The Buried Gold in the SR-52

Clif Penn
911 Northlake
Richardson TX 75080

About the Author

Clif Penn reports on these features as an enthusiastic user of the SR-52. Though he is employed by TI's Central Research Laboratories in Dallas, he wrote the article as an individual user, and much of it is based upon information passed around by the SR-52 users' grapevine in Dallas.

In the April 1976 issue of BYTE, a good overview comparison of the programmable SR-52 and the HP-65 was presented by Bradley Flippin [page 36]. Now some hidden but powerful features of the SR-52 organization will be discussed. At this time these features have not yet appeared in the literature Texas Instruments supplies with the SR-52, but the capabilities are worth documenting for readers who use this calculator.

Register Organization

The SR-52 is arranged internally with 100 programmable registers numbered 00 through 99. The first 20 registers (00 to 19) are those normally used for data storage and called from the keyboard such as RCL 06, STO 19, *EXC 05 and so on. All 20 of these user data registers are cleared simultaneously by pressing *CMs. [All secondary functions are shown with an asterisk (*) convention rather than writing (2nd) (CMs).] Many users have discovered by accident that there are other registers which may be accessed from the keyboard but have an incomplete understanding of how to take advantage of them. Table 1 shows a detailed listing of the registers, their conventional use and how they are cleared.

Register	Normal Use	Clearing Function
00 - 19	Data Storage	*CMs
20 - 59	Internal and not externally available	—
60 - 69	Operational stack	CLR
70 - 97	Program storage with 8 program steps per register	Affected by program edit
98 - 99	None	None

Table 1: SR-52 Register Organization. The documentation of the SR-52 mentions user programmable data storage in registers 00 to 19. In fact, the internal organization of the machine has a total of 100 registers allocated according to this map. Registers 60 to 69 are the operational stack used in parsing algebraic data entry (see BYTE's February 1976 issue for two articles on the subject). Registers 70 to 97 normally store the calculator's program with 8 program steps per register. Registers 98 and 99 are "free" and can be used for temporary data storage or as a flag. (The registers from 20 to 59 are not available for user programs.)

All of the registers except the internal 20 through 59 can be used in exactly the same way as the conventional 00-19, that is, indirectly addressed, conditionally addressed and so on.

Operational Stack Registers 60-69

If you use a "0" strike over rather than the clear button, registers 60 to 69 become available. Remember, however, that any time a "(" is actually necessary in your program, you use these registers in order from the bottom up. It is rare to use all levels of internested brackets, so start from 69 and work down if you need extra storage. (When in doubt, do the problem manually and RCL the register of interest and check for 0 contents.)

Program Storage Registers 70-97

The program storage registers 70 through 97 normally store the program at 8 steps per register. These registers are loaded either manually or when you read a preprogrammed magnetic card. In addition they are recorded on the magnetic card in the WRITE mode. This allows you to store data on a magnetic card for later use or updating. The statistics program used as an example incorporates this feature. Any time you delete or insert program steps *after* storing data in the

program memory, you will alter the register contents. BEWARE! To make life easier, table 2 shows the program locations normally stored in each register.

If you use this feature regularly, here are some memory aids – 8 program steps per register; the first register stores locations starting with 000 (ending with 007 so register 70 may be associated with this); register 80 starts with location 080; register 97 is last.

Reg	Loc	Reg	Loc	Reg	Loc
70	000 - 007	80	080 - 087	90	160 - 167
71	008 - 015	81	088 - 095	91	168 - 175
72	016 - 023	82	096 - 103	92	176 - 183
73	024 - 031	83	104 - 111	93	184 - 191
74	032 - 039	84	112 - 119	94	192 - 199
75	040 - 047	85	120 - 127	95	200 - 207
76	048 - 055	86	128 - 135	96	208 - 215
77	056 - 063	87	136 - 143	97	216 - 223
78	064 - 071	88	144 - 151		
79	072 - 079	89	152 - 159		

Table 2: Program Storage Registers and Locations Stored. This table gives the correspondence between program step numbers and register locations 70 to 97. Note that editing operations shift program data throughout this region, so any use of the program storage registers for data should be avoided when editing programs.

Bonus Storage Registers 98-99

There are numerous cases when you wish access to the CLR and *CMs feature without losing a constant you may be using regularly. Registers 98-99 are quite useful for this. None of the clearing functions affect them. Power off of course kills everything. You may encounter cases (as on the included programs) where you desire the effect of a flag but are still free to use CLR, *CMs, and reset. Although not nearly as efficient as flag usage, you can simulate a flag condition by storing any number in register 98 or 99 and use the following sequence:

. . . . .	
. . . . .	*LBL
*EXC	*1'
9	*EXC
9	9
* if zro	9
*1'	.
*EXC	.
9	.
9	.
.	.
.	.

Just as flag usage, this preserves the data in the display register for further use following the conditional branch instructions. Keep in mind you may need to clear 98 or 99 as an

A Note About Special Features to Save Program Steps

As a preamble let me emphasize forcefully that short routines should be written with parentheses in normal algebraic form without worrying about the "bells and whistles." This will use more program steps than needed but less time in programming and debugging.

Invariably you will encounter long programs where you need every "twist of the screw" to reduce program steps. The main thing to master is the algebraic hierarchy (pages 46-48 in the *SR-52 Owner's Manual*). Except as altered by parentheses, the order of operations is:

1. Immediate function evaluations (sin, cos, tan, etc).
2. Exponentiation and root extraction (x^2, y^x, $\sqrt{x}$ $\sqrt[x]{y}$).
3. Multiplication and division.
4. Addition and subtraction.
5. Perform operations from left to right on each hierarchy level.

For example:

$$(5 \times 7) + (8 \div 2) = 39$$

with or without the parentheses while

$$(5 + 7) \times (8 - 2) = 72$$

but

$$5 + 7 \times 8 - 2 = 59.$$

One equal sign at the end of the equation may replace several right parentheses one might require at the end of an expression.

Another useful feature involves recalling the display register contents by the use of either math functions or memory functions.

$3 + \sqrt{3}$ = may be programmed $3 + \sqrt{x}$ =
3 + 1/3 may be written $3 + 1/x$ =
3 + 3 = *cannot* be keyed 3 + =
but rather 3 + RCL = 6.

Any of the memory functions may be used in this "dummy instruction" manner. On occasion you may wish to store an intermediate result at the very same time as you use the display register contents like this:

4 x 5 + STO 01 x 3 = 80,
20 stored in register 01.

The "+" sign causes the first multiplication to take place, the STO inserts the display register contents back in the equation and the 01 directs 4 x 5 = 20 to be stored in register 01. Had you wanted 5 stored instead, you could have used a dummy memory instruction – – –

4 x 5 STO 01 + RCL x 3 = 80,
5 stored in register 01.

SR-52 User Instructions

TITLE STANDARD DEVIATION (6 BINS) PAGE 1 OF 2

◄A				◄B			
	D/A		BIN #	MEM ⇄			
AVE	DEV	−Σ	LAST x	ENTER			

STEP	PROCEDURE	ENTER	PRESS	DISPLAY
	OPERATING REGISTERS ARE ALWAYS R01, R02, R03. R01 STORES THE SUM OF ALL ENTRIES; R02 STORES THE SUM OF THE SQUARES OF ALL ENTRIES; R03 STORES THE NUMER OF DATA ENTRIES.			
	$AVE = \frac{R01}{R03} = \frac{\Sigma x}{n}$			
	$STD\ DEV = \sqrt{\frac{1}{n-1}\left[\Sigma x^2 - \frac{(\Sigma x)^2}{n}\right]}$			

SR-52 Coding Form

TITLE STD. DEVIATION (6 BINS) PAGE 1 OF 2
PROGRAMMER CLIF PENN DATE 5/19/76

LOC	CODE	KEY	COMMENTS	LOC	CODE	KEY	COMMENTS	LOC	CODE	KEY	COMMENTS	LABELS
000 112	46	*LBL			03	3	DISPLAY		03	3		A AVE
	15	E	ENTER		43	RCL	NEW		75	−		B DEV
	42	STO	DATA	040 152	00	0	n		01	1		C −Σ
	00	0	LAST		03	3			95	=		D LAST x
	00	0	ENTRY		81	HLT		080 192	30	*√x		E ENTRY
005 117	44	SUM			46	*LBL			42	STO	STD	A
	00	0	Σx		11	A	AVE		09	9	DEV	B DEV/AVE
	01	1		045 157	43	RCL			09	9		C
	40	*x²			00	0	Σx		81	HLT		D BIN #
	44	SUM			01	1	n	085 197	46	*LBL		E MEM ⇄
010 122	00	0	Σx²		55	÷			17	*B′	DEV/AVE	REGISTERS
	02	2			43	RCL			43	RCL		00 LAST x
	01	1		050 162	00	0			09	9		01 Σx
	44	SUM	NUMBER		03	3			09	9	DEV	02 Σx²
	00	0	ENTRIES		95	=		090 202	55	÷		03 n
015 127	03	3	n		42	STO			43	RCL		04 BIN
	43	RCL			01	1	AVE		01	1	AVE	05 #
	00	0	DISPLAY	055 167	09	9			09	9		06 2
	03	3	n		81	HLT			95	=		07 BIN
	81	HLT			46	*LBL		095 207	81	HLT		08 #
020 132	46	*LBL			12	B	DEV		46	*LBL	MEMORY	09 3
	13	C	−Σ		43	RCL			10	*E′	BIN	10 BIN
	42	STO	ROUTINE	060 172	00	0	Σx²		60	*if flg	SHIFT	11 #
	00	0	TO		02	2			00	0	RESTORE	12 4
	00	0	REMOVE		75	−		100 212	01	1	TO	13 BIN
025 137	94	+/−	PREVIOUS		43	RCL			01	1	ORDER	14 #
	44	SUM	DATA		00	0	Σx		03	3		15 5
	00	0	FROM	065 177	01	1			42	STO	ENTER	16 BIN
	01	1	AVE		40	*x²	(Σx)²		09	9	NEW	17 #
	40	*x²	÷		55	÷		105 217	08	8	BIN	18 6
030 142	94	+/−	DEV		43	RCL			90	*if zro	#	19
	44	SUM			00	0	n		89	*3′	0=ERROR	FLAGS
	00	0		070 182	03	3			75	−	if	0 BIN SORT
	02	2			54	)			07	7	greater	1
	01	1			55	÷		110 222	95	=	than 7	2
035 147	94	+/−			53	(			80	*if pos	= ERROR	3
	44	SUM			43	RCL	n	TEXAS INSTRUMENTS INCORPORATED				4
	00	0		075 187	00	0						

SR-52 User Instructions

TITLE STANDARD DEVIATION (6 BINS) PAGE 2 OF 2

◄A					◄B			
	D/A	n	BIN #	MEM ⇄				
AVE	DEV	−Σ	LAST x	ENTER				

STEP	PROCEDURE	ENTER	PRESS	DISPLAY
	BIN CONTROL ROUTINE FIRST PRESS OF *E′ (USING BIN #4 AS AN EXAMPLE)			
	BIN # 1	BIN # 4		
	R01 Σx_1 ⇄	R10 Σx_4		
	R02 Σx_1^2 ⇄	R11 Σx_4^2		
	R03 n_1 ⇄	R12 n_4		
	RESULT DISPLAY = 4			
	R01 Σx_4	R10 Σx_1		
	R02 Σx_4^2	R11 Σx_1^2		
	R03 n_4	R12 n_1		
	NEXT PRESS OF *E′ RESTORES ORIGINAL DATA LOCATIONS ASSUMING NO DELIBERATE ACTION HAS RESET FLAG 0 (SUCH AS *rset OR INV *st flg 0)			

SR-52 Coding Form

TITLE STD. DEVIATION PAGE 2 OF 2
PROGRAMMER CLIF PENN DATE 5/19/76

LOC	CODE	KEY	COMMENTS	LOC	CODE	KEY	COMMENTS	LOC	CODE	KEY	COMMENTS	LABELS
000 112	89	*3′			09	9	EXC'S			C′		A
	43	RCL			09	9	R01			RCL	Find	B
	09	9	LOCATES	040 152	36	*IND	R02			0	n	C
	08	8	NEW		48	*EXC	R03			3	after	D
	65	×	BIN		00	0	WITH	080 192		HLT	mem	E
005 117	03	3	ADDRESS		00	0	R0X				shift	A
	42	STO			36	*IND	R0Y					B
	00	0	3	045 157	42	STO	R0Z					C n
	00	0			09	9						D
	54	)			09	9		085 197				E
010 122	42	STO			01	1						REGISTERS
	09	9	3 × BIN		94	+/−						00
	09	9		050 162	44	SUM						01
	51	SBR			09	9						02
	01	1			09	9		090 202				03
015 127	04	4			58	*dsz						04
	08	8			01	1						05
	60	*if flg	"FLIP-	055 167	04	4						06
	00	0	FLOP"		08	8						07
	01	1	MEMORY		56	*rtn		095 207				08
020 132	04	4	TO		46	*LBL						09
	00	0	STORE		89	*3′	FLASH					10
	43	RCL	STATUS	060 172	85	+	ERROR					11
	09	9			85	+						12
	08	8			81	HLT		100 212				13
025 137	50	*st flg			46	*LBL						14
	00	0			19	*D′						15
	81	HLT		065 177	43	RCL						16
	01	1			09	9	BIN					17
	22	INV			08	8	#	105 217				18
030 142	50	*st flg			81	HLT						19
	00	0			46	*LBL						FLAGS
	42	STO		070 182	14	D						0
	09	9			43	RCL						1
	08	8			00	0	LAST	110 222				2
035 147	81	HLT			00	0	x					3
	36	*IND	← SBR		81	HLT		TEXAS INSTRUMENTS INCORPORATED				4
	43	RCL	LOOP	075 187		*LBL						

Figure 1: Standard Deviation Program with 6 Bins. This program allows one to accumulate the statistics for six different sets of data. The "bin control" routine selects which of the six variables is to receive new data. This routine is used to exchange sets of data.

initialization procedure if used in this manner.

Standard Deviation with 6 Bins

This program may be used conventionally without the memory management technique to calculate averages, standard deviations and normalized standard deviations. With the memory management shown, up to six different bins of data may be contained. For example, six different clerks could be compared as to their average orders filled as well as the consistency of their performance. Using the last program card you can store the results and update as often as desired. Likewise you can delete data in an orderly manner to maintain a four week running average and so on.

The user defined keys are used as shown on the program sheet. The bin exchange key— —

*E', performs the bin exchange function in a "flip flop" manner. The bin number is entered from the keyboard, say bin #4. The first time *E' is pressed, registers 01, 02, 03 exchange contents with registers 10, 11, 12 respectively. This places the data of bin 4 in position to be updated by the stored program. The next time you press *E', regardless of what you think you told it, the memory contents are automatically returned to their proper order with a 1 displayed to point out that bin 1 is in the update position. Any time 0 or a number larger than 6 is used as a bin number the display flashes an error.

Each bin consists of three registers which store (1) the sum of all the data entries, (2) the sum of the squares of all the data entries and (3) the number of datum points. The equation used is illustrated on the coding sheet.

Memory to Magnetic Card Program

A program can be written to magnetically record up to 22 memory registers, but this

Figure 2: Memory Management Program. When it is desired to save the data prepared by a program such as the standard deviation program of figure 1, this memory management program is read into the "A" side of the calculator. Its purpose is to copy data from the user data registers (M) to program registers (CARD), and vice versa under control of two keyboard commands. To save data, copy the user registers into the program registers (M → CARD) and write the program on both sides of card; to recover this data, read both "A" and "B" sides of the card, then perform the (CARD → M) transfer. Now load both sides of the standard deviation program to inspect these data.

SR-52 User Instructions

TITLE MEMORY MANAGEMENT (6×3) PAGE 1 OF 1

◄A► ◄B►

M→CARD CARD→M

STEP	PROCEDURE	ENTER	PRESS	DISPLAY
	TAILORED FOR STANDARD DEVIATION PROGRAM WHICH USES 6 BINS OF 3 REGISTERS EACH. WHEN BINS ARE IN PROPER ORDER, R98=1.			
	AS A SAFETY SIGNAL A BLINKING DISPLAY MUST PRECEDE PRESSING "A". +, =, A IS SUGGESTED. THIS TRANSFERS THE CONTENTS OF REGISTERS 01 THRU 18 TO REGISTERS 80 TO 97 IN ORDER. REGISTERS 80 TO 97 ARE RECORDED ON THE MAGNETIC CARD ALONG WITH THE PROGRAM. IF R98 ≠ 1 NO TRANSFER WILL OCCUR WHEN A IS PRESSED			
	AFTER READING IN BOTH SIDES OF THE CARD, PRESSING "E" TRANSFERS R80 THRU R97 IN ORDER BACK TO R01 THRU R18 WITH A "1" IN THE DISPLAY TO SIGNAL THE REGISTERS ARE AT THEIR HOME POSITIONS.			

SR-52 Coding Form

TITLE MEMORY MGMT (6×3) PAGE 1 OF 1
PROGRAMMER CLIF PENN DATE 5/19/76

LOC	CODE	KEY	COMMENTS	LOC	CODE	KEY	COMMENTS	LOC	CODE	KEY	COMMENTS	LABELS
000 112	46	*LBL	ALL FLGS		18	*C'						A M→CARD
	87	*1'	RESET		42	STO	18					B
	81	HLT	END	040 152	09	9	IN					C
	46	*LBL			09	9	STO PTR					D
	17	*B'	SBR		09	9		080 192				E CARD→M
005 117	36	*IND	FOR		07	7				DATA		A
	43	RCL	BOTH		42	STO	97			FROM		B
	09	9	CARD	045 157	09	9	IN			REGISTER 01		C
	08	8	TO		08	8	RCL PTR					D
	36	*IND	MEM		17	*B'		085 197		BIN		E
010 122	42	STO	AND		46	*LBL						REGISTERS
	09	9	MEM		11	A	MEM					00 CONTROL
	09	9	TO	050 162	22	INV	TO			#		01
	01	1	CARD		70	*iferr	CARD					02
	94	+/-			87	*1'	MUST	090 202		1		03
015 127	44	SUM	RCL		24	CE	START					04
	09	9	POINTER		43	RCL	WITH			DATA FROM		05
	08	8		055 167	09	9	BLINKING			REGISTER		06
	44	SUM	STO		08	8	DISPLAY			02		07
	09	9	POINTER		75	-	4 1	095 207				08
020 132	09	9			01	1	IN R98					09
	58	*dsz			95	=				DATA		10
	17	*B'		060 172	22	INV				FROM		11
	01	1	MATCHES		90	*ifzro				REGISTER		12
	42	STO	STD		87	*1'		100 212		03		13
025 137	09	9	DEV		18	*C'	18					14
	08	8	BIN #		42	STO	IN					15
	86	*rset		065 177	09	9	RCL					16
	46	*LBL			08	8	POINTER					17
	18	*C'			09	9		105 217		BIN		18
030 142	01	1	NUMBER		07	7	97			# 2		19
	08	8	OF		42	STO	IN			AND		FLAGS
	42	STO	MEM	070 182	09	9	STO			SO ON		0
	00	0	REGISTERS		09	9	POINTER			:		1
	00	0	CONTROL		17	*B'		110 222		:		2
035 147	56	*rtn	REGISTER							TO 223		3
	46	*LBL	CARD									4
	15	E	TO MEM	075 187								

DATA ALL

TEXAS INSTRUMENTS INCORPORATED

one records only 18 to match the previous program. Register 98 is used in a compatible manner with the previous program to protect against off loading the updated data in the wrong order, even if the flags have been reset. Several safety precautions are used to protect against losing a long accumulation of data. When the card is dumped into the calculator memory, the card still retains the old data in case of blunders. After the data are updated, if the registers are in their home position, register 98 will contain 1. In the memory to card program this branch condition is used. Up until this point you have lost no data that could not be restored. If you start to transfer memory to card and find the registers are incorrect, don't panic! Simply reload the standard deviation card, press *E' which replaces the registers where they belong.

To alert you that caution should be used, a blinking error condition display must exist to start the alternate transfer of memory to card with key A. This is readily accomplished in several ways but + = is the one I use. The only keys used are

E — — transfer program storage registers (magnetic card) to data memory

+ = A — — transfer data memory to program storage registers for card writing.■