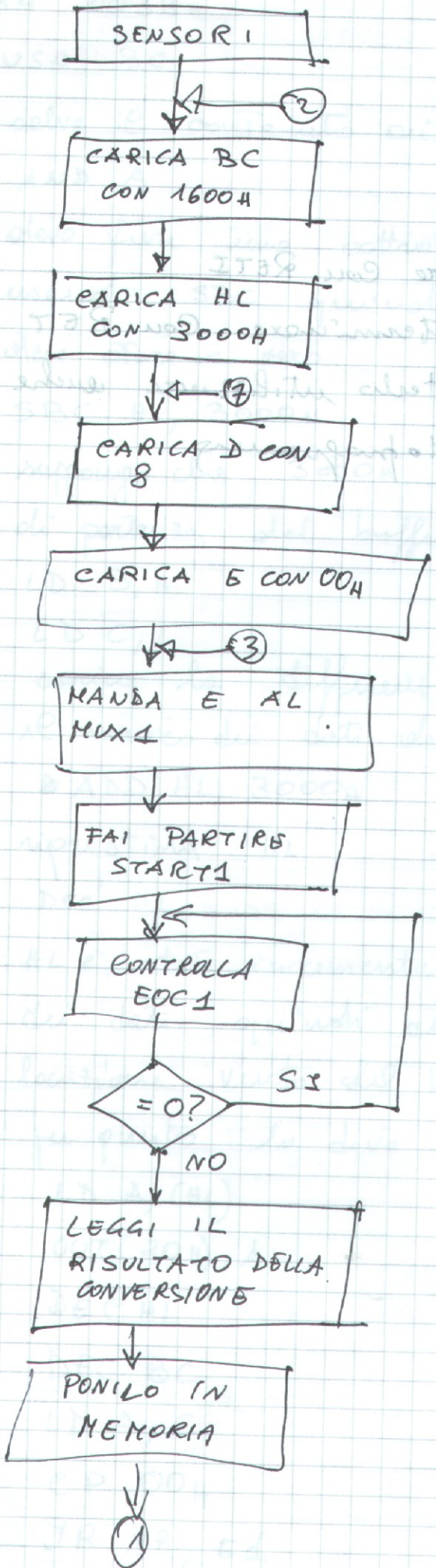

ORG 0038H
SALVA B
DEC HL
OTDR
INCREMENTA HL
RIPRISTINA B
ORG 0038H
SALVA BC
SOTTRAI AD HL L'INDIRIZZO INIZIALE DEL BUFFER DATI
PONI IL RISULTATO IN BC
DECREMENTA HL
RIPRISTINA HL
DECREMENTA HL
1

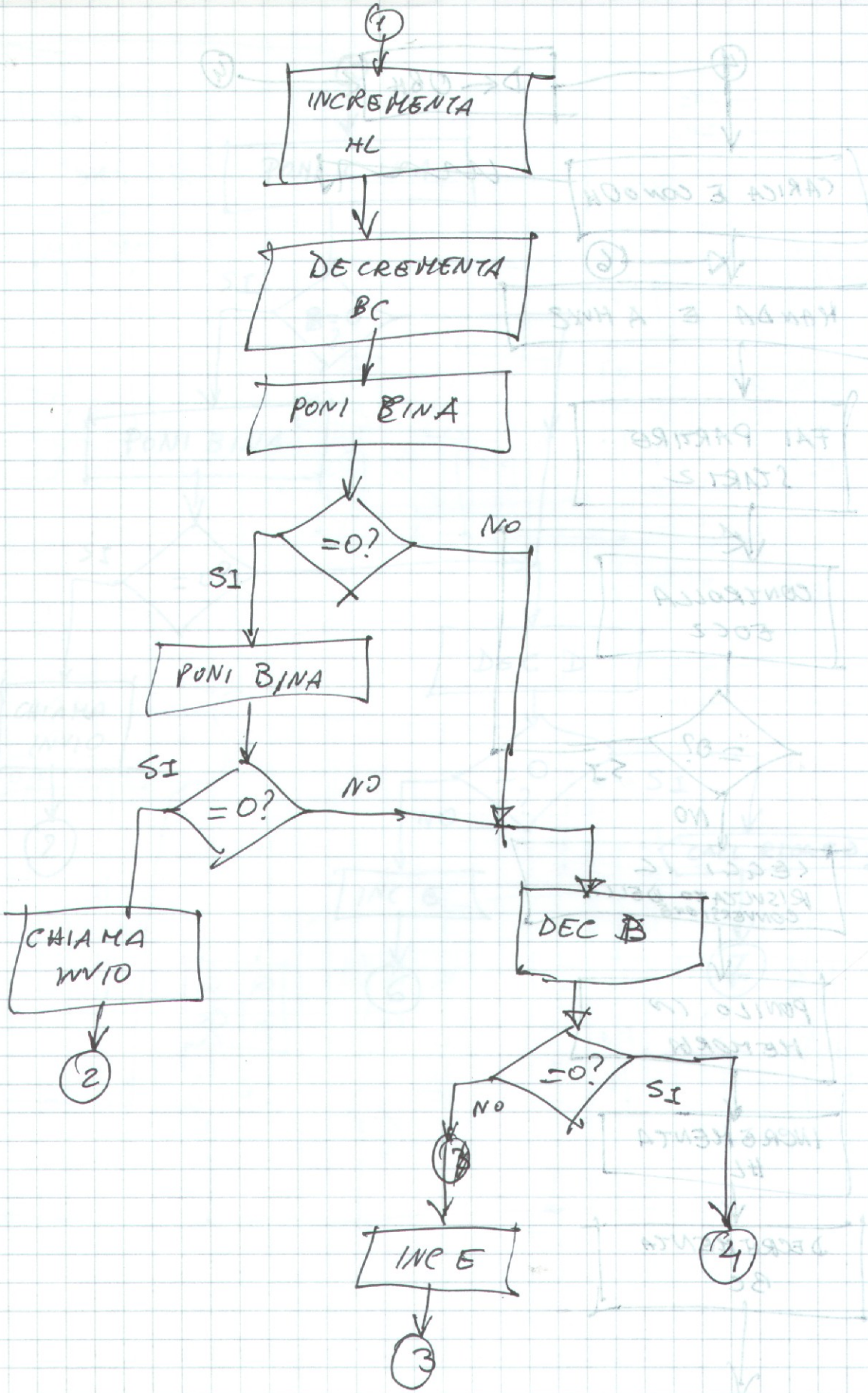

PONI IN HL L'INDIRIZZO DELLA 1ª LOCAZIONE DEL BUFFER DATI
PONI IN E 00H
INVIA E AL MUX 1
FAI
1
A
CARICA IN A IL CONTENUTO DI HL
INVIALO ALLA PORTA 20H
DEC HL
DEC BC
PONI IN A, C
SI
=0
NO
PONI IN A, B
1
=0
NO
SI
RITORNA
RIPRISTINA BC
RITORNA

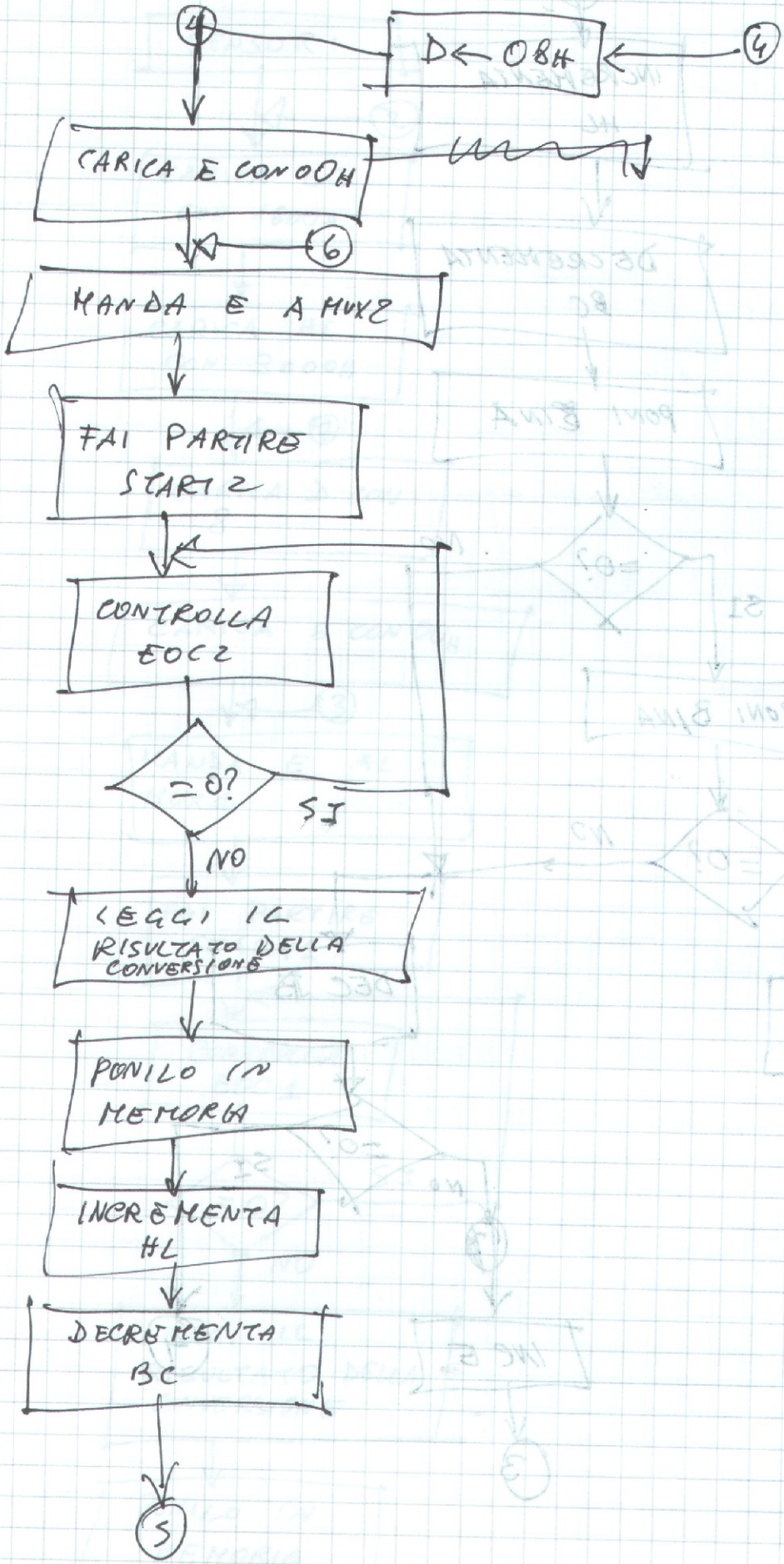
```
        ORG  0038H
INVIO : PUSH BC
        ; salva il contenuto originario nello stack
        ; AND A
        ; devo fare una sottrazione a 16 bit
        ; usando SBC quindi il flag di carry
        ; deve stare a zero
          SBC HL, 3000H
        ; suppongo che 3000H sia l'indirizzo
        ; di partenza del buffer dei dati
          LD B, H
          LD C, L
        ; carico la differenza che rappresenta
        ; il numero dei dati che trasferire in uscita
          ADD HL, 3000H
        ; ripristino HL
          DEC HL
        ; HL è stato incrementato dopo ogni i/o
        ; dei dati, quindi adesso punta alla pr
        ; locazione vuota del buffe
        ; per questo lo devo decrementare
E1 :    LD A, (HL)
        OUT (20H), A
        DEC HL
        DEC BC
        LD A, C
        CP 00H
        JR NZ, E1
```

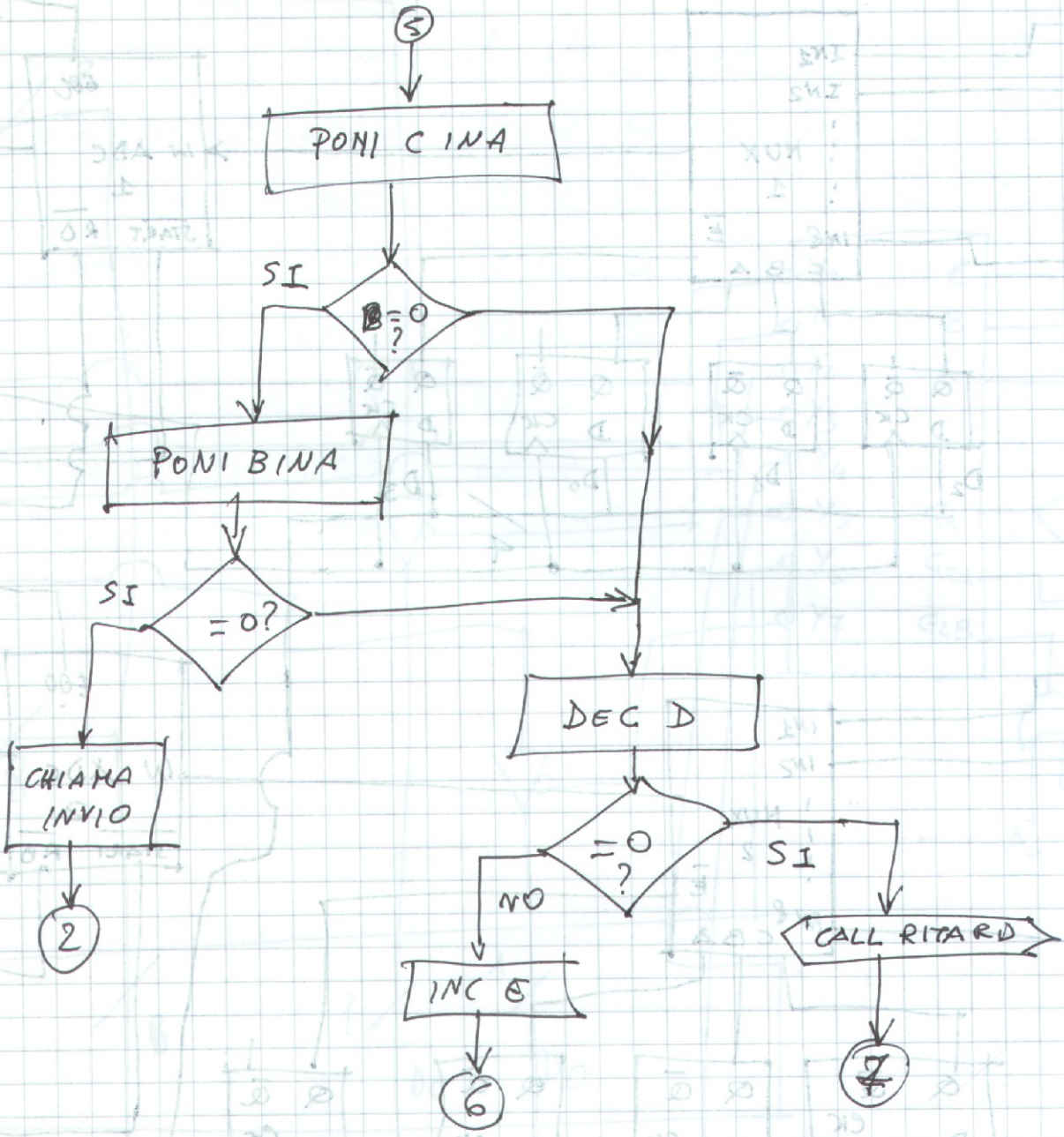
```
INVIO : PUSH BC
        ; salva il contenuto originario nello stack
        ; AND A
        ; devo fare una sottrazione a 16 bit
        ; usando SBC quindi il flag di carry
        ; deve stare a zero
          SBC HL, 3000H
        ; suppongo che 3000H sia l'indirizzo
        ; di partenza del buffer dei dati
          LD B, H
          LD C, L
        ; carica la differenza che rappresenta
        ; il numero dei dati che trasferire in uscita
          ADD HL, 3000H
        ; ripristina HL
          DEC HL
        ; HL è stato incrementato dopo ogni immissione
        ; dei dati, quindi adesso punta alla prima
        ; locazione vuota del buffer
        ; per questo lo devo decrementare
E1 :    LD A, (HL)
        OUT (20H), A
        DEC HL
        DEC BC
        LD A, C
        CP 00H
        JR NZ, E1
        LD A, B
```

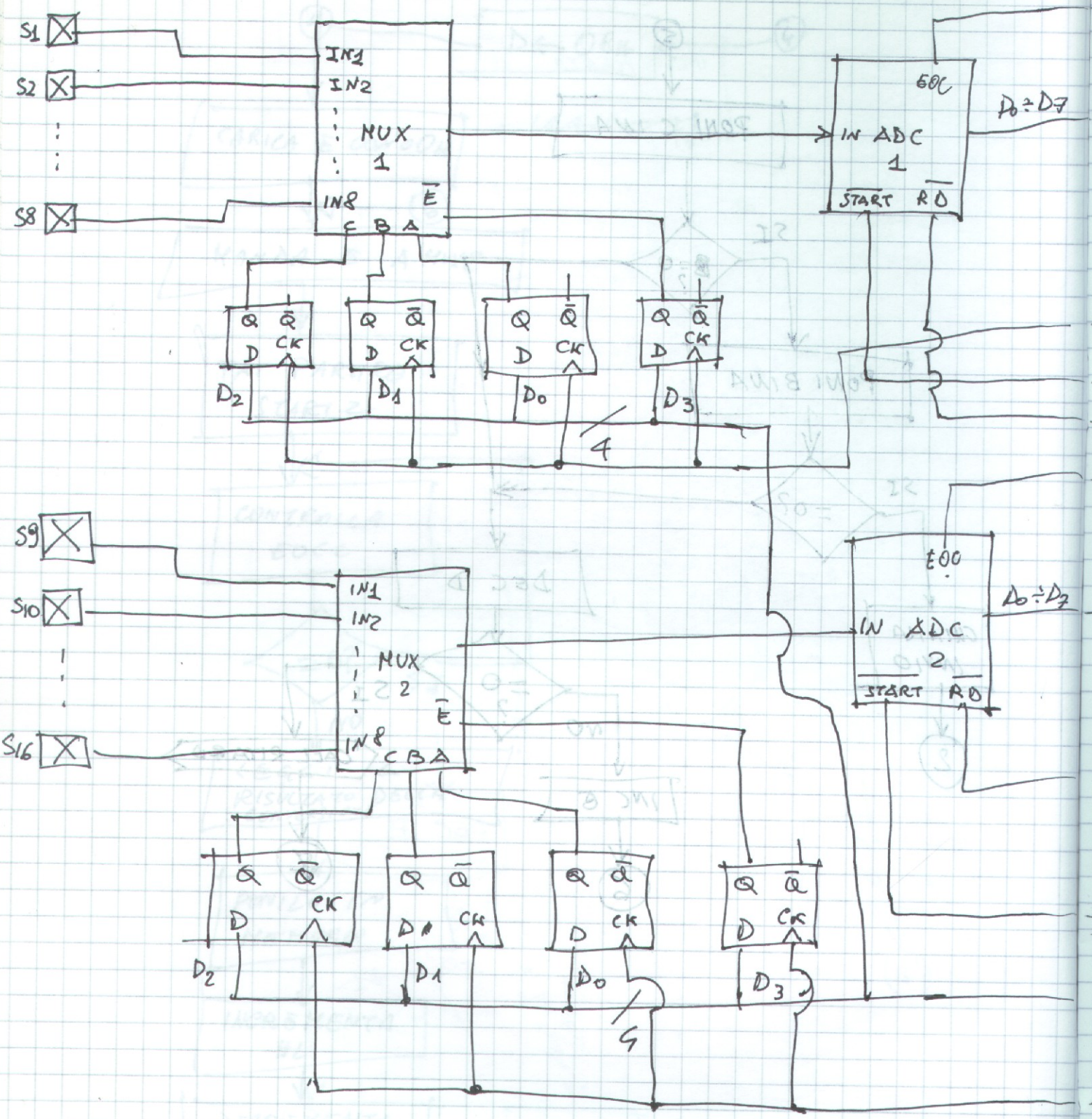
```
CP 00H
JR NZ, E1
~~RET~~ POP BC
; RET
; dovrebbe Terminare con RETI
; ma lo faccio Terminare con RET
; in modo da poterlo utilizzare anche
; come normale sottoprogramma
```

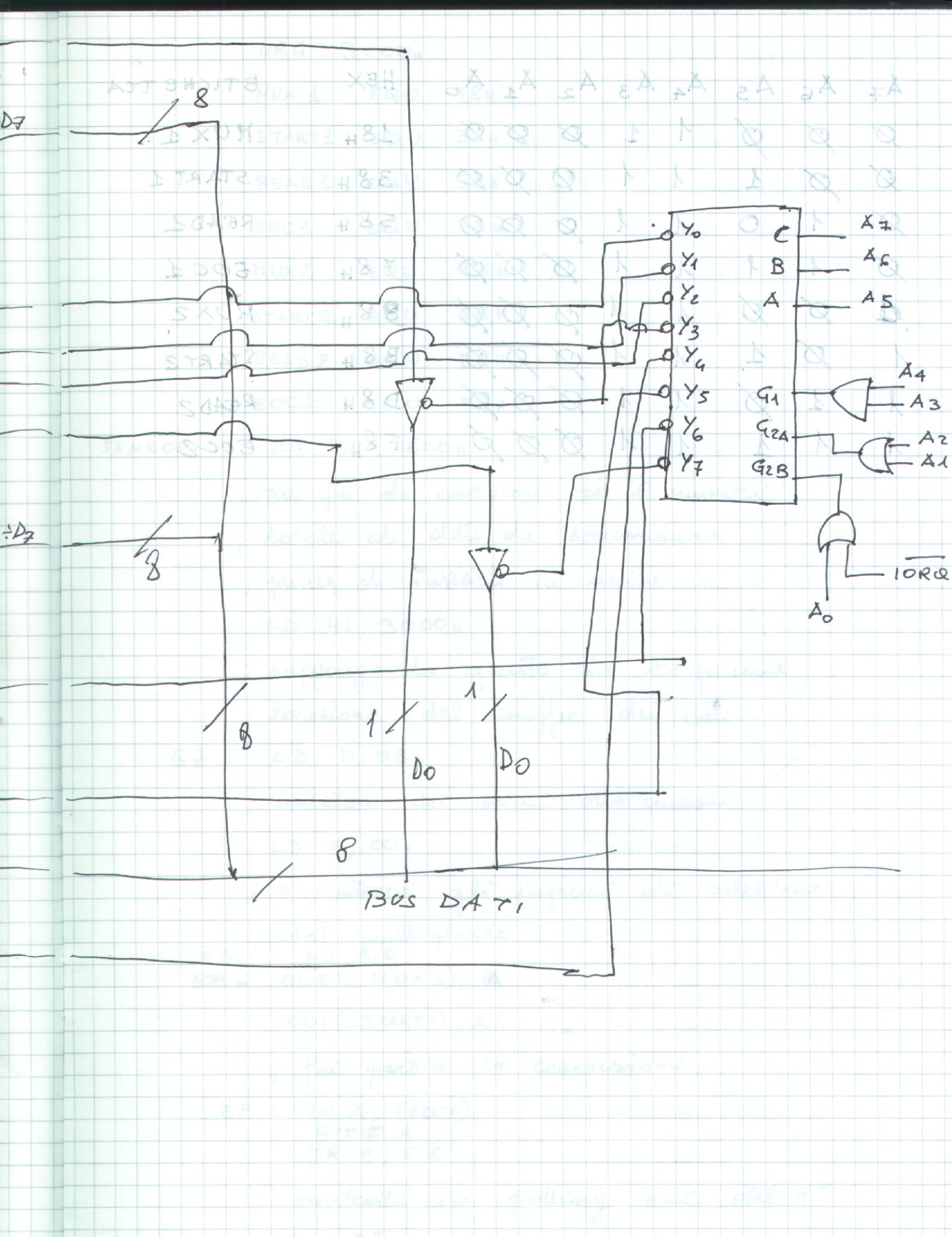

SENSORI
2
CARICA BC CON 1600H
CARICA HL CON 3000H
7
CARICA D CON 8
CARICA E CON 00H
3
MANDA E AL MUX1
FAI PARTIRE START1
CONTROLLA EOC1
= 0?
SI
NO
LEGGI IL RISULTATO DELLA CONVERSIONE
PONILO IN MEMORIA
1


1
INCREMENTA HL
DECREMENTA BC
PONI B/NA
=0?
NO
SI
PONI B/NA
SI
=0?
NO
CHIAMA INVIO
2
DEC B
=0?
SI
NO
INC E
3
4


4
D ← 08H
4
CARICA E CON 00H
6
MANDA E A MUX2
FAI PARTIRE START2
CONTROLLA EOC2
=0?
SI
NO
LEGGI IL RISULTATO DELLA CONVERSIONE
PONILO IN MEMORIA
INCREMENTA HL
DECREMENTA BC
5


5
PONI C INA
SI
B=0
?
PONI BINA
SI
=0?
CHIAMA
INVIO
2
DEC D
=0
?
SI
NO
CALL RITARD
INC E
6
7


S1
S2
S8
IN1
IN2
MUX 1
IN8
E
C B A
EOC
D0 ÷ D7
IN ADC 1
START RD
Q Q̄ D CK
D2
D1
D0
D3
4
S9
S10
S16
MUX 2
EOC
D0 ÷ D7
IN ADC 2
START RD
D2
D1
D0
D3
4


8
D7
÷D7
8
8
1
1
1
D0
D0
8
BUS DATI
Y0
Y1
Y2
Y3
Y4
Y5
Y6
Y7
C
B
A
G1
G2A
G2B
A7
A6
A5
A4
A3
A2
A1
IORQ
A0

| $A_7$ | $A_6$ | $A_5$ | $A_4$ | $A_3$ | $A_2$ | $A_1$ | $A_0$ | HEX | ETICHETTA |
|---|---|---|---|---|---|---|---|---|---|
| 0 | 0 | 0 | 1 | 1 | 0 | 0 | 0 | $18_H$ | MUX1 |
| 0 | 0 | 1 | 1 | 1 | 0 | 0 | 0 | $38_H$ | START1 |
| 0 | 1 | 0 | 1 | 1 | 0 | 0 | 0 | $58_H$ | READ1 |
| 0 | 1 | 1 | 1 | 1 | 0 | 0 | 0 | $78_H$ | EOC1 |
| 1 | 0 | 0 | 1 | 1 | 0 | 0 | 0 | $98_H$ | MUX2 |
| 1 | 0 | 1 | 1 | 1 | 0 | 0 | 0 | $B8_H$ | START2 |
| 1 | 1 | 0 | 1 | 1 | 0 | 0 | 0 | $D8_H$ | READ2 |
| 1 | 1 | 1 | 1 | 1 | 0 | 0 | 0 | $F8_H$ | EOC2 |

```
        ORG 3C00H
MUX1    EQU 18H
START1  EQU 38H
READ1   EQU 58H
EOC1    EQU 78H
MUX2    EQU 98H
START2  EQU B8H
READ2   EQU D8H
EOC2    EQU F8H
SENSORI : LD BC, 1600H
        ; BC fa da contatore per il numero
        ; totale di dati da accumulare
        ; prima di inviarli in uscita
          LD HL, 3000H
        ; suppongo che questo sia la prima
        ; locazione del buffer dei dati
E7 :    LD D, 08H
        ; contatore dei primi otto sensori
          LD E, 00H
        ; E conterrà gli ingressi di selezione
        ; del multiplexer
E3:       LD A,E
          OUT(MUX1), A
          OUT(START1), A
        ; fai partire la conversione
E8 :    IN A, (EOC1)
          BIT 0, A
          JR Z, E8
        ; controlla in polling EOC del 1°
```

```
MUX1    EQU  18H
START1  EQU  38H
READ1   EQU  58H
EOC1    EQU  78H
MUX2    EQU  98H
START2  EQU  B8H
READ2   EQU  D8H
EOC2    EQU  F8H
SENSORI : LD BC, 1600H
        ; BC fa da contatore per il numero
        ; totale di dati da accumulare
        ; prima di inviarli in uscita
          LD HL, 3000H
        ; suppongo che questo sia la prima
        ; locazione del buffer dei dati
E7 :    LD D, 08H
        ; contatore dei primi otto sensori
          LD E, 00H
        ; E conterrà gli ingressi di selezione
        ; del multiplexer
E3:       LD A, E
E8 :    OUT (MUX1), A
          OUT (START1), A
        ; fai partire la conversione
E8 :    IN A, (EOC1)
          BIT 0, A
          JR Z, E8
        ; controlla in polling EOC del 1°
        ; convertitore
```

```
        IN A, (READ1)
        ; quando EOC è tornato ad 1
        ; è terminata la conversione
        ; quindi puoi leggere il risultato
        ; della conversione
        LD (HL), A
        INC HL
        DEC BC
        LD A, C
        CP 00H
        ; controllo se BC è arrivato a zero
        JR NZ, AVANTI
        ; se C è nullo controllo B
        LD A, B
        CP 00H
        ; se anche B è nullo invio i dati
        CALL INVIO
        ; e torno all'inizio
        JP, SENSORI
AVANTI: DEC D
        JR Z, AVANTI2
        INC E
        JP, E3
        ; se non ho finito continuo con un
        ; altro sensore del primo gruppo
AVANTI2: LD D, 08H
        ; altrimenti passo ai sensori
```

```
; quando EOC è tornato ad 1
; è terminata la conversione
; quindi puoi leggere il risultato
; della conversione
    LD (HL), A
    INC HL
    DEC BC
    LD A, C
    CP 00H
; controllo se BC è arrivato a zero
    JR NZ, AVANTI
; se C è nullo controllo B
    LD A, B
    CP 00H
; se anche B è nullo invio i dati
    CALL INVIO
; e torno all'inizio
    JP, SENSORI
AVANTI: DEC D
    JR Z, AVANTI2
    INC E
    JP, E3
; se non ho finito continuo con un
; altro sensore del primo gruppo
AVANTI2: LD D, 08H
; altrimenti passo ai sensori
; del secondo gruppo
```

```
        LD E, 00H
E6 :    LD A, E
        OUT (MUX2), A
        OUT (START2), A
E9 :    IN A, (EOC2)
        JR Z BIT 0, A
        JR NZ, E9
        IN A, (READ2)
        LD (HL), A
        INC HL
        DEC BC
        LD A, C
        CP 00H
        JR NZ, AVANTI 3
        LD A, B
        CP 00H
        JR NZ, AVANTI 3
        CALL INVIO
        JP, SENSORI
AVANTI 3 : DEC D
        JR Z, AVANTI 4
        INC E
        JP, E6
AVANTI 4 : CALL RITARD
        ; sono finiti anche i sensori del 2° gruppo
        ; e aspetto un minuto
```

```
        OUT (MUX2), A
        OUT (START2), A
E9 :    IN A, (EOC2)
        JR Z BIT 0, A
        JR NZ, E9
        IN A, (READ2)
        LD (HL), A
        INC HL
        DEC BC
        LD A, C
        CP 00H
        JR NZ, AVANTI 3
        LD A, B
        CP 00H
        JR NZ, AVANTI 3
        CALL INVIO
        JP, SENSORI
AVANTI 3 : DEC D
        JR Z, AVANTI 4
        INC E
        JP, E6
AVANTI 4 : CALL RITARD
        ; sono finiti anche i sensori del 2° gruppo
        ; e aspetto un minuto
        JP, E7.
```