MICROCONTROLLORE AT89C2051

Nello Alessandrini

Un piccolo-grande processore con un economico sistema di sviluppo.

4ª parte

Premessa

In questo numero verranno presentate alcune istruzioni del microcontrollore 2051 e tre programmi di esempio per l'uso della linea seriale in collegamento verso un monitor o un P.C. in emulazione terminale. Se non si possiede un terminale video con linea RS-232 si può utilizzare come programma di emulazione terminale il CROSS-TALK o meglio ancora il GET-51 della GRIFO.

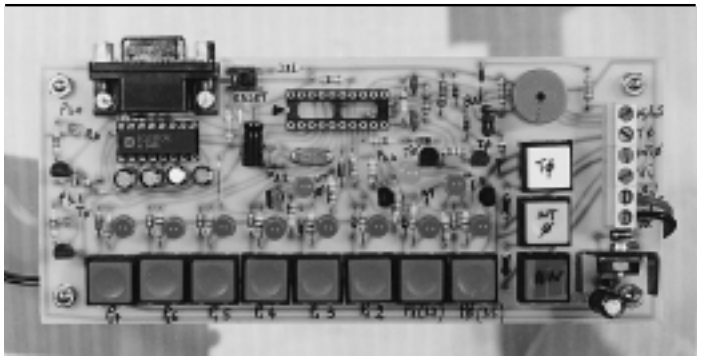
Emulazione con GET-51

Il software in questione è molto semplice da usare e può essere spiegato in poco tempo. In pratica è sufficiente lanciare GET51, premere invio due volte, alla comparsa del menù premere il tasto F10, portarsi con il tasto freccia di sinistra sulla voce Options e confermare con invio (vedi figura 1).

A questo punto possiamo scegliere l'opzione desiderata tramite i tasti freccia e confermarla con invio. L'opzione Serial Port ci consente di utilizzare la COM1 oppure la COM2, COM3 o COM4, ed anche la velocità di comunicazione (Baud Rate). L'utente potrà scegliere la COM che vuole, ma per quanto riguarda la velocità dovrà attenersi a 9.600.

L'opzione Video consente di togliere o mettere il colore, mentre l'opzione Terminal è quella che ci riguarda, ed è quella che una volta settata la COM e il Baud Rate, verrà direttamente richiamata.

Settata l'opzione Terminal e confermata con invio si è già in presenza di un terminale video completo. Per uscire da questo stato sarà sufficiente premere il tasto F10, portarsi su File, confermare con invio, selezionare con il tasto freccia l'opzione Exit e ripremere invio (figura 2).



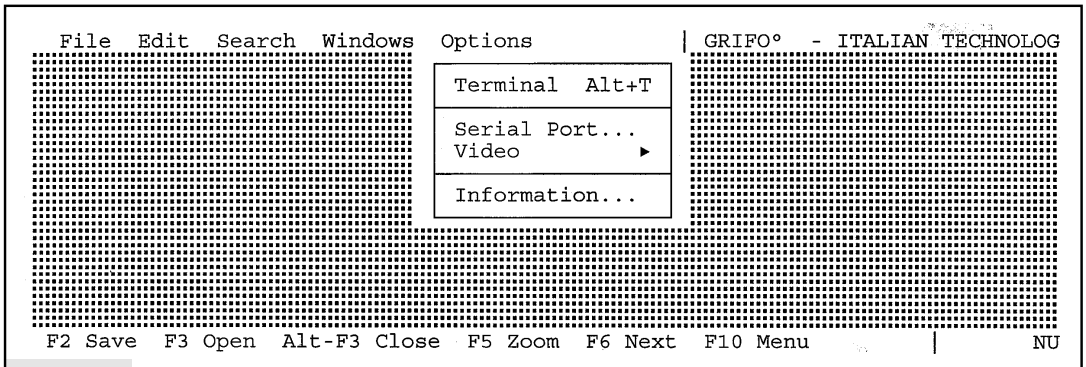


figura 1

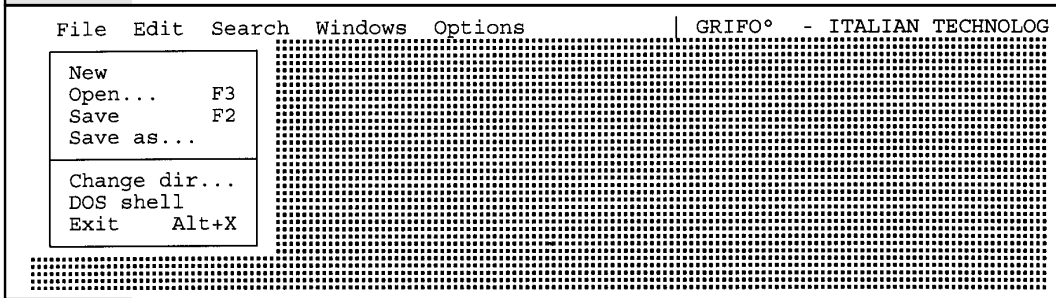


figura 2

Microcontroller Instruction Set

For interrupt response time information, refer to the hardware description chapter.

Instructions that Affect Flag Settings(1)

Instruction	Flag			Instruction	Flag		
	C	OV	AC		C	OV	AC
ADD	X	X	X	CLR C	O		
ADDC	X	X	X	CPL C	X		
SUBB	X	X	X	ANL C,bit	X		
MUL	O	X		ANL C,/bit	X		
DIV	O	X		ORL C,bit	X		
DA	X			ORL C, bit	X		
RRC	X			MOV C,bit	X		
RLC	X			CJNE	X		
SETB C	1						

Note 1. Operations on SFR byte address 208 or bit addresses 209-215 (that is, the PSW or bits in the PSW) also affect flag settings.

The Instruction Set and Addressing Modes

Rn	Register R7-R0 of the currently selected Register Bank.
direct	8-bit internal data location's address. This could be an Internal Data RAM location (0-127) or a SFR [i.e., I/O port, control register, status register, etc. (128-255)].
@Ri	8-bit internal data RAM location (0-255) addressed indirectly through register R1or R0.
#data	8-bit constant included in instruction.
#data 16	16-bit constant included in instruction.
addr 16	16-bit destination address. Used by LCALL and LJMP. A branch can be anywhere within the 64 Kbyte Program Memory address space.
addr 11	11-bit destination address. Used by ACALL and AJMP. The branch will be within the same 2 Kbyte page of program memory as the first byte of the following instruction.
rel	Signed (two's complement) 8-bit offset byte. Used by SJMP and all conditional jumps. Range is -128 to +127 bytes relative to first byte of the following instruction.
bit	Direct Addressed bit in Internal Data RAM or Special Function Register.

figura 3

Set Istruzioni

Anche se potrà sembrare sprecato ritengo opportuno inserire in questo numero e nei prossimi, tutto il set di istruzioni (anche se a livello sommario) della famiglia 51. In questo modo sarà possibile, anche al lettore che non potrà avere il manuale completo, conoscere le potenzialità del nostro micro. Nelle figure 3, 4 e 5 sono visibili le tabelle.

Programma n.1

Il programma seguente consente al nostro circuito TEST presentato in precedenza, di trasferire sul video i caratteri premuti dalla tastiera. Consiglio al lettore - programmatore di studiare bene la parte relativa alla gestione della seriale, ponendo particolare attenzione al settaggio dei registri specifici e ai relativi commenti. Questo programma come pure gli altri, ha come origine 1000h perché , come già detto, ciò è necessario per lavorare in emulazione.

Instruction Set Summary

	0	1	2	3	4	5	6	7
0	NOP	JBC bit,rel [3B, 2C]	JB bit, rel [3B, 2C]	JNB bit, rel [3B, 2C]	JC rel [2B, 2C]	JNC rel [2B, 2C]	JZ rel [2B, 2C]	JNZ rel [2B, 2C]
1	AJMP (P0) [2B, 2C]	ACALL (P0) [2B, 2C]	AJMP (P1) [2B, 2C]	ACALL (P1) [2B, 2C]	AJMP (P2) [2B, 2C]	ACALL (P2) [2B, 2C]	AJMP (P3) [2B, 2C]	ACALL (P3) [2B, 2C]
2	LJMP addr16 [3B, 2C]	LCALL addr16 [3B, 2C]	RET [2C]	RETI [2C]	ORL dir, A [2B]	ANL dir, A [2B]	XRL dir, a [2B]	ORL C, bit [2B, 2C]
3	RR A	RRC A	RL A	RLC A	ORL dir, #data [3B, 2C]	ANL dir, #data [3B, 2C]	XRL dir, #data [3B, 2C]	JMP @A + DPTR [2C]
4	INC A	DEC A	ADD A, #data [2B]	ADDC A, #data [2B]	ORL A, #data [2B]	ANL A, #data [2B]	XRL A, #data [2B]	MOV A, #data [2B]
5	INC dir [2B]	DEC dir [2B]	ADD A, dir [2B]	ADDC A, dir [2B]	ORL A, dir [2B]	ANL A, dir [2B]	XRL A, dir [2B]	MOV dir, #data [3B, 2C]
6	INC @R0	DEC @R0	ADD A, @R0	ADDC A, @R0	ORL A, @R0	ANL A, @R0	XRL A, @R0	MOV @R0, @data [2B]
7	INC @R1	DEC @R1	ADD A, @R1	ADDC A, @R1	ORL A, @R1	ANL A, @R1	XRL A, @R1	MOV @R1, #data [2B]
8	INC R0	DEC R0	ADD A, R0	ADDC A, R0	ORL A, R0	ANL A, R0	XRL A, R0	MOV R0, #data [2B]
9	INC R1	DEC R1	ADD A, R1	ADDC A, R1	ORL A, R1	ANL A, R1	XRL A, R1	MOV R1, #data [2B]
A	INC R2	DEC R2	ADD A, R2	ADDC A, R2	ORL A, R2	ANL A, R2	XRL A, R2	MOV R2, #data [2B]
B	INC R3	DEC R3	ADD A, R3	ADDC A, R3	ORL A, R3	ANL A, R3	XRL A, R3	MOV R3, #data [2B]
C	INC R4	DEC R4	ADD A, R4	ADDC A, R4	ORL A, R4	ANL A, R4	XRL A, R4	MOV R4, #data [2B]
D	INC R5	DEC R5	ADD A, R5	ADDC A, R5	ORL A, R5	ANL A, R5	XRL A, R5	MOV R5, #data [2B]
E	INC R6	DEC R6	ADD A, R6	ADDC A, R6	ORL A, R6	ANL A, R6	XRL A, R6	MOV R6, #data [2B]
F	INC R7	DEC R7	ADD A, R7	ADDC A, R7	ORL A, R7	ANL A, R7	XRL A, R7	MOV R7, #data [2B]

figura 4

```
;Programma POLLING.SRC                  echo di ogni carattere
;                                       ricevuto dalla seriale.
;
;
ADDR        EQU     1000h               ;Inizio codice in EPROM.
;
;
            org     ADDR+0000h          ;Vettore di reset.
            ljmp    START               ;Salto all'inizio del codice.
;
            org     ADDR+0035h          ;Inizio del codice.
;
;
START:      clr     TR1                 ;Timer/counter T1 in off.
            clr     TR0                 ;Timer/counter T0 in off.
            mov     SCON,#  01010010b   ;Settaggio registro SCON.
;                           ||||||||___ RI deve essere inizializzato a 0.
;                           |||||||____ TI deve inizializzare a 0.
;                           ||||||_____ Non rilevante (nel nostro caso).
;                           |||||______ Non rilevante (nel nostro caso).
;                           ||||_______ Abilitazione flag di ricezione.
;                           |||________ Disabilitazione multipr.commun.
;                           ||_________
;                           |__________>8 bits UART
            mov     TMOD,#  00100001b   ;Timers/counters mode register
;                           ||||||||___
```

```
;                          |||||||__>  T0 16 bits timer/count.register
;                          ||||||____  T0 utilizzato come timer
;                          |||||_____  T0 in off
;                          ||||______
;                          |||_______>  T1 8 bits timer/counter
;                                       con autoreload
;                          ||_________  T1 utilizzato come timer
;                          |__________  T1 in off
            mov     PCON,#  00000000b  ;Il MSB in PCON deve essere a 0
;                           |_________  se 1, raddopppia il baud rate
            mov     TL1,#0FDh          ;Si caricano i registri di T1 per
            mov     TH1,#0FDh          ;un baud rate di 9600 Baud
            mov     TL0,#00h           ;Si resettano i registri di T0
            mov     TH0,#00h           ;
            setb    TR1                ;T1 in on, abil. alla comunicaz.
;
            mov     DPTR,#MES1
            acall   VISMES             ;Visualizzazione del messaggio.
MAIN1:      acall   RXBYTE             ;Si attende un carattere.
            acall   TXBYTE             ;Trasmissione su seriale (echo).
            sjmp    MAIN1
;
;
;
;************************************************************
;** TXBYTE: Routine di trasmissione di un byte sulla seriale.
;** IN: ACC contenente il byte da trasmettere.
;** OUT: Nessuno.
;************************************************************
TXBYTE:     jnb     TI,TXBYTE          ;Attesa a trasmettere.
            clr     TI                 ;Reset del flag TI.
            mov     SBUF,A             ;Byte nel buffer di trasmissione.
            ret
;
;
;******************************************************
;** RXBYTE: Routine di ricezione di un byte dalla seriale.
;** IN: Nessuno.
;** OUT: ACC contenente il byte ricevuto.
;******************************************************
RXBYTE:     jnb     RI,RXBYTE          ;Si attende di ricevere il byte.
            mov     A,SBUF             ;Si legge il byte dal buffer
;                                      della seriale.
            clr     RI                 ;Reset del flag RI.
            ret
;
;
;************************************************
;** VISMES: Routine che visualizza un messaggio.
;** IN: DPTR contenente l'indirizzo del messaggio.
;** OUT: Nessuno.
;************************************************
VISMES:     push    ACC                ;Visualizzazione dei caratteri
VISMES1:    mov     A,#0               ;di un messaggio puntato da DPTR
            movc    A,@A+DPTR          ;fino al carattere "$".
            cjne    A,#36,VISMES2
            sjmp    VISMESF
VISMES2:    lcall   TXBYTE
            inc     DPTR
            sjmp    VISMES1
VISMESF:    pop     ACC
            ret
```

```
;
;
;
MES1:       DB      12
            DB      '** GESTIONE DELLA SERIALE IN POLLING **'
            DB      13,10,10
            DB      'Trasmissione sulla seriale dei caratteri '
            DB      'che riceve dalla medesima(ECHO).'
            DB      13,10,10,'$'
;
;
            end
```

Instruction Set Summary (Continued)

	8	9	A	B	C	D	E	F
0	SJMP REL [2B, 2C]	MOV DPTR,# data 16 [3B, 2C]	ORL C, /bit [2B, 2C]	ANL C, /bit [2B, 2C]	PUSH dir [2B, 2C]	POP dir [2B, 2C]	MOVX A, @DPTR [2C]	MOVX @DPTR, A [2C]
1	AJMP (P4) [2B, 2C]	ACALL (P4) [2B, 2C]	AJMP (P5) [2B, 2C]	ACALL (P5) [2B, 2C]	AJMP (P6) [2B, 2C]	ACALL (P6) [2B, 2C]	AJMP (P7) [2B, 2C]	ACALL (P7) [2B, 2C]
2	ANL C, bit [2B, 2C]	MOV bit, C [2B, 2C]	MOV C, bit [2B]	CPL bit [2B]	CLR bit [2B]	SETB bit [2B]	MOVX A, @R0 [2C]	MOVX wR0, A [2C]
3	MOVC A, @A + PC [2C]	MOVC A, @A + DPTR [2C]	INC DPTR [2C]	CPL C	CLR C	SETB C	MOVX A, @RI [2C]	MOVX @RI, A [2C]
4	DIV AB [2B, 4C]	SUBB A, #data [2B]	MUL AB [4C]	CJNE A, #data, rel [3B, 2C]	SWAP A	DA A	CLR A	CPL A
5	MOV dir, dir [3B, 2C]	SUBB A, dir [2B]		CJNE A, dir, rel [3B, 2C]	XCH A, dir [2B]	DJNZ dir, rel [3B, 2C]	MOV A, dir [2B]	MOV dir, A [2B]
6	MOV dir, @R0 [2B, 2C]	SUBB A, @R0	MOV @R0, dir [2B, 2C]	CJNE @R0, #data, rel [3B, 2C]	XCH A, @R0	XCHD A, @R0	MOV A, @R0	MOV @R0, A
7	MOV dir, @R1 [2B, 2C]	SUBB A, @R1	MOV @R1, dir [2B, 2C]	CJNE @R1, #data, rel [3B, 2C]	XCH A, @R1	XCHD A, @R1	MOV A, @R1	MOV @R1, A
8	MOV dir, R0 [2B, 2C]	SUBB A, R0	MOV R0, dir [2B, 2C]	CJNE R0, #data, rel [3B, 2C]	XCH A, R0	DJNZ R0, rel [2B, 2C]	MOV A, R0	MOV R0, A
9	MOV dir, R1 [2B, 2C]	SUBB A, R1	MOV R1, dir [2B, 2C]	CJNE R1, #data, rel [3B, 2C]	XCH A, R1	DJNZ R1, rel [2B, 2C]	MOV A, R1	MOV R1, A
A	MOV dir, R2 [2B, 2C]	SUBB A, R2	MOV R2, dir [2B, 2C]	CJNE R2, #data, rel [3B, 2C]	XCH A, R2	DJNZ R2, rel [2B, 2C]	MOV A, R2	MOV R2, A
B	MOV dir, R3 [2B, 2C]	SUBB A, R3	MOV R3, dir [2B, 2C]	CJNE R3, #data, rel [3B, 2C]	XCH A, R3	DJNZ R3, rel [2B, 2C]	MOV A, R3	MOV R3, A
C	MOV dir, R4 [2B, 2C]	SUBB A, R4	MOV R4, dir [2B, 2C]	CJNE R4, #data, rel [3B, 2C]	XCH A, R4	DJNZ R4, rel [2B, 2C]	MOV A, R4	MOV R4, A
D	MOV dir, R5 [2B, 2C]	SUBB A, R5	MOV R5, dir [2B, 2C]	CJNE R5, #data, rel [3B, 2C]	XCH A, R5	DJNZ R5, rel [2B, 2C]	MOV A, R5	MOV R5, A
E	MOV dir, R6 [2B, 2C]	SUBB A, R6	MOV R6, dir [2B, 2C]	CJNE R6, #data, rel [3B, 2C]	XCH A, R6	DJNZ R6, rel [2B, 2C]	MOV A, R6	MOV R6. A
F	MOV dir, R7 [2B, 2C]	SUBB A, R7	MOV R7, dir [2B, 2C]	CJNE R7, #data, rel [3B, 2C]	XCH A, R7	DJNZ R7, rel [2B, 2C]	MOV A, R7	MOV R7, A

Key:
[2B] = 2 Byte, [3B] = 3 Byte, [2C] = 2 Cycle, [4C] = 4 Cycle, Blank = 1 byte/1 cycle

figura 5

Programma n.2

Questo programma consente di abilitare o disabilitare le uscite dei PORT P1 e P3 collegate con i LED montati sopra i pulsantini del circuito TEST. Premendo una prima volta il tasto specifico (ad esempio il tasto 1) della tastiera si illuminerà il LED (in questo caso quello connesso a P3.7); ripremendo il tasto 1 lo stesso LED si spegnerà. In questa applicazione la visualizzazione (il messaggio è in fondo al programma) non è modificabile dalla premuta dei tasti, ma è comunque utile per confermarci l'esattezza del collegamento del cavo seriale e del programma.

```
;Programma SR-SER.SRC   Set/Reset di linee di I/O.

;                  "0" –> P3.5
;                  "1" –> P3.7
;                  "2" –> P1.2
;                  "3" –> P1.3
;                  "4" –> P1.4
;                  "5" –> P1.5
;                  "6" –> P1.6
;                  "7" –> P1.7
;
ADDR       EQU     1000h               ;Inizio codice per Emulaz.
;
;
           org     ADDR+0000h          ;Vettore di reset.
           ljmp    START               ;Salto all'inizio del codice.
;
           org     ADDR+0035h          ;Inizio del codice.
;
;
START:     clr     TR1                 ;Timer/counter T1 in off.
           clr     TR0                 ;Timer/counter T0 in off.
           mov     SCON,#  01010010b   ;Settaggio registro SCON.

           mov     TMOD,#  00100001b   ;Settaggio registro TMOD.

           mov     PCON,#  00000000b   ;
;                          |_________   SMOD, se a 1, raddopppia
;                                       il baud rate
           mov     TL1,#0FDh           ;Si caricano i registri di T1
           mov     TH1,#0FDh           ;per un baud rate di 9600.
           mov     TL0,#00h            ;Si resettano i registri di T0
           mov     TH0,#00h            ;
           setb    TR1                 ;Abilitazione alla comunicazione
;
           mov     DPTR,#MES1
           acall   VISMES              ;Visualizzazione del messaggio.
MAIN1:     acall   RXBYTE              ;Si attende un carattere.
           cjne    A,#'0',MAIN2
           cpl     P3.5
           sjmp    MAIN1
MAIN2:     cjne    A,#'1',MAIN3
           cpl     P3.7
           sjmp    MAIN1
MAIN3:     cjne    A,#'2',MAIN4
           cpl     P1.2
           sjmp    MAIN1
MAIN4:     cjne    A,#'3',MAIN5
           cpl     P1.3
           sjmp    MAIN1
MAIN5:     cjne    A,#'4',MAIN6
           cpl     P1.4
           sjmp    MAIN1
MAIN6:     cjne    A,#'5',MAIN7
           cpl     P1.5
           sjmp    MAIN1
MAIN7:     cjne    A,#'6',MAIN8
           cpl     P1.6
           sjmp    MAIN1
MAIN8:     cjne    A,#'7',MAIN9
           cpl     P1.7
MAIN9:     sjmp    MAIN1
```

```
;
;************* PROCEDURA **********************
;** TXBYTE: Routine di trasmissione di un byte.
;** IN: ACC contenente il byte da trasmettere.
;** OUT: Nessuno.
;*********************************************
TXBYTE:     jnb     TI,TXBYTE           ;Si attendel'abilitazione.
            clr     TI                  ;Reset del flag TI.
            mov     SBUF,A              ;Si mette il byte nel buffer
;                                       di trasmissione.
            ret
;
;*********************************************
;** RXBYTE: Routine di ricezione di un byte.
;** IN: Nessuno
;** OUT: ACC Contenente il byte ricevuto
;*********************************************
RXBYTE:     jnb     RI,RXBYTE           ;Attesa del byte.
            mov     A,SBUF              ;Si legge il byte dal buffer
;                                       della seriale.
            clr     RI                  ;Reset del flag RI.
            ret
;
;
;************************************************
;** VISMES: Routine che visualizza un messaggio.
;** IN: DPTR contenente l'indirizzo del messaggio.
;** OUT: Nessuno.
;************************************************
VISMES:     push    ACC                 ;Si visualizzano i caratteri
VISMES1:    mov     A,#0                ;di un messaggio puntato da
;                                       DPTR fino a carattere "$".
            movc    A,@A+DPTR
            cjne    A,#36,VISMES2
            sjmp    VISMESF
VISMES2:    lcall   TXBYTE
            inc     DPTR
            sjmp    VISMES1
VISMESF:    pop     ACC
            ret
;
;
;******** Messaggi **************
;
;
MES1:       DB      12
            DB      '* SET/RESET DI LINEE DI I/O *'
            DB      13,10,10
            DB      'Attesa di un carattere dalla '
            DB      'seriale ("0"..."7") e settaggiò
            DB      ' della I/O associata:'
            DB      10,13,'"0" –> P3.5'
            DB      10,13,'"1" –> P3.7'
            DB      10,13,'"2" –> P1.2'
            DB      10,13,'"3" –> P1.3'
            DB      10,13,'"4" –> P1.4'
            DB      10,13,'"5" –> P1.5'
            DB      10,13,'"6" –> P1.6'
            DB      10,13,'"7" –> P1.7'
            DB      13,10,'$'
;
            end
```

Programma n.3

Questo programma visualizza sul video lo stato logico dei PORT P1 e P3. Premendo i pulsanti relativi ai vari bit (i pulsantini sono quelli montati sul circuito TEST) si vedranno sul video le variazioni degli stati logici.

```
;Programma P1P3-SER.SRC          Si visualizza su monitor lo
;                                stato degli ingressi di P1 e P3.
;
ADDR       EQU    1000h               ;Inizio codice Emulaz.
;
;
           org    ADDR+0000h          ;Vettore di reset.
           ljmp   START               ;Salto all'inizio del codice.
;
           org    ADDR+0035h          ;Inizio del codice.
;
;
START:     clr    TR1                 ;Timer/counter T1 in off.
           clr    TR0                 ;Timer/counter T0 in off.
           mov    SCON,#  01010010b   ;Settaggio registro SCON
;                         |________>   8 bits UART
           mov    TMOD,#  00100001b   ;Timers/counters mode register
;                         |_________   T1 in off
           mov    PCON,#  00000000b   ;Il MSB in PCON deve essere a 0
           mov    TL1,#0FDh           ;Si caricano i registri di T1 per
           mov    TH1,#0FDh           ;un baud rate di 9600 Baud
           mov    TL0,#00h            ;Si resettano i registri di T0
           mov    TH0,#00h            ;
           setb   TR1                 ;abilitazione alla comunicazione
;
           mov    DPTR,#MES1
           acall  VISMES              ;Visualizzazione del messaggio.
MAIN1:     mov    c,P1.7              ;Visualizzazione stato di P1.
           acall  VISSTAT
           mov    c,P1.6
           acall  VISSTAT
           mov    c,P1.5
           acall  VISSTAT
           mov    c,P1.4
           acall  VISSTAT
           mov    c,P1.3
           acall  VISSTAT
           mov    c,P1.2
           acall  VISSTAT
           mov    c,P1.1
           acall  VISSTAT
           mov    c,P1.0
           acall  VISSTAT
;
           mov    A,#' '              ;Si trasmette uno SPACE.
           acall  TXBYTE
           mov    A,#' '              ;Si trasmette uno SPACE.
           acall  TXBYTE
           mov    A,#' '              ;Si trasmette uno SPACE.
           acall  TXBYTE
           mov   A,#' '               ;Si trasmette uno SPACE.
           acall  TXBYTE
           mov    A,#' '              ;Si trasmette uno SPACE.
           acall  TXBYTE
;
           mov    c,P3.7              ;Visualizzazione stato di P3.
           acall  VISSTAT
```

```
            mov     c,P3.6
            acall   VISSTAT
            mov     c,P3.5
            acall   VISSTAT
            mov     c,P3.4
            acall   VISSTAT
            mov     c,P3.3
            acall   VISSTAT
            mov     c,P3.2
            acall   VISSTAT
            mov     c,P3.1
            acall   VISSTAT
            mov     c,P3.0
            acall   VISSTAT
            mov     A,#13               ;Si trasmette un CR.
            acall   TXBYTE
;
            sjmp    MAIN1
;
;
TXBYTE:     jnb     TI,TXBYTE           ;Abilitazione a trasmettere.
            clr     TI                  ;Reset del flag TI.
            mov     SBUF,A              ;Byte nel buffer di trasmissione.
            ret
;
;
RXBYTE:     jnb     RI,RXBYTE           ;Attesa di un byte.
            mov     A,SBUF              ;Si legge il byte dal buffer
;                                       della seriale
            clr     RI                  ;Reset del flag RI.
            ret
;
;
VISMES:     push    ACC                 ;Visualizzazione caratteri di
VISMES1:    mov     A,#0                ; un messaggio puntato da DPTR
            movc    A,@A+DPTR           ;fino alla premuta di "$".
            cjne    A,#36,VISMES2
            sjmp    VISMESF
VISMES2:    lcall   TXBYTE
            inc     DPTR
            sjmp    VISMES1
VISMESF:    pop     ACC
            ret
;
;
;*********************************************************
;** VISSTAT: Routine che visualizza lo stato della linea.
;** IN: Carry contenente lo stato della linea.
;** OUT: Nessuno.
;*********************************************************
VISSTAT:    jnc     VISS1
            mov     A,#'0'
            acall   TXBYTE
            ret
VISS1:      mov     A,#'1'
            acall   TXBYTE
            ret
;
;
;***** Messaggi **************
;
;
```

```
MES1:       DB      12
            DB      '* STATO DELLE LINEE DI P1 E P3 *'
            DB      13,10,10
            DB      'Questo DEMO, visualizza lo stato '
            DB      'dei ports P1 e P3 della CPU.'
            DB      13,10,10
            DB      '  P1      P3  ',10,13,'$'
;
;
            end
```

Reperibilità e costi

KIT completo di microcontrollore ..... £ 75.000
Programmatore-Emulatore SIM2051 ..... £ 400.000
Software ASM-51 ..... £ 240.000
Chip 89C2051 ..... £ 15.000
CD ROM manuale del 2051 ..... £ 145.000

Ai prezzi sopra riportati occorre aggiungere le spese di spedizione.

Indirizzare richieste e informazioni a:

Nello Alessandrini - via Timavo, 10
40131 Bologna - tel. e fax 051/649.10.80

Nelle richieste sia telefoniche che fax ricordarsi di lasciare anche un recapito telefonico.