52-NOTES

Volume 2 Number 9 48/39 September 1977

Newsletter of the SR-52 Users Club
published at
9459 Taylorsville Road
Dayton, OH 45424

HIR OPERATIONS: A MAJOR DISCOVERY(58/59)

Heinrich Schnepf (376), who publishes the German PPC newsletter DISPLAY, discovered that p82 can be used to access the 8 pending arithmetic registers. TI acknowledges awareness of this capability, but has no comment when asked why it was not announced. A PC-100A listing gives p82 the mnemonic: HIR, which I interpret to stand for Hierarchy Internal Register. I will henceforth use HIR mn to refer to an operation, and HIR n to refer to HIR number n. Heinrich found that the sequence: HIR mn does the following: With n taking on the address of one of the 8 HIRs (n=1,2,..8), m=0 produces STO, m=1: RCL, m=3: SUM, m=4: Prd, m=5: INV SUM, and m=6,7,8 or 9: INV Prd. Nested arithmetic operations push operands first into HIR 1, then on into 2,3,...8; print buffering assigns HIR 5 to Op 1, HIR 6 to Op 2, HIR 7 to Op 3, and HIR 8 to Op 4; INV P/R uses the first 2 available HIRs, and HIRs 7 and 8; P/R uses the first available, and 7 and 8; D.MS and INV D.MS use the first 2 available, and 8 ... which hold potentially useful separations of some of the D.MS'd elements; sigma + and - use 7 and 8; $\bar{x}$ (x-bar) uses the first available; Op 11 uses the first 2 available; Ops 12 and 15 use the first 3 available; Op 13 uses the first 4 available; and Op 14 uses the first 3 available and 8.

Unlike the analogous SR-52 Reg 60-69, the HIRs do not reformat pushed operands, nor appear to attach operators to them. Thus operands used in nested arithmetic operations stay "clean" and may be retrieved intact with the appropriate HIR recall (HIR 1n); stacked operands may be changed (by HIR 0n) or modified (by HIR mn, m=3,4,5,6) prior to operator execution. There is no (or doesn't seem to be a) fractured digits potential. Neither CLR nor CMs nor CP clears a HIR. The only way I've found that does (short of storing zeros or turning the machine off) is by Op 00, which only clears HIRs 5 through 8, and even then only if the 58/59 is plugged into the printer; without the printer, Ops 0-5 execute as Int.

HIRs 5-8 do reformat print code, and do not perform normal register arithmetic on print code contents produced by Ops 1-4. However, print code can be synthesized and put into HIRs 5-8 in such a way as to make possible normal register arithmetic. This leads to the ability to modify print code directly in a print buffer, obviating the need to use ordinary registers for such modifications. The printer appears to act upon the ten LSDs of the 13-digit mantissa of a print-buffer HIR: a situation similar to SR-52 fractured digits synthesis (V1N2p5).

The SR-52 Users Club is a non-profit loosely organised group of TI PPC owners/users who wish to get more out of their machines by exchanging ideas. Activity centers on a monthly newsletter, 52-NOTES edited and published by Richard C Vanderburgh in Dayton, Ohio. The SR-52 Users Club is neither sponsored nor officially sanctioned by Texas Instruments, Inc. Membership is open to any interested person: $6.00 includes six future issues of 52-NOTES; back issues start June 1976 @ $1.00 each.

When synthesizing print code, it is necessary to fill the 3 MSDs with non-zero numerals, then follow these with the desired ten digits of print code. For example, to artificially get the printer to print ABCDE in the leftmost print sector, key 99913141 EE 5, store with HIR 05, key 51617, SUM with HIR 35. Now key Op 5, and see ABCDE printed. At this point, if you want to change the C to say an *, key 3.6 EE 5, and sum with HIR 35. Key Op 5 and see AB*DE. If you try to start off with 1314151617 Op 1, HIR 35 register summing won't work; anyone know why?

Register arithmetic in the HIRs only works for integer, full integer-fraction, or floating point operands. The implied decapower sign of a fixed point fraction gets changed from minus to plus (Position B changes from 4 or 6 to zero or 2 (see V1N1p5 and V1N4p5)).

User access to the HIRs will undoubtedly find many practical as well as esoteric applications. In an arithmetic teaching application, Heinrich has written a program that zeros HIR 1 prior to halting for the student's answer. Following the input answer, HIR 1 is tested for zero, as a cheating indicator. In a biorhythm program, he uses HIRs 2-8 for storage of intermediate results, freeing data registers for print code. This program also demonstrates the plotting of multiple points, and makes use of CROM program ML-20 to perform calendar calculations. Fine tuning the partitioning to an effective 463.61 saves otherwise wasted space, but requires repartitioning during program/data entry: 479.59 to key in the program, and 399.69 to prestore the last two print constants. The program that follows contains an English translation of Heinrich's German messages; run it by pressing A and following printed instructions. It appears that clever accessing of the ML-20 program via the SBR nnn approach (V2N7p1) and making appropriate register reassignments could save quite a few precall data formatting steps, and perhaps some of you will want to try this. With the listing of this program, I am trying out some new symbol conventions: The * symbol for 2nd is dropped, L=Lbl, S=STO, R=RCL, *=Ind, rtn=INVSBR, and groups of numerals requiring synthesis will be bracketted by the " symbol. For example, DSZ 56 182 is written: Dsz "56" 1 "82" indicating that the 56 and 82 must be artificially merged; p82 is written HIR which, ofcourse, must be synthesized. At least one member has confused xEt with x=t, so I am substituting xXt for xEt. Leading zeros are omitted when address codes are followed by non-numbers.

Robert Snow (212) was probably within a few weeks of making Heinrich's discovery independently, having reported same to me today (1 Sept 77). Also today, a question from A B Winston (707) pointed to a use of the HIRs to provide a means for a running program to detect whether the 58/59 is plugged into the printer: Op 1 puts printer-formatted display into HIR 5 only if the 58/59 is plugged into the printer. The sequence: ... 0 HIR 05 7 Op 1 HIR 15 CP x=t 1'... transfers to L1' if the printer is not connected, continues if it is. This would allow programs to be I/O-optimized for operation both with and without the printer... a feature that would especially enhance many CROM programs.

TI-59/PC-100A Program: Biorhythm Analysis Heinrich Schnepf (376)/Ed

Program Listing:

```
000: LE S7 rtn LE' 4 LD' S6 Op 0 R*7 EE INV EE Op*6 Op 27 Dsz6 014 Op5
029: CLR rtn LA' 1/x X R4 = INV Int X 360 = sin X 7 = fix 0 EE INV fix
054: INV EE - 7 = div 5 = +/- rtn LB' S7 INV Int EE 1 INV log = SUM*7
077: CLR rtn LC' div 10 S42 S0 - INV Int Prd 0 + 1.2 SUM 0 = Int EE 2
103: + R0 INV xGEt 112 + 2 = INV EE rtn LB 27 E E' E' INV stflg7 Adv
127: rtn LC 35 E 1 D' R/S Prt EE +/- 4 HIR 2 2 D' R/S Prt EE 2 HIR "32"
150: 1 D' R/S Prt + HIR "12" = S0 Adv rtn LA 7 Op 17 Adv Adv 50 E E'
173: Adv E'E' Adv 6 Op17 43 E 3 D' 2 D' C Pgm20 A Op19 INV ifflg7 202
198: B GTO 180 2 D' R0 Pgm20 D S42 xXt 4 INV x=t 222 8 E GTO 230 10
224: SUM42 R*42 S42 2 D' Adv 43 E 3 D' 2 SUM07 2 D' C Pgm20 B Pgm20 C
250: Op19 ifflg7 260 CP xGEt 264 B GTO 233 S04 78 SUM5 Adv 17 E 2 D'
276: 2 D' 2 D' Adv Adv E' R5 - (365.25 X (1/x X (R5 - 122.1 )) Int)
310: Int = S3 div R14 = Int HIR 4 X R14 = Int INV SUM03 1 xXt HIR14"-
334: 13 = xGEt 344 + 12 = HIR4 10 xXt R3 C' EE 4 INV EE + HIR"14" C'
360: + 2 EE 5 = S3 CLR S0 S1 S02 23 A' HIR8 xXt 33 A' HIR6 28 A' HIR7
390: x=t 404 HIR"16" x=t 412 47 X HIR"18" B' HIR"17" xXt HIR"16" x=t
410: 418 50 X HIR"17" B' 51 X HIR"16" B' .24 xXt R1 EE div 6 INV log =
436: INV Int xGEt 447 24 EE 4 SUM1 CLR E E' Op25 Op24 Dsz"42" 3"61"
458: GTO 283
```

Prestored Data:

```
08: 3616134500 4317163117 3613374135 364131 303231 37411736 30.6001
15: 3723413536 213524 3624151327 4764332345 3713270000 5064301731
21: 3524376527 5164363324 3332364000 4000010000 311722 1613371700
27: 3033413717 6537001532 1632173631 1613371700 2213243140 24370013
33: 1700373545 3327171336 4517133520 7100000000 3032313723 1613457100
39: 13000000 2437002436 1613454000 364131 24310000 1700261745
45: 3327171336 1613371720 1424353723 16133717 3637133537 4536243651
51: 13311327 2345372330 5114243235 3723351717 3600243100 16133717
57: 1731371735 35633640 43243723 4017131523 3313353736
```

- - - - - - - - - -

PRINTER HEAD CLEANING (PC-100/PC-100A)

Since TI reports that running one of its printer head cleaning programs with heavy paper cleans the head by heating the dot-matrix elements, it would appear that programs energizing the most dots would be best. The TI-58/59 program (page VI-12 of the owner's manual) prints repeated lines of 20 8s, which exercises only 340 of the 700 dots. A routine along the lines of: LB Op1 Op2 Op3 Op4 Op5 rtn LA 3232323232 B 7676767676 B 2424242424 B GTO A cycles through the symbols 0 (code 32), capital pi (code 76), and I (code 24) which together exercise all 700 dots, when run by pressing A, and stopped with R/S. With regular printing paper installed, this program may be used to check out proper functioning of all the print dots.

The comparable 52 and 56 programs can also be substantially improved. Here the apparent objective is to exercise all the dots that can possibly be energized by these machines. At any position a numeral can be created (2-12, 14-16, 18-19), all 35 dots should be exercised; one should try to find the minimum set of trace mnemonics that covers all position 20 possibilities. I'll publish the best SR-52 and SR-56 printer head cleaning routines you send in.

MAGNETIC CARDS (59)

Exploring Mag Card Protection: Lou Cargile (625) has found that he can get a protected card to read without setting the protection flag by deliberately causing it to misread. The types of misreads that work may be machine-dependent; others trying this are invited to share results. In the meantime, a few questions from Jared Weinberger (221) bring up an important aspect of card protection not clearly covered in the owner's manual: reading any protected side of a card sets the protection flag which affects the whole machine until cleared with a manual CP or by switching the power off. For example, if the machine is partitioned "959." (no data registers) and Bank 1 recorded as protected on one side of a card, when that side is read, all of memory is protected. Repartitioning and/or p31 instructions encountered during program execution of protected code are ignored. However, any bank that is not fully partitioned for program memory can be copied normally on a separate card even though the protect flag has been set. The protect flag can then be cleared, the card read, and the code examined. For example, cycle your machine off-on, then key 10 Op17 LRN LA 123 R/S LRN 999 S99 888 S98. Now key 1 +/- Write and feed a card through. Cycle off-on, key 10 Op17 CLR, and read the card just written; see -1 displayed. Press A, see 123; RCL99, see 999, R98, see 888. But press LRN, and the machine stays in RUN mode, and any attempts to repartition fail. Now key 1 Write and feed a blank cardside through; see 1 displayed. Cycle off-on, then key 10 Op17 CLR and read the card; see 1 displayed. Press A, and the RCLs, and find things the same as for the protected card. But now you can get into LRN mode and/or repartition. The first cardside was sort of partially protected: once read, the program part could not be examined in LRN mode, but it could be copied to a second cardside normally, which could then be read and examined normally. This partial protection situation also applies to all of memory, not just to the one bank that has been read from a protected card. For example, a 319.79 partition says that all of Bank 1 is program, Bank 2 is a mix of program and data, and Banks 3 and 4 are all data. A cardside protect-recorded with this partitioning from any of the 4 banks, when subsequently read denies only Bank 1 from being recorded. A consequence of all this is that the read-in of one protected cardside, by virtue of its having set the protection flag and frozen the partitioning, predetermines the rules by which all of memory will behave, regardless of whether subsequent card-reads are by protected or unprotected sides.

Incidently, although resident code affected by a protected cardside read cannot be SST'd, you can access it selectively by SBR nnn. This may make it possible in some cases to synthesize protected code by observing results when SBR nnn is executed repeatedly for all nnn from the end of the program partition back up through step 000. Then, if you're lucky, you'll intercept a sequence like: ...GTO 182 A rtn, and a SBR call to the step containing the 82 will return with the contents of HIR 1 in the display!

Mis-reads-writes: Joel Rice (3) reports considerable difficulty getting cards to read properly. We may find that because of the greater information density, 59 card read/write won't be as reliable as the 52's. Others having serious read/write problems are invited to share them. I've found that clearing program and data registers before a read, or a write key-in appears to help, as does wiping each card with a clean lint-free cloth, which should also be used if after read or write you can't remove the card without getting a fingerprint on it.

Mag Card Read and Write Under Program Control: The owner's manual (VII-5) notes that mag cards can be read under program control, but doesn't say what happens when executing code is overwritten by a read, or that card write can also be effected under program control: a feature brought to my attention by Bob Moore (488).

It turns out that if a cardside destined for Bank m is read by an INV Write instruction being executed in Bank n, $n \neq m$, execution resumes at the step following the INV Write after the card has been read. But if m=n, and this bank number precedes the INV Write, execution of the old code continues past the m INV Write until that part of the read has been completed. Then execution continues on into the new code at the step following where the old code last executed. This amounts to as much as a second or so of parallel processing: card read and code execution occurring simultaneously. Duration depends upon where the INV Write is located within its bank: the closer to the end of the bank, the longer the parallel processing.

SR-52 Cards for the TI-59: Mack Maloney (246) has found that SR-52 mag cards are the same thickness as the 59's (.008") and that if they are trimmed to the 59's card width (.635") they appear to read and write properly. The .025" extra length doesn't seem to matter. As more users try this, we should be able to determine whether there is any problem with TI-59 re-writes over SR-52 code, which may not erase all the old code (a reported problem for HP-67 users when attempting to use old HP-65 cards). To be safe, you can first demagnetize the SR-52 cards.

A PC-100A TYPEWRITER (58/59)

Thomas Cox (9) suggests that "It would be interesting if someone could devise a useable routine for the TI-58/59 that would use the Rausch keyboard to generate the corresponding characters on the PC-100A." The following program does what Thomas has asked for, running somewhat like a slow typewriter, providing analogues of the horizontal position indicator, carriage return, end-of-line bell warning, and replaceable characters of real typewriters. This is the sort of program that is worth refining to cut execution time, since there are broad practical applications. So send me your better versions... perhaps TI could be persuaded to put the best one in a new CROM.

Incidently, the rather surprising coincidence (or did TI plan it that way?) that the last column number (20) is also the print code for the hyphen symbol saves a couple of keystrokes. For the 58, partition 159.39; for the 59 turn-on will do.

TI-58/59 Program: PC-100A Typewriter Ed

User Instructions:

1. Initialize: Press A, see 1 (ready for first letter)
2. Key ith letter via Rausch overlay (V1N3p2), see i=1; display flashes when i+1=20, in which case either Rausch-key last letter to fill line, or press E to end line with a hyphen. Repeat step 2 until message is complete. To end an incomplete line with blanks, follow last letter with R/S. To generate a non-letter character, key the 2-digit print code and press E (instead of Rausch keying).

TI-58/59 Program: PC-100A Typewriter Ed

Program Listing:

```
000:  LA Op00 20 xXt 4 S28 1 S31 S33 L2' 5 S00 1 EE 8 S30 CLR S32 L1'
030:  R33 INV x=t 3' Op55 L3' R/S R32 Op*31 GTO 4' LD + 9 LC + 9 = LB
057:  S29 CE R*29 LE X R30 = SUM 32 100 INV prd 30 1 SUM 33 Dsz0 1' R32
084:  Op*31 1 SUM 31 Dsz"28" 2' L4' Op5 GTO A
```

Prestored Data:

```
01:  36 42 45 25 30 33 13 16 22 37 43 46 26 31 34 14 17 23 41 44 0 27
23:  32 35 15 21 24
```

- - - - - - -

MEMBERSHIP ADDRESS CHANGES

3: 230 W 107th St #6J New York, NY 10025; 48: 3611 Wyatt Dr Holiday, FL 33590; 75: Box 699 College Station, TX 77840; 281: 1502 42nd St #3 Brooklyn, NY 11219; 306: 4523½ Avocado St Los Angeles, CA 90027; 343: 1325 Quaker St Golden, CO 80401; 346: 37 Winding Way Madison, NJ 07940; 347: VQ-3 Box 65 FPO San Francisco 96637; 440: 148 Transmitter Rd Oak Harbor, WA 98277; 449: 807 Ruggles Hall Columbia Univ New York, NY 10027; 471: Box 461 Storrs, CT 06268; 584: 266 Main St #325 Windsor Locks, CT 06096; 587: 501 Loring Ave Los Angeles, CA 90024; 594: 45A Hillview Ave Rensselaer, NY 12144.

MISCELLANY

How the PPCs Work: Gene Werner (120) has kept a record of some 20 or so books and magazine articles published over the past few years covering various aspects of calculator and PPC design. Send Gene a SASE and a few stamps for his bibliography.

TI Trig Algorithms: Joel Pitcairn (514) reports having mechanized the CORDIC technique (see BYTE Aug 77 p 142) for calculating the tangent function and comparing results with the SR-52 built-in tan function. For base ten and Max=13, Joel finds such close agreement, especially for critical values, as to conclude that this is the approach used by TI in the firmware. Execution time comparisons suggest that the sin and cos functions are derived from the tangent; it also appears that TI also uses the CORDIC algorithm to get arctan. Joel notes an error in the BYTE article: arctan xo=zi+xi, not just zi. Joel's findings are likely to apply to the other TI PPCs as well.

Owner's Manual Versions (58/59): Mack Maloney (246) reports that there are currently 2 versions: 1014983-1 and -2 (look on the lower right corner of the back outside cover to see what yours is). The -2 reportedly corrects most of the -1 errors, and has its own addendum sheet (#1019286-4).

Numerical Solutions To Partial Differential Equations: Larry Gearhart (442) would like to get in touch with anyone pursuing numerical solutions to PDEs via one of the TI PPCs.

More on INV and Ind Viability (52): Claude Belanger (254) notes that while INV and Ind viability are maintained through direct subroutine calls (to A-E') (see V2N5p4), viability is lost if the INV or Ind is followed by GTO or SBR.

TI Notes: Check with your dealer or recent TI ads for details concerning the free Leisure Library CROM for all TI-58/59 owners.

PPX-59 is expected to get under way 1 Oct 77 with an initial catalog covering CROM programs and conversions of "the top 200" PPX-52 programs.