MICROCONTROLLORE AT89C2051

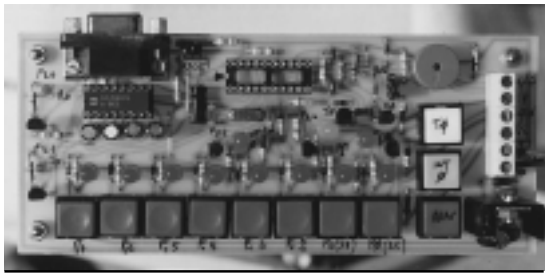
Nello Alessandrini

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

6ª ed ultima parte

Premessa

In questo numero completeremo le istruzioni relative al microcontrollore 2051 ed esamineremo un programma di esempio seguendo tutti i passaggi; dalla stesura al debugger, alla memorizzazione su circuito dedicato alle istruzioni del set 51, questa volta sono separate secondo i codici in esadecimale (vedi tabelle di figura 1,2,3,4).



Programma SHIFT

Ci proponiamo ora di realizzare un programma che faccia scorrere un LED sui port P1.7, P1.6, P1.5, P1.4, P1.3, P1.2, P3.7, P3.5 ottenendo uno shift da sinistra verso destra sulla scheda di test in corrispondenza dei LED presenti sopra gli 8 pulsantini affiancati (vedi fotografia).

Una volta terminate le prove, se vorremo, si potrà realizzare un circuito specifico senza tutta quella componentistica che per questa applicazione non è utilizzata. Il programma proposto è stato realizzato in questo modo per facilitare i programmatori alle prime armi. Inoltre, per fornire più dati, è stata scelta la stampa del file SHIFT.LST.

Come è noto, dalla procedura di assembler si ricava oltre al file con estensione .OBJ anche il file di stampa .LST, nel quale sono presenti oltre alle locazioni di memoria in esadecimale, anche i messaggi di errore e le loro individuazione.

```
SHIFT             Thu May 22 09:27:09 1997          PAGE 1

OBJECT MODULE PLACED IN SHIFT.OBJ
ASSEMBLER INVOKED BY: MA51 SHIFT
LOC  OBJ        LINE                          SOURCE

                1                             ; nome file shift.src
1000            2                             ADDR    EQU   1000H
```

```
1100             3                      ORG   ADDR+0100H
1100 7590FF      4            START:    MOV   P1,#0FFH
1103 75B0E0      5                      MOV   P3,#0E0H
1106 121110      6                      LCALL SHIFT
                 7            ;**** BREAK POINT N°2 *****
1109 C2B3        8                      CLR   P3.3
110B D2AA        9                      SETB  EX1
110D 00          10                     NOP
                 11           ;*************************
                 12
                 13
```

Hex Code	Number of Bytes	Mnemonic	Operands
00	1	NOP	
01	2	AJMP	code addr
02	3	LJMP	code addr
03	1	RR	A
04	1	INC	A
05	2	INC	data addr
06	1	INC	@R0
07	1	INC	@R1
08	1	INC	R0
09	1	INC	R1
0A	1	INC	R2
0B	1	INC	R3
0C	1	INC	R4
0D	1	INC	R5
0E	1	INC	R6
0F	1	INC	R7
10	3	JBC	bit addr, code addr
11	2	ACALL	code addr
12	3	LCALL	code addr
13	1	RRC	A
14	1	DEC	A
15	2	DEC	data addr
16	1	DEC	@R0
17	1	DEC	@R1
18	1	DEC	R0
19	1	DEC	R1
1A	1	DEC	R2
1B	1	DEC	R3
1C	1	DEC	R4
1D	1	DEC	R5
1E	1	DEC	R6
1F	1	DEC	R7
20	3	JB	bit addr,code addr
21	2	AJMP	code addr

Hex Code	Number of Bytes	Mnemonic	Operands
22	1	RET	
23	1	RL	A
24	2	ADD	A,#data
25	2	ADD	A,data addr
26	1	ADD	A,@R0
27	1	ADD	A,@R1
28	1	ADD	A,R0
29	1	ADD	A,R1
2A	1	ADD	A,R2
2B	1	ADD	A,R3
2C	1	ADD	A,R4
2D	1	ADD	A,R5
2E	1	ADD	A,R6
2F	1	ADD	A,R7
30	3	JNB	bit addr,code addr
31	2	ACALL	code addr
32	1	RETI	
33	1	RLC	A
34	2	ADDC	A,#data
35	2	ADDC	A,data addr
36	1	ADDC	A,@R0
37	1	ADDC	A,@R1
38	1	ADDC	A,R0
39	1	ADDC	A,R1
3A	1	ADDC	A,R2
3B	1	ADDC	A,R3
3C	1	ADDC	A,R4
3D	1	ADDC	A,R5
3E	1	ADDC	A,R6
3F	1	ADDC	A,R7
40	2	JC	code addr

figura 1 - Istruzioni OpCode elencati in ordine esadecimale (continua in figura 2).

```
110E 80F0       14                      SJMP  START
1110 75907F     15              SHIFT:  MOV   P1,#01111111b
1113 75B0E0     16                      MOV   P3,#11100000b
1116 12115E     17                      LCALL RIT
1119 7590BF     18                      MOV   P1,#10111111b
111C 75B0E0     19                      MOV   P3,#11100000b
111F 12115E     20                      LCALL RIT
1122 7590DF     21                      MOV   P1,#11011111b
1125 75B0E0     22                      MOV   P3,#11100000b
                23
                24              ;*** BREAK POINT N°1 *****
1128 C2B3       25                      CLR   P3.3
112A D2AA       26                      SETB  EX1
112C 00         27                      NOP
```

Hex Code	Number of Bytes	Mnemonic	Operands
41	2	AJMP	code addr
42	2	ORL	data addr,A
43	3	ORL	data addr,#data
44	2	ORL	A,#data
45	2	ORL	A,data addr
46	1	ORL	A,@R0
47	1	ORL	A,@R1
48	1	ORL	A,R0
49	1	ORL	A,R1
4A	1	ORL	A,R2
4B	1	ORL	A,R3
4C	1	ORL	A,R4
4D	1	ORL	A,R5
4E	1	ORL	A,R6
4F	1	ORL	A,R7
50	2	JNC	code addr
51	2	ACALL	code addr
52	2	ANL	data addr,A
53	3	ANL	data addr,#data
54	2	ANL	A,#data
55	2	ANL	A, data addr
56	1	ANL	A,@R0
57	1	ANL	A,@R1
58	1	ANL	A,R0
59	1	ANL	A,R1
5A	1	ANL	A,R2
5B	1	ANL	A,R3
5C	1	ANL	A,R4
5D	1	ANL	A,R5
5E	1	ANL	A,R6
5F	1	ANL	A,R7

Hex Code	Number of Bytes	Mnemonic	Operands
60	2	JZ	code addr
61	2	AJMP	code addr
62	2	XRL	data addr,A
63	3	XRL	data addr,#data
64	2	XRL	A,#data
65	2	XRL	A,data addr
66	1	XRL	A,@R0
67	1	XRL	A,@R1
68	1	XRL	A,R0
69	1	XRL	A,R1
6A	1	XRL	A,R2
6B	1	XRL	A,R3
6C	1	XRL	A,R4
6D	1	XRL	A,R5
6E	1	XRL	A,R6
6F	1	XRL	A,R7
70	2	JNZ	code addr
71	2	ACALL	code addr
72	2	ORL	C,bit addr
73	1	JMP	@A+DPTR
74	2	MOV	A,#data
75	3	MOV	data addr,#data
76	2	MOV	@R0,#data
77	2	MOV	@R1,#data
78	2	MOV	R0,#data
79	2	MOV	R1,#data
7A	2	MOV	R2,#data
7B	2	MOV	R3,#data
7C	2	MOV	R4,#data
7D	2	MOV	R5,#data
7E	2	MOV	R6,#data

figura 2 - Istruzioni OpCode elencati in ordine esadecimale (continua in figura 3).

```
                28                      ;*************************
112D 12115E     29                              LCALL RIT
1130 7590EF     30                              MOV   P1,#11101111b
1133 75B0E0     31                              MOV   P3,#11100000b
1136 12115E     32                              LCALL RIT
1139 7590F7     33                              MOV   P1,#11110111b
113C 75B0E0     34                              MOV   P3,#11100000b
113F 12115E     35                              LCALL RIT
1142 7590FB     36                              MOV   P1,#11111011b
1145 75B0E0     37                              MOV   P3,#11100000b
```

Hex Code	Number of Bytes	Mnemonic	Operands
7F	2	MOV	R7,#data
80	2	SJMP	code addr
81	2	AJMP	code addr
82	2	ANL	C,bit addr
83	1	MOVC	A,@A+PC
84	1	DIV	AB
85	3	MOV	data addr,data addr
86	2	MOV	data addr,@R0
87	2	MOV	data addr,@R1
88	2	MOV	data addr,R0
89	2	MOV	data addr,R1
8A	2	MOV	data addr,R2
8B	2	MOV	data addr,R3
8C	2	MOV	data addr,R4
8D	2	MOV	data addr,R5
8E	2	MOV	data addr,R6
8F	2	MOV	data addr,R7
90	3	MOV	DPTR,#data
91	2	ACALL	code addr
92	2	MOV	bit addr,C
93	1	MOVC	A,@A+DPTR
94	2	SUBB	A,#data
95	2	SUBB	A,data addr
96	1	SUBB	A,@R0
97	1	SUBB	A,@R1
98	1	SUBB	A,R0
99	1	SUBB	A,R1
9A	1	SUBB	A,R2
9B	1	SUBB	A,R3
9C	1	SUBB	A,R4
9D	1	SUBB	A,R5
9E	1	SUBB	A,R6
9F	1	SUBB	A,R7

Hex Code	Number of Bytes	Mnemonic	Operands
A0	2	ORL	C,/bit addr
A1	2	AJMP	code addr
A2	2	MOV	C,bit addr
A3	1	INC	DPTR
A4	1	MUL	AB
A5		reserved	
A6	2	MOV	@R0,data addr
A7	2	MOV	@R1,data addr
A8	2	MOV	R0,data addr
A9	2	MOV	R1,data addr
AA	2	MOV	R2,data addr
AB	2	MOV	R3,data addr
AC	2	MOV	R4,data addr
AD	2	MOV	R5,data addr
AE	2	MOV	R6,data addr
AF	2	MOV	R7,data addr
B0	2	ANL	C,/bit addr
B1	2	ACALL	code addr
B2	2	CPL	bit addr
B3	1	CPL	C
B4	3	CJNE	A,#data,code addr
B5	3	CJNE	A,data addr,code addr
B6	3	CJNE	@R0,#data,code addr
B7	3	CJNE	@R1,#data,code addr
B8	3	CJNE	R0,#data,code addr
B9	3	CJNE	R1,#data,code addr
BA	3	CJNE	R2,#data,code addr
BB	3	CJNE	R3,#data,code addr
BC	3	CJNE	R4,#data,code addr
BD	3	CJNE	R5,#data,code addr
BE	3	CJNE	R6,#data,code addr
BF	3	CJNE	R7,#data,code addr

figura 3 - Istruzioni OpCode elencati in ordine esadecimale (continua in figura 4).

```
1148 12115E     38                      LCALL RIT
114B 7590FF     39                      MOV   P1,#11111111b
114E 75B060     40                      MOV   P3,#01100000b
1151 12115E     41                      LCALL RIT
1154 7590FF     42                      MOV   P1,#11111111b
1157 75B0C0     43                      MOV   P3,#11000000b
115A 12115E     44                      LCALL RIT
115D 22         45                      RET
115E 7920       46              RIT:    MOV   R1,#20H
1160 7A08       47              RIT1:   MOV   R2,#08H
1162 7BFF       48              RIT2:   MOV   R3,#0FFH
1164 DBFE       49              RIT3:   DJNZ  R3,RIT3
```

Hex Code	Number of Bytes	Mnemonic	Operands
C0	2	PUSH	data addr
C1	2	AJMP	code addr
C2	2	CLR	bit addr
C3	1	CLR	C
C4	1	SWAP	A
C5	2	XCH	A,data addr
C6	1	XCH	A,@R0
C7	1	XCH	A,@R1
C8	1	XCH	A,R0
C9	1	XCH	A,R1
CA	1	XCH	A,R2
CB	1	XCH	A,R3
CC	1	XCH	A,R4
CD	1	XCH	A,R5
CE	1	XCH	A,R6
CF	1	XCH	A,R7
D0	2	POP	data addr
D1	2	ACALL	code addr
D2	2	SETB	bit addr
D3	1	SETB	C
D4	1	DA	A
D5	3	DJNZ	data addr,code addr
D6	1	XCHD	A,@R0
D7	1	XCHD	A,@R1
D8	2	DJNZ	R0,code addr
D9	2	DJNZ	R1,code addr
DA	2	DJNZ	R2,code addr
DB	2	DJNZ	R3,code addr
DC	2	DJNZ	R4,code addr
DD	2	DJNZ	R5,code addr
DE	2	DJNZ	R6,code addr
DF	2	DJNZ	R7,code addr

Hex Code	Number of Bytes	Mnemonic	Operands
E0	1	MOVX	A,@DPTR
E1	2	AJMP	code addr
E2	1	MOVX	A,@R0
E3	1	MOVX	A,@R1
E4	1	CLR	A
E5	2	MOV	A,data addr
E6	1	MOV	A,@R0
E7	1	MOV	A,@R1
E8	1	MOV	A,R0
E9	1	MOV	A,R1
EA	1	MOV	A,R2
EB	1	MOV	A,R3
EC	1	MOV	A,R4
ED	1	MOV	A,R5
EE	1	MOV	A,R6
EF	1	MOV	A,R7
F0	1	MOVX	@DPTR,A
F1	2	ACALL	code addr
F2	1	MOVX	@R0,A
F3	1	MOVX	@R1,A
F4	1	CPL	A
F5	2	MOV	data addr,A
F6	1	MOV	@R0,A
F7	1	MOV	@R1,A
F8	1	MOV	R0,A
F9	1	MOV	R1,A
FA	1	MOV	R2,A
FB	1	MOV	R3,A
FC	1	MOV	R4,A
FD	1	MOV	R5,A
FE	1	MOV	R6,A
FF	1	MOV	R7,A

figura 4 - Istruzioni OpCode elencati in ordine esadecimale.

```
1166 DAFA       50                                  DJNZ  R2,RIT2
1168 D9F6       51                                  DJNZ  R1,RIT1
116A 22         52                                  RET
                53                                  END
                54
                55

SHIFT              Thu May 22 09:27:09 1997           PAGE 2

SYMBOL TABLE LISTING

N A M E    T Y P E   V A L U E       A T T R I B U T E S


ADDR.......    NUMB      1000H A
EX1........ B ADDR     00A8H.2 A
P1......... D ADDR       0090H A
P3......... D ADDR       00B0H A
RIT........ C ADDR       115EH A
RIT1....... C ADDR       1160H A
RIT2....... C ADDR       1162H A
RIT3....... C ADDR       1164H A
SHIFT...... C ADDR       1110H A
START...... C ADDR       1100H A

REGISTER BANK(S) USED: 0

ASSEMBLY COMPLETE, NO ERROR FOUND
```

Come si può osservare sono stati inseriti due BREAK POINT rispettivamente alla riga 24 e 7. Per ottenere un B.P. è necessario seguire la sintassi proposta oppure la sua copia utilizzando il P3.2. In quest'ultimo caso si dovrà scrivere:

```
CLR   P3.2
SETB  EX0
NOP
```

Nel caso presente nel programma si utilizzerà nelle operazioni di simulazione l'opzione **Single**

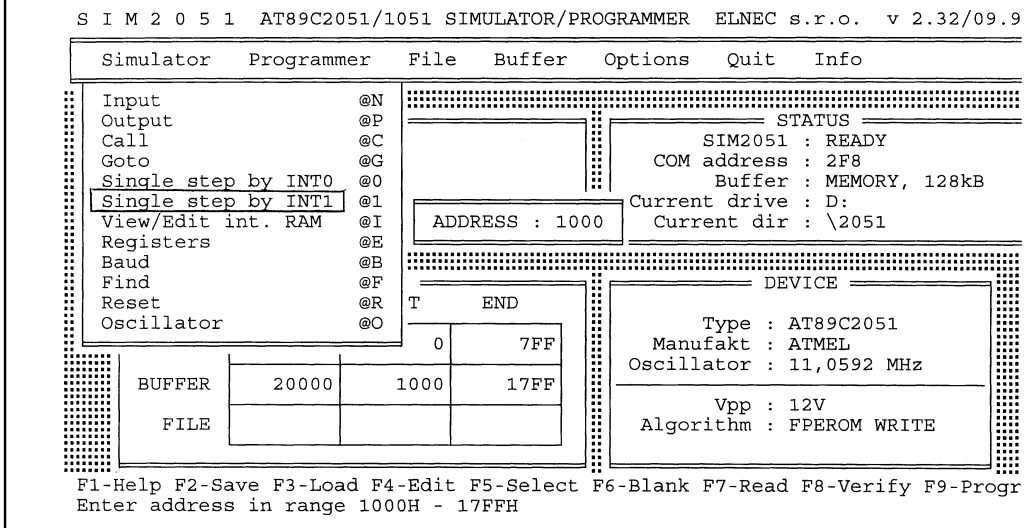


figura 5

Step by INT1, mentre nel caso visto sopra si utilizzerà il comando **Single Step by INT0.**

Ammesso di avere completato tutte le istruzioni si procederà allo svolgimento dei comandi secondo quanto visto nei numeri scorsi e cioè:

1) Dare al file il nome con estensione SRC ad esempio**SHIFT.SRC**
2) Digitare **MA51 SHIFT**
3) Controllare eventuali errori richiamando il file **SHIFT.LST**
4) Quando tutto è a posto lanciare il file BAT (vedi nella 3ª puntata) che chiameremo 2051.bat così: **2051 SHIFT**
5) Alla comparsa della pagina di simulazione caricare il file e portarsi su Simulator scegliendo l'opzione **Single Step by INT1** e premere invio due volte (figura 5).
6) Dopo qualche istante dalla premuta del secondo invio del punto 5 comparirà una videata corrispondente alla simulazione step. Si potrà procedere passo-passo tramite il tasto invio, oppure si potrà lanciare il programma tramite il comando **@U** che si avrà premendo i 3 tasti **Alt Gr - @ - U**.
7) Il programma girerà fino al primo Break Point per ripartire solo un nuovo comando **@U**.
8) Verificato ciò si potrà uscire dallo stato di step premendo il tasto ESC. Nelle figure 6 e 7 sono visibili gli stati dei registri dopo il primo (figura 6) e il secondo (figura 7) B.P.
9) Uscendo dallo step confermare l'opzione reset e poi lanciare il programma con GOTO.
10) Se tutto è confermato cambiare con 0000h l'origine al file e programmare il microcontrollore seguendo le istruzioni pubblicate nella terza puntata.

Esecuzione finale

Con il circuito test si è potuto controllare il software ma se l'applicazione non prevede che 8 LED conviene modificare il circuito realizzandone uno dedicato. Non ritenendo conveniente una applicazione così riduttiva mi limiterò a

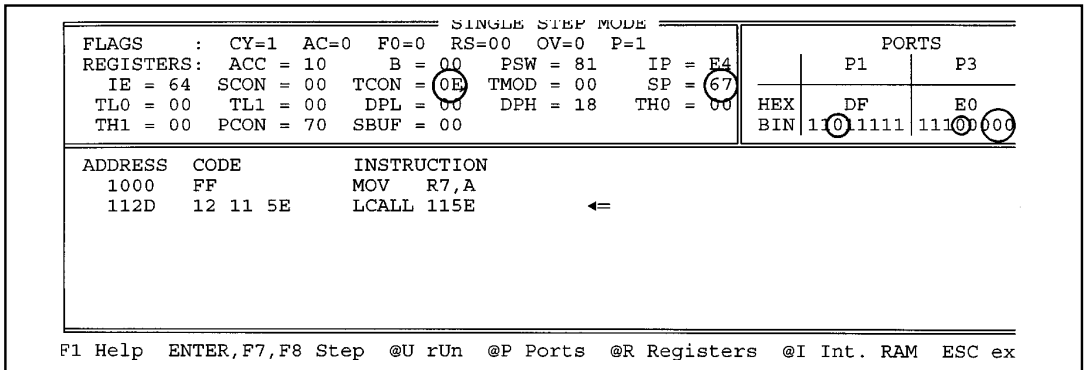


figura 6

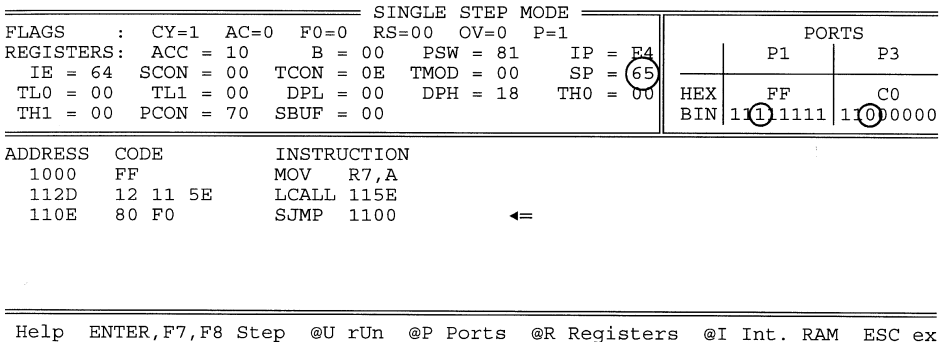


figura 7

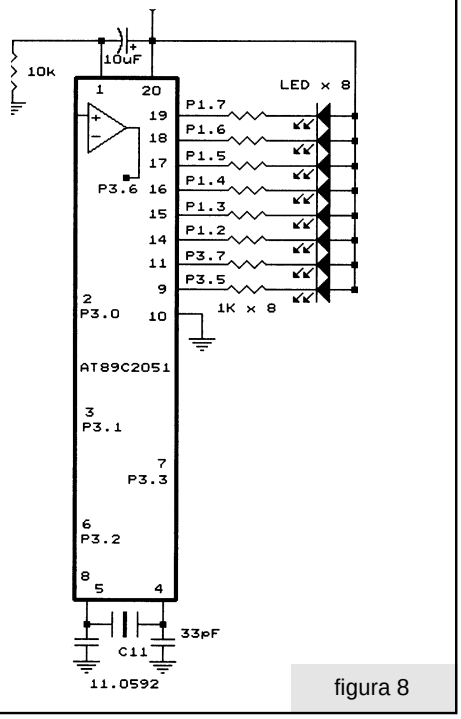


figura 8

proporre solamente il circuito elettrico come da figura 8.

Nell'attesa di proporre kit dedicati rimane validissimo il kit di test che proponiamo di seguito.

Reperibilità e costi

Anche se verranno presentati nei prossimi numeri ritengo opportuno comunicare i prezzi dell'intero sistema per consentire a coloro che fossero interessati di potersi eventualmente regolare di conseguenza.

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

Ai prezzi sopra riportati occorre aggiungere le spese di spedizione.

Per qualsiasi richiesta e/o informazioni rivolgersi a Nello Alessandrini tramite la Redazione.