

PROGRAMMAZIONE SINTETICA

Durante gli anni di utilizzo delle calcolatrici programmabili della Texas Instruments TI-57/58/58C/59 ed, in parte, 66 con varie sperimentazioni e man mano che progrediva la conoscenza dell'hardware delle macchine si è potuto allargare i confini di applicazione delle calcolatrici stesse: quella che in gergo è stata chiamata "programmazione sintetica". In queste note si presentano alcuni di questi risultati.

I REGISTRI INTERNI DELLA TI59/58C/59

Poiché verrà fatto spesso riferimento agli 8+8 registri dei TMC0582 e TMC0583 eccone una rappresentazione grafica esplicativa:

	Digit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0			
TMC0582 Registers	0	10 User Flags					0	0	0	RAM Address Second ROM Address			Byte Prog SRC FLG	Last Key		Fix ptd		Hierarchy Stack		
	1	Mantissa															Hierarchy Stack			
	2																			
	3																			
	4																			
	5																			
	6																			
	7																			
8																				
TMC0583 Registers	9	List of data flag	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	Hierarchy Stack	
	A	RAM or CONSTANT ROM PROGRAMCODES																		
	B	T-REGISTER Mantissa															Exponent	Sign		
	C	PC	OP	PC	OP	PC	OP	PC	OP	PC	OP	PC	OP	PC	OP	PC	OP	PC	OP	Opcode & Par. Count for Hierarchy stack
		1	1	2	2	3	3	4	4	5	5	6	6	7	7	8	8	8	8	
	D	Page in run	0	0	0	0	0	0	0	0	RAM memory min. address			RAM memory max. address			No of pending ops	Parenttheses count	Deg Rad Grd	
	E	Level 6 RAM Address Const &2nd ROM Address				Byte NR	Prog SRC FLG	Level 5 RAM Address Const &2nd ROM Address				Byte NR	Prog SRC FLG	Level 4 RAM Address Const &2nd ROM Address				Byte NR	Prog SRC FLG	Sup Routine Stack
F	Level 3 RAM Address Const &2nd ROM Address				Byte NR	Prog SRC FLG	Level 2 RAM Address Const &2nd ROM Address				Byte NR	Prog SRC FLG	Level 1 RAM Address Const &2nd ROM Address				Byte NR	Prog SRC FLG		
																Cond. Rtn Flag	No of SBR levels			

Inoltre i TMC0582 e 0583 hanno anche 512 byte di memoria (256+256) nella quale sono memorizzate le routine statistiche e di conversione (da qui il nome "ROM statistica") e le costanti utilizzate per il calcolo delle funzioni trigonometriche e logaritmiche (secondo l'algoritmo CORDIC).

CODICI ESADECIMALI DELLA TI59/TI-58C/TI58

I codici di tasto delle TI-59/58C/58 sono composti da due cifre XY codificate in BCD e quindi comprese tra 0 e 9, il che dà 100 codici di tasto diversi.

Si possono generare 60 nuovi codici esadecimali e precisamente quelli **la cui seconda cifra Y è compresa tra A ed F**.

La visualizzazione avviene convertendo il codice in un formato numerico BCD nel seguente modo:

$$[X][Y]_{\text{hex}} \rightarrow [X+1][Y-10]_{\text{dec}} \quad [\text{se } X = 10 \text{ allora } X = 0]$$

Ad esempio il codice 7E viene visualizzato come 84 (tenendo conto che $E_{\text{hex}} = 14_{\text{dec}}$).

● Comportamento

I codici esadecimali hanno in generale le seguenti caratteristiche:

- 1) possono essere introdotti *solo nei passi multipli di 8*;
- 2) sono trasformati in codici "normali" se spostati con **Ins** o **Del**;
- 3) inibiscono la ricerca di una label posta dopo la loro posizione;
- 4) si comportano diversamente se la TI-59/58C/58 è attaccata alla stampante PC-100 o se questa è in **TRACE**;
- 5) se sono parte di una istruzione composta utilizzano il registro 00.

● Generazione

I codici esadecimali sono generati con un'opportuna sequenza - nell'esempio utilizziamo il modulo Master Library:

- 1) posizionamento al passo NNN
- 2) premere CLR Pgm 19 SBR 945 D.MS LRN Ins (una o più volte) RST

Il codice viene generato in funzione del numero di Ins da premere ed è stabilito dalla seguente formula:

$$\mathbf{A0 - (passo NNN della ROM statistica) + (passo NNN della RAM)}$$

Ad esempio il passo 008 della ROM statistica contiene il codice 04 per cui un Ins aggiunge la quantità $A0-04 = 9C$.

Il codice da mettere in RAM ed il numero di Ins da premere può essere calcolato dal seguente programma (tratto dalla rivista *MC-Microcomputer* n. 20).

```

LBL A - INT STO 01 = * 1 0 0 = STO 02 CLR STO 00 R/S - INT STO 03 = * 1 0 0
      = STO 04
LBL INV RCL 04 X:T RCL 02 GE LNX OP 31 + 1 6 =
LBL LNX - 1 0 X:T = OP 20 STO 02 GE INV
RCL 01 - RCL 03 * RCL 00 = CP
LBL CLR GE CE + 1 0 = GTO CLR
LBL CE * 1 0 + RCL 02 + RCL 00 / 1 0 = R/S

```

Esempio di utilizzo: generare il codice 4E al passo 112.

4.14 A

e sapendo che al passo 112 si somma 5D

5.13 R/S → 91.1

cioè impostare 91 {R/S} in RAM e premere una volta INS nella sequenza di generazione.

In pratica se in RAM abbiamo valore 00 la formula si riduce a:

A0 - valore ROM statistica

Ad esempio al passo 112 c'è 43 quindi $A0 - 43 = 5D$

Ad esempio al passo 000 c'è 82 quindi $A0 - 82 = 1E$

000 --> 1E
 008 --> 9C
 016 --> 4B
 024 --> 4B
 032 --> 3B
 040 --> 4C
 048 --> 4D
 056 --> 6E
 064 --> 6E
 072 --> 9D
 080 --> 9D
 088 --> 5D
 096 --> 4D
 104 --> 4C
 112 --> 5D
 152 --> 9A

ecc...

TABELLA RIASSUNTIVA

Y→ ↓X	A	B	C	D	E	F
0	10 [2]	11 [2][12]	12 [2]	13 [2]	14 [2]	15 [2]
1	20	21	22	23	24 [11]	25 [3]
2	30 CLR	31 CLR [1]	32 TLR [10]	33 _X: [10]	34 CLR	35 CLR
3	40 TAN	41 TAN [1]	42 V2N	43 XIN	44 TAN [9]	45 TAN [10]
4	50	51 [1]	52 BS [9]	53 EE [10]	54 [6]	55 [7]
5	60 X 	61 X [9]	62 OSS	63 LST [7]	64 X [10]	65 X [9]
6	70 DEG	71 DEG [10]	72 *SB [10]	73 *ST [4]	74 DEG [7]	75 DEG [9]
7	80 RAD	81 RAD [7]	82 *GT	83 *PG [10]	84 RAD	85 RAD [9]
8	90 GRD	91 GRD	92 NR/ [9]	93 [7]	94 GRD [5]	95 GRD [8]
9	00	01	02	03 [9]	04 LST [10]	05

In blu i mnemonici stampati da PC-100 durante il “trace”, tra parentesi quadrate il numero di nota a pagina seguente.

NOTE ALLA TABELLA

[1] 2b, 3b, 4b è la funzione $\sin(x)$ mentre INV 2b, 3b,4b fornisce il risultato di una *funzione non nota* che può essere comunque approssimata dalla funzione $\arctan(x) * e^{P(x)}$ dove $P(x)$ è un polinomio di grado pari con termine noto nullo. Il polinomio $P(x)$ è stato calcolato dall'autore fino al grado $n=12$ e fornisce risultati esatti mediamente fino alla quinta cifra decimale.

[2] Impostano il corrispondente valore in decimale nell'esponente. **0C (h12)** viene utilizzato per il passaggio **da programma** al cosiddetto "*fast mode*" che consente, al prezzo di talune limitazioni, di raddoppiare praticamente la velocità delle TI via software. Tale modalità operativa è quella utilizzata nei moduli CROM.

[3] Modo **grafico** per la stampante PC-100 (come esempio di utilizzo vedi il programma *PLOT60*): h25 blocca la stampa di un carattere alla prima o seconda riga di pixel (dipende dall'hardware) .

[4] Incrementa di 1 il fix dei decimali, aggiornando il registro 0 di TMC0582.

[5] Arrotonda come EE-INV-EE e divide il per $10^{(12-n)}$ dove n è il numero di cifre decimali del numero.

[6] Richiama il registro 0 di TMC0582. La sequenza **x:t 4E** copia il registro t nel display.

[7] Codici a 4 byte (con funzione non determinata)

[8] Codice con vari significati:

8F N7 XX YY → GTO XXYY

8F N3 XX YY → GTO XXYY

8F N6 XX → PRD XX

8F N2 XX → PRD XX

8F N0 XX → PRD XX (se $XX > 9$ PRD 00, se $XX = 40$ PRD IND YY)

8F N1 XX → ABS

8F N5 XX → ABS

8F N9 XX → ABS

8F N4 XX → ABS

[9] Eseguono un RCL XY.

[10] Reset della TI.

[11] Trasforma i successivi 7 byte nel modo seguente:

1E AB CD EF GH IJ KL MN → DA FC HE JG LI MK ON

[12] Se incontrato durante un'istruzione di List genera la cosiddetta "stampa circolare" (vedi "ROM statistica" della TI-59).

Programmazione sintetica sulla TI 57.

Le istruzioni “segrete” si dividono in due gruppi, a seconda che si possano o meno introdurre da tastiera.

- A) Funzioni introducibili da tastiera. L'unica funzione appartenente a questo classe è “SBR” senza numero di etichetta, impostabile premendo SBR SST. Essa agisce così: calcola l'esponente che avrebbe il numero contenuto nel visualizzatore se fosse in notazione esponenziale, poi passa al primo passo di programma che corrisponde all'esponente; se questo è un numero maggiore di 9, in valore assoluto, si ha errore. Facciamo un esempio: sul display è contenuto il numero 2550, e cioè $2.5 \text{ E } 03$; la calcolatrice cerca il primo passo di programma che contiene il numero “3”. Se si imposta il programma *SBR (SST) 0 1 2 3 4 5 6 7 8 9 = R/S* e si preme *2500 RST R/S* apparirà *456789*. Se si preme *1000000 RST R/S* apparirà *789*. Questa istruzione costituisce una specie di SBR indiretto: la sequenza $n \cdot 10^x$ *SBR (SST)* salta al primo passo di programma contenente il numero n.
- B) Funzioni non direttamente introducibili da tastiera. La maggior parte delle istruzioni nascoste sulla TI 57 è di questo tipo: il fatto di non poter agevolmente introdurre un qualunque byte, some succede invece nelle TI 58 e TI 59 (ad esempio l'istruzione HIR), costringe a “salti mortali” da parte del programmatore. Infatti è in generale necessario introdurre un piccolo *programma generatore*, a partire dal passo 00, programma che contiene una sequenza alquanto anomala di istruzioni:

Exc (SST) Lbl 5 sequenza generatrice R/S

dove il tasto SST è premuto per generare un “Exc” senza indice e la “sequenza generatrice” è una di quelle riportate in tabella:

N. \ s.g.	3 =	+/-	3 +/-	3 +/- EE	. 3 =	EE 3	.3 +/- =
0	8	11	-11	12	-12	13	-13
1	9	21	-21	22	-22	23	-23
2	A	31	-31	32	-32	33	-33
3	B	41	-41	42	-42	43	-43
4	C	51	-51	86 7	51 7	86 8	51 8
5	D	61	-61	86 4	51 4	86 5	51 5
6	E	71	-71	86 1	51 1	86 2	51 2
7	F	81	-81	86 0	51 0	83	-83
8	8	8	8	8	8	8	8
9	9	9	9	9	9	9	9

Successivamente si preme LRN (per uscire dal modo ai apprendimento), RST R/S (si elabora il programmino), LRN (si ritorna alla programmazione della 57) ed un qualsiasi tasto numerico: si sarà così generato il codice indicato sulla tabella.

Facciamo un esempio: impostiamo il programma

Exc (SST) Lbl 5 3 = R/S

dove “3 =” è la prima sequenza generatrice: premiamo poi LRN. Abbiamo in questo modo introdotto il programma generatore e a seconda di ciò che faremo da ora in poi avremo risultati differenti. Premiamo ora RST R/S LRN: sul display vedremo 05 00 0, come se dovessimo inserire un codice composto. Premiamo ora 3 e successivamente BST per andare a vedere che cosa abbiamo impostato in realtà. Come per incanto comparirà il codice “11”: per vedere che

cosa fa premiamo LRN per uscire dal modo di apprendimento e poi SST. Sul display apparirà una “b” minuscola

Infatti il numero 11 in esadecimale è proprio “b” ! Se ripetiamo la stessa cosa coi numeri 2 3 4 5 6 7 comparirà il codice che si riferisce alle lettere **A b C d E F** esadecimali.

Questi codici, una volta generati, possono essere spostati usando le funzioni Ins e Del, e possono essere usati per scritte sul visualizzatore, oppure opportunamente inseriti in un programma di conversione decimale-esadecimale. Tra l’altro far scrivere la parola “CIAO” alla calcolatrice è diventata una bazzecola !

C’è però una stranezza: se il codice viene eseguito mentre il display non è azzerato e si tratta della prima “cifra”, viene impostata la notazione esponenziale. Ad esempio il codice 14 da “E” se il contenuto del display è zero, “4 EE 00” altrimenti.

- C) Esempio n° 2. Se si usa la sequenza generatrice “+/-“ si ottengono vari codici, tra cui quelli anomali sono 11, 21, 31, 41 ottenuti premendo i tasti numerici 0, 1, 2, 3. Il codice “11” è apparentemente uguale al codice 11 esadecimale, mentre esplica una funzione diversa (evidentemente esiste una differenza non visualizzabile). Provando a eseguire un’operazione (ad esempio $14 \times 2 = 28$) e premendo CLR per cancellare il display, se si esegue il codice “11” è l’istruzione “=” nel visualizzatore torna il risultato che era stato cancellato e cioè 28. Questo numero resta da qualche parte ed il codice “11” lo recupera, analogamente all’istruzione “Last x” delle HP. *[In realtà recupera il valore del registro “C” di TMC1501- l’eventuale segno ‘-’ di mantissa e/o esponente viene perso].* Il codice “21” quando viene incontrato nel corso dell’esecuzione del programma (cioè con R/S e non con SST) il display passa in LRN ed il contatore di programma comincia ad avanzare velocemente distruggendo il programma stesso.
- D) Usando la sequenza generatrice “3 +/- EE” si genera un solo codice anomalo, il “12”, ancora una volta da non confondere con l’analogo codice “12” esadecimale. La sua esecuzione moltiplica per una potenza di 10 il numero contenuto nel display. *[Tale esponente è così calcolabile: detto n il numero di cifre visualizzate, a prescindere dalla virgola, l’esponente vale $-(10 - n - |vecchio EE|)$ e lo si applica sul valore assoluto del numero originale: ad es. π^{12} vale 0.0314159 perché in questo caso “vecchio” EE = 0 (il numero non è in notazione esponenziale), $n = 8$ e quindi $esp = -(10 - 8) = -2$ cioè $3.1415927 E^{-2} = 0.0314159$ con gli arrotondamenti uguali a quelli dell’istruzione EE].* Gli altri codici ottenibili con le sequenze riportate in tabella sono o gli inversi dei comandi già visti, oppure dei codici del tutto normali, tranquillamente impostabili da tastiera. Rimangono ancora “misteriosi” molti codici: 16, 17, 37, 47 e tutti quelli corrispondenti alla posizione di tasti numerici.

“Memoria costante” per la TI 57. Ormai chi riteneva che la TI 57 fosse una calcolatrice ben conosciuta in tutte le sue caratteristiche si deve ricredere: già la volta scorsa abbiamo visto come generare alcune funzioni “nuove”, non impostabili direttamente da tastiera, e addirittura come far comparire le prime sei lettere dell’alfabeto sul display, che insieme alle cifre da 0 a 9 formano i caratteri esadecimali.

Questa volta attingiamo alcune notizie dall’estero, per la precisione dalla rivista “L’Ordinateur de Poche”, la quale, come dice il nome, si occupa delle calcolatrici programmabili, in generale le TI, le HP le Sharp, le Casio ecc. Da un articolo ricaviamo un metodo per spegnere il display della TI 57, lasciando viceversa alimentato l’unico circuito integrato. E’ ben noto che la parte preponderante del consumo di una calcolatrice con display a LED è proprio causata da quest’ultimo, anche se si utilizzano tecniche di multiplexaggio il consumo dei LED risulta sempre elevato, specie a confronto di quello del display LCD.

Ecco che, riuscendo a spegnere il display, senza viceversa spegnere la calcolatrice con l’interruttore, si ottiene innanzitutto una riduzione dei consumi ma come conseguenza notevolissima si ha che in questo modo la calcolatrice stessa manterrà memorizzate le informazioni contenute nelle memorie e soprattutto il programma: trasformeremo così una TI 57 in una “TI 57C”, cioè con

memoria costante. Vediamo ora come si realizza questo “interruttore software”: le istruzioni da impostare sono analoghe a quelle viste nello scorso numero per la generazione di nuove funzioni.

Ancora una volta la chiave è la sequenza

Exc (SST) Lbl 1

dove (SST) serve ad eliminare l’operando dell’istruzione *Exc* e cioè il numero del registro: in questo modo si ottengono due passi di programma consecutivi contenenti rispettivamente 48 (il codice dell’*Exc* solitario) e 86 1 (codice di *Lbl 1*), che evidentemente vanno proprio a “scavare” nelle lacune del sistema operativo della TI 57.

Ora questa sequenza terminata con un R/S può essere posta all’inizio della memoria di programma (passo 00) cioè prima del programma che abbiamo caricato e che vogliamo mantenere, oppure alla fine del programma stesso, a partire dal passo 47. Supponiamo perciò di aver introdotto un certo programma; premendo da tastiera GTO 2nd 47 LRN entriamo in modo di “apprendimento” e introduciamo la sequenza *Exc (SST) Lbl 1 R/S*.

Dato che abbiamo così usato tutti i 50 passi previsti, la calcolatrice uscirà automaticamente dal modo LRN. Ora da tastiera facciamo elaborare questa mini-sequenza, premendo SBR 2nd 47: sul display avremo ancora 0. Ora premiamo *INV STO 3*. È facile constatare l’assurdità di questa sequenza in quanto l’istruzione *STO* non ha una “inversa”: sul display, inaspettatamente, comparirà soltanto un segno “-” posto nel secondo LED da destra; premendo il tasto “+/-” questo “-” si sposterà di una posizione verso destra e premendo ancora una volta “+/-”, scomparirà. Il display è così spento. La TI 57 è invece ancora accesa, tanto è vero che premendo a caso alcuni tasti può capitare di veder riaccendere il display: la calcolatrice però sembrerà impazzita, tanto da non riuscire più ad effettuare calcoli coerenti. Per ripristinarne il funzionamento, senza ovviamente spegnerla e riaccenderla, basta premere “*INV Fix*”; si potrà verificare che il programma è ancora lì, come pure le memorie.

Vediamo ora la faccenda dei consumi: con un milliamperometro in serie all’alimentazione abbiamo misurato alcuni valori. Se sul display compare solo lo “0” lo strumento misurerà circa 13 mA, che salgono vertiginosamente non appena si impostano delle cifre. In particolare con tutti i segmenti accessi (“-8.888888-88”) si hanno più di 55 mA. Tali valori non cambiano anche se introduciamo in memoria un programma e se riempiamo tutti i registri. Usando invece l’“interruttore software” si ottiene un consumo di 8 mA.

Conclusione: per mantenere i dati memorizzati si ottiene un risparmio di circa il 40% rispetto alla soluzione di lasciare accesa “tutta” la calcolatrice.

Altri codici sintetici sulla TI-57. Nell’Angolo n° 8, che parlava della creazione di codici sintetici sulla TI-57, si terminava l’articolo invitando a scoprire le sequenze generatrici relative ai codici 16, 17, 37, 47 e a quelli corrispondenti alla posizione dei tasti numerici. Per questi ultimi non c’è niente da fare: quando ci si aspetta che vengano fuori, escono sempre altri codici composti; riguardo ai primi 4 codici citati, li abbiamo trovati tutti (anzi, abbiamo trovato i loro inversi ma, a quanto sembra, in questi casi dovrebbe essere la stessa cosa.

Il primo, il codice 16, è ottenibile usando come sequenza generatrice 3:3:3: e “0” come numero da impostare successivamente; il suo funzionamento è uguale a quello del codice “11”.

I codici 47, 37 e 17 vengono generati dalla sequenza 3:3:3:+/- e dai numeri, rispettivamente “3”, “2” e “0”: i primi due sono uguali come funzioni esplicabili, all’istruzione *Nop*; il “17” è del tutto simile al codice “12”.

Fix esteso sulla TI 57. Oltre alle normali operazioni di *Fix* da 0 a 9 esistono altre ben 6 operazioni di *Fix*. La cosa più interessante è che, oltre a questa possibilità, possiamo usarle anche per spegnere il display in un modo particolare e molto malleabile, il display della TI 57. Le istruzioni sono: *Fix A*, *Fix b*, *Fix C*, *Fix d*, *Fix E*, *Fix F*. L’istruzione *Fix F* è ben nota perché serve a fare un *Fix -1* e serve inoltre per spegnere il display, o meglio a far comparire un segno “-”. Le altre si comportano, partendo da *Fix A*, come: *Fix -6*, *Fix -5*, *Fix -4*, *Fix -3*, *Fix -2*, *Fix -1*. Consideriamole una ad una.

La prima, *Fix -6 (Fix A)* si comporta come nel seguente modo: se nel display c'è un numero minore di 99999, viene visualizzato il numero come se ci fosse un *Fix 7* o un *Fix 8*. Se c'è un numero compreso tra 100.000 e 499.999 il display si spegne completamente, Se c'è un numero maggiore di 499.999, c'è l'arrotondamento al milione immediatamente superiore o inferiore del numero. La cosa più interessante è che se vogliamo, possiamo memorizzare il numero 100.000 in una qualsiasi memoria e quando vorremmo spegnere il display basterà richiamarla (a patto però che non sia stato modificato il *Fix*).

Tutti gli altri, fino al *Fix d*, si comportano come quello descritto sopra, solo che il numero per spegnere il display deve essere compreso fra 10.000 e 49.999 per il *Fix b*, fra 1.000 e 4.999 per il *Fix C*, fra 100 e 499 per il *Fix d*. Naturalmente se il numero è inferiore al limite viene lasciato così com'è, mentre se è maggiore è arrotondato a seconda del fissaggio in uso. Un discorso a parte merita il *Fix E*: come condizioni di arrotondamento è come gli altri, nel senso che arrotonda al centinaio superiore o inferiore il numero nel display, e lo lascia intatto se minore di 10. La cosa diversa consiste nello spegnimento del display: mentre negli altri *Fix* si spegneva completamente, ora se il numero è compreso tra 10 e 49 si avranno a destra del display (al posto delle cifre destinate all'esponente per intenderci) i caratteri "0-". Le utilizzazioni di queste nuove istruzioni, oltre al già troppo volte citato spegnimento del display, sono molto varie: possiamo usarle per arrotondamenti mediante la sequenza EE INV EE e poi moltiplicarle per il fattore di arrotondamento, oppure spegnere il display quando un risultato superi o eguagli una certa quantità.

[Segue poi una serie di considerazioni, abbastanza inutili a dire il vero, sui codici "21", e "11" e la scoperta - errata - che il fattore di moltiplicazione del codice "12" è 1E-9].