TI PPC NOTES

NEWSLETTER OF THE TI PROGRAMMABLE CALCULATOR CLUB

P.O. Box 1421, Largo, FL 34294

Volume 9, Number 1 January/February 1984

Welcome back for the fifth year of the TI Programmable Calculator Club, and for its newsletter TI PPC Notes. I apologize for the tardiness of the first issue. I had minor surgery in January, and have had a slow recovery. I expect that the second issue for 1984 will also be late.

The big news in this issue is the arrival of the TI-66. Dave Leising has made substantial progress in understanding that machine. No HIR, fast mode or high resolution graphics yet; but, we already have a more versatile absolute addressing capability than described in the manual. In the next issue we will examine more of Dave's findings on mnemonic codes. There are many more than indicated in the manual--a full 16x16 matrix of them it seems.

Bob Fruit offers an updated version of his index again this year. It now covers the full four years of TI PPC Notes. This issue also has more tutorial material on fast mode. One accomplishment last year was a much more widespread use of fast mode techniques. Programs include a decimal/octal conversion, a loan schedule generator, and an improved solution of linear equations.

The response to the programming puzzles in V8N6 was overwhelming. Was it something to do over the holidays? I have barely started sorting through all the submissions for the use of a single digit integer three times to yield a 24 in the display. There are more programming challenges in this issue. So, happy puzzling.

Magnetic Card Service

Magnetic cards will be sent for programs in this issue, or the 1983 issues, for a price of one dollar per card plus a stamped and self addressed envelope. I will not combine different programs on one card. For example, David Mah's program requires one card, Robert Prins' one, and Carl Rabe's, two cards.

ERRATA:

BRITISH FLAG - George Thomson and Don Graham note that Reginald van Genechten's flag program on V8N6P22/23 results in a printout that is similar to the British flag, not the English flag. They also noted that the diagonal crosses were not correctly formed. I honestly didn't understand why the slight deviation in form was so important until I read the following description of the Union Jack from the family's Grolier Encyclopedia:

> "British national flag. It combines the red cross on the white of St. George's banner for England, the diagonal white cross on blue of St. Andrew for Scotland, and the diagonal red cross on white of St. Patrick for Ireland. St. Patrick's red cross is superimposed on St. Andrew's white one in such a way that the wide white stripe signifying Scotland is uppermost in two quarters of the flag and the red stripe signifying Ireland--since 1920, only Northern Ireland--is uppermost in the other two. Thus the emblem of one country does not fly above that of the other."

I suspect that all of this will serve to convince both Reginald and I to leave the drawing of flags to the nationals of the respective countries.

SHELL SORT - George Thomson. In V8N6P21 the S in Shell sort should be capitalized.

HARDWARE AVAILABILITY - V8N6P3 reported four potential sources for TI-59's in the Washington D.C. area. In response to telephone calls in late January all four indicated that they no longer had TI-59's.

V8N6P13 reported that Don Lambert had TI-59's and PC-100's for sale. A telephone call in late January revealed that he had only one TI-59 remaining, and was asking the list price of $300.

I have one friend here in Florida who is willing to sell a TI-59/PC-100 combination for $150 plus packing and shipping.

PRINTER PAPER - A number of members have commented on the quality of printer paper, with problems of discoloration with age, poor contrast, etc. I have tried to contact the source of paper mentioned in V7N1/2P10 but I get a response indicating no recognition of Mr. Kolb. Recently I discovered that one of the local discount houses had a full case of printer paper packaged the old way, that, is with the three rolls in line to form a long cylinder, anf including a head cleaning card. Some of the rolls that I purchased shoW slight discoloration on the outside, but once two or three layers are stripped off the remainder provides good printouts. While they last I will purchase and ship a package of three rolls anywhere in the USA for twelve dollars--sorry, I cannot assume any responsibility for the quality of the printout from individual packages.

MORE ON COLLATZ'S ALGORITHM - D. H. Taysum of Murray, Utah wrote to remind me that in February 1983 he had sent the following clipping making the connection between Ulam's Conjecture (V6N9/10P13) and Collatz's Algorithm (V8N6P13):

> "Mathematicians have been stumped for 48 years. Ever since Lothar Collatz discovered the mathematical sequence that bears his name, they've been wondering. Are there any other cycles? Brigham Young University mathematicians have come up with an answer--maybe. But if there are other cycles, they're big cycles. If that doesn't make any sense, maybe this will: Pick a number, any number. If it's even, divide it in half. If it's odd, multiply it by three and add one. OK, you've got a new number. Do the same thing with that one-- ... ...Keep it up, it can be fun. Sooner or later, your're going to wind up with the number 4, then 2, then 1, then 4 again, the 2 again, etc., etc., etc. That's a Collatz sequence. ..." (The Salt Lake Tribune, March 10, 1981)

The remainder of the article discusses work at BYU which indicated that any other end cycle other than 4, 2, 1 must have more than 116,000 terms.

Brian Hayes' column "Computer Recreations" in the January 1984 issue of Scientific American discussed the Collatz sequence in great detail. The article suggests that the problem may actually have been independently invented several times. Hayes gives the class of such sequences the name "hailstone numbers" as the path of the series is like the trajectory of a hailstone through a storm cloud.

After reading the Scientific American article I decided to examine whether there might be a faster algorithm for the TI-59 than that used by Professor Widmer in V6N9/10P13. One that I thought might be successful was a STflg Ind - Ifflg sequence (see V8N2P17) since that would not require manipulation of the entire value, but only tests of the ones digit:

```
000  69 DP     011  00  00     022  08  08     033  01  01     044  03  03
001  23  23    012  32  32     023  00  00     034  55  ÷      045  91  R/S
002  86 STF    013  87 IFF     024  32  32     035  02  2      046  76 LBL
003  40 IND    014  04  04     025  44 SUM     036  95  =      047  11  A
004  01  01    015  00  00     026  01  01     037  67 EQ     048  47 CMS
005  87 IFF    016  32  32     027  44 SUM     038  00  00     049  32 X:T
006  00  00    017  87 IFF     028  01  01     039  43  43     050  01  1
007  00  00    018  06  06     029  69 DP     040  42 STO     051  32 X:T
008  32  32    019  00  00     030  21  21     041  01  01     052  42 STO
009  87 IFF    020  32  32     031  81 RST     042  81 RST     053  01  01
010  02  02    021  87 IFF     032  43 RCL     043  43 RCL     054  81 RST
```

That illustrates the concept. The execution times are actually about 25 per cent slower than the Widmer routine. Furthermore, the program cannot be used to obtain a reasonable printout of the sequence since the printer will go into the TRACE mode the first time the ones digit is a nine and Flag 9 is set (V8N1P24). Any other ideas out there?

NO A.O.S. WITH THE BA-55 - In V8N5P9 I gave a brief description of the BA-55 "Professional Business Analyst" financial calculator. This programmable has up to 40 program steps or up to 5 data memories depending on the partitioning--a limited capablity. But, both the TI-66 and the BA-55 are useable with the PC-200, and I had hoped to get a head start on use of the PC-200 while waiting for the TI-66 to become available. Unfortunately, PC-200's did not become available.

While using my BA-55 recently to perform some routine calculations I made an amazing discovery. The BA-55 does not have the standard algebraic hierarchy we have come to associate with TI calculators (A.O.S.). As explained on pages 1-6 and 1-7 of the manual for the BA-55, the TI Professional Business Analyst Guide, pressing an arithmetic key completes the previous operation, brings the result to the display, and defines the next operation to be performed. For example, if the user presses the Add (+) key the result of previous calculations will be displayed, and the next number entered will be added to the displayed value. Thus, with the BA-55:

2 x 3 + 5 = 11 but 5 + 2 x 3 = 21

The A.O.S. system employed with all the other TI programmables, including the TI-MBA, will yield the same answer (11) for either sequence. Does anyone have any idea why TI would abandon A.O.S. for the BA-55?

SOLID STATE SOFTWARE MODULE AND MAGNETIC CARD AVAILABILITY

Mr. J. M.Gallego reports that he purchased the following items from a small company which had gone out of business:

68	40 Blank Magnetic Cards with card case
15	Business Decisions modules
9	Securities Analysis modules
10	Real Estate & Investment modules
10	Aviation modules
10	Navigation modules
13	Surveying modules
2	Agriculture modules

These are new items. He will sell them for sixteen dollars ($16.00) for each module, and eight dollars ($8.00) for each box of magnetic cards while they last. Shipping is included in those prices. U. S. members should send money orders only to:

Q. Jose M. Gallego
250 Quintard Avenue, Apt. 96
Chula Vista CA 92011/4924

Members from other countries should write to make appropriate arrangements.

THE TI-66 ARRIVES - Palmer Hanson. V8N3P12 reported that TI had announced that we would have a new programmable calculator. V8N5P18 and V8N6P20 carried limited coverage of the new calculator - it had not yet become available at retailers. In mid-December the TI-66 arrived at the local discount department stores. Typical prices are in the fifty dollar range versus the suggested retail price of seventy dollars. The companion printer, the PC-200, continues to be unavailable.

Preliminary work with the TI-66 shows that it is essentially equivalent to a TI-58C, but without the Solid State Software capability. The clear advantages are:

* The liquid crystal display.

* The reduction in power required with the LCD display. The TI-66 user is freed from all those concerns with battery charging which accompanied the LED display type calculators.

* 512 program steps versus the normal 480 with the TI-58C; although 512 program steps are available with the TI-58C using the techniques developed by Patrick Acosta (see V6N1P15).

* Smaller size - you can really carry the TI-66 in your pocket.

* The user does not have to remove the battery to get access to the printer.

* A battery-operated printer - when it becomes available.

* Displayed mnemonics for the operation codes in LRN mode.

* A command (Part) which allows the user to partition for any number of data registers from 0 to 63.

* Revised print code which may provide easier print code conversion routines. That cannot be determined for sure until we get a printer and find out if there is an unannounced extended print code table.

With all those gains relative to the TI-58C, are there any losses? Yes there are:

* The lack of a solid state software module capability.

* The execution speed is slow, even when compared with normal mode of the TI-58C/59. For the test case of 1 + 1 + 1 + ... + 1 + RST which was used in V8N2P19, the TI-66 will accumulate a sum of about 346 in a minute. For the same problem the TI-58C would count to about 668, and the TI-59 would count to over 800. Run times for other test problems are typically about twice that which would be required with a TI-58C. The BA-55, the other TI programmable which will operate with the PC-200 printer, provides about the same speed as the TI-66; it will count to 360 in a minute. That is very close to the speed of the older TI-MBA.

The TI-66 Arrives - (cont)

* The TI-66 user will really notice the slower execution speed when in LRN mode. It is very easy to enter commands faster than the calculator can accept them, with the result that you get gibberish programs. I never had that problem with the TI-59.

* In LRN mode the instruction you key in is always inserted after the instruction in the display. There is no write-over capability. I find this inconvenient. My sentiments are exactly the same as those expressed by Robert Bowden when reviewing a word processing program in an article "Computer Boredom" in the January 14, 1984 issue of the St. Petersburg Times:

> "...BankStreet Writer has another flaw. It's constantly in an 'insert' mode. This means a user can not simply back up and strike over a typed error. ... The mistyped letter must first be deleted, then the corrected letters must be inserted. It seems to me most writers strike over far more than they insert. Thus an insert word processing program is a backward procedure."

  That feature isn't limited to the "Bank Street Writer". The text processor on my Model 100 responds the same way. I find that inconvenient too.

* The PC-200 will only have a 16 character line. This means that we will not be able to write those nice programs which will print out thirteen digits, the exponent, and the memory location.

And what of hidden features? As reported in V8N5P18, the TI-66 manual states (see page F-3):

> "There are no HIR commands or other hidden features on the TI-66 that you may have accessed on the TI-58/58C/59 through illegal key sequences."

Dave Leising has been searching diligently for unlisted commands and the like. His results are reported later in this issue. To date he has been able to identify many additional mnemonics which can be used as labels, but has not found HIR commands, fast mode entry techniques, and the like. I have verified that some old quirks no longer work and have identified some new quirks:

* There is no indirect Stflg/Ifflg capability to operate on the least significant digit of a value in the indirect register as reported in the May/June 1982 issue of PPX Exchange. There is a limited Fix Ind quirk of the type described in V8N2P17. The Fix Ind XX sequence will set the fix mode based on the ones digit of the value in XX if the value not greater than 99. If the value in XX is negative, Fix 0 is set.

* There is no problem with INV $\Sigma+$ in the EE mode as with the TI-58,58C/59 (see V8N1P7).

The TI-66 Arrives - (cont)

* Dsz works only on zero through 9, whether the register is defined in the code or is from an indirect call.

* There is a potential step-saver for absolute addresses between 00 and 99. Suppose you want to use a GTO to jump to location 023. With the TI-58/58C/59, and according to the instructions for the TI-66, a three step sequence will be required: GTO-00-23. With the TI-58/58C/59 if you delete the 00 the sequence reverts to a GTO-Ln sequence because the 23 is read as a label. Not so with the TI-66, where each command has its own mnemonic, and the 23 continues to be read as an address. But there are other rules to follow when you use the truncated absolute address. Dave Leising describes the various options in an article on page 8.

THE PC-200 PRINTER - Dave Leising has obtained an engineering model PC-200. The printer is expected to become available at dealers late in the first quarter of this year. The print is much smaller than with the PC-100. Sample printouts appear at the right.

Dave has determined that there is an extended print code table for the print codes of the TI-66,, just as there was with the TI-58/59. The extended table for the TI-66 appears below, where the extended code appears in the columns 8 and 9, and in row 9.

```
THIS IS A SAMPLE
PC-200 PRINTOUT
SHOWING THE FONT
OF THE TI-66

 A B C D E F G
 H I J K L M N O
P Q R S T U V W
 X Y Z n √ x ÷ =
0 1 2 3 4 5 6 7
 8 9 - + ' . ?
Σ ( ) ≠ / ! e π
 x̄ < > ≥ % ² Δ ,
↑ x̄ ≤ II " * ° ∴
```

```
          0.  00
          0.  01
24101123.     02
  112300.     03
 1002301.     04
15201405.     05
20035242.     06
40400020.     07
22111624.     08
17252400.     09
23101727.     10
11160700.     11
24100500.     12
 6171624.     13
17060024.     14
10050024.     15
11524646.     16
          0.  17
          0.  18
          0.  19
          0.  20
```

```
      ST
000   1
001   4
002   STO
003   00
004   2
005   STO
006   01
007   RCL*
008   01
009   DP
010   01
011   DP
012   21
013   RCL*
014   01
015   DP
016   02
017   DP
018   21
019   RCL*
020   01
021   DP
022   03
023   DP
024   21
025   RCL*
026   01
027   DP
028   04
029   DP
030   21
031   DP
032   05
033   DSZ
034   00
035   00
036   07
037   ADV
038   ADV
039   ADV
040   ADV
041   CLR
042   R/S
043   RST
```

TENS DIGIT	UNITS DIGIT 0	1	2	3	4	5	6	7	8	9
0	blank	A	B	C	D	E	F	G	blank	A
1	H	I	J	K	L	M	N	O	H	I
2	P	Q	R	S	T	U	V	W	P	Q
3	X	Y	Z	n	√	x	÷	=	X	Y
4	0	1	2	3	4	5	6	7	0	1
5	8	9	-	+	'	.	?	blank	8	9
6	Σ	(	)	≠	/	!	e	π	Σ	(
7	'	<	>	≥	%	²	Δ	,	'	<
8	↑	x̄	≤	II	"	*	°	∴	↑	x̄
9	H	I	J	K	L	M	N	O	H	I

ALTERNATE TI-66 ADDRESSING MODES - Dave Leising. When a GTO or a SBR is encountered while running a TI-66 program the operating system looks at the immediately following location to determine the transfer address. If a non-numeric code is found the system assumes that label addressing is to be used. If the location contains a code of numeric significance, either the numeric mnemonics 0 through 9, or the absolute address mnemonics 00 through 99, then absolute addressing is assumed and the system looks for a sufficient number of numerically significant mnemonics to form an absolute address. In normal operation the LRN mode will assemble the instruction GTO 123 into three locations as GTO 01 23. But, by an appropriate key-in sequence, say GTO A Del 1 2 3 , the user can assemble a four step GTO which will also transfer the program to location 123.

If an insufficient number of numerically significant mnemonics to generate a full three digit absolute address follow the GTO or SBR then some very interesting things begin to happen. In general, the program counter is first set to the address defined by the incomplete numeric value, the instruction immediately following the insufficient numeric mnemonics is executed, and then program control is transferred to the new location of the program counter. If the instruction following the insufficient numerics is a multipart instruction (STO, STF, GTO, etc.) the pending syntax is preserved and completed by the code at the destination address. This feature, which is not described in the manual for the TI-66, would seem to provide some powerful and versatile programming options. Rules for use and examples follow:

Rules:

1. GTO MN X , where MN is the address mnemonic 00 through 09 and X is an instruction code, will go to MN and execute X.

2. GTO MN X , where MN is the address mnemonic 10 through 99 and X is an instruction code, will go to MN but will not execute X.

3. GTO M X , where M is the numeric mnemonic 0 through 9 and X is an instruction code, will go to M and execute X.

4. GTO M N X , where M and N are numeric mnemonics 0 through 9 and X is an instruction code, will go to MN and execute X.

5. GTO M N P X , where M, N, and P are numeric mnemonics 0 through 9 and X is an instruction code, will go to MNP but not execute X.

6. In cases where GTO (address) X yields the execution of X, and X is a multipart instruction, the pending syntax will be preserved through the transfer and will be completed by the code at the transfer address.

7. In cases where GTO (address) yields the execution of X, and X is a user-defined keycode (A through E'), the subroutine X will be executed, but upon return control will be transferred to location (address) + 1 , not to the calling location + 1.

8. In cases where GTO (address) X yields the execution of X, sequences of the form

   (address) GTO (address) GTO

   will cause a machine crash if the code starts at (address).

Alternate TI-66 Addressing Modes - (cont)

9. In all cases where GTO (address) X yields execution of X, the execution of X takes place after the program counter is set to (address). It is as if instruction X is executed at the location (address) without it really being there or changing in any way the code located at (address).

10. GTO can be replaced by SBR in the above rules. In the case of the crash syntax (rule 8 above), the crash state will continue until six levels of subroutine are exceeded, then a halt upon error will occur.

Examples:

1. a. Write code starting at location 085: 1 5 R/S
   b. Write code starting at location 100: LBL B 4 5 GTO 8 5 STO (Note: You can get the sequence GTO 8 5 STO without any deletes due to the automatic insert function in LRN. You can simply press GTO STO Bst 8 5)
   c. Go out of LRN and press B. The calculator will stop with 45 in the display. Press cLR and RCL 15 and again see 45 in the sisplay, indicating that the sequence stored 45 in data register 15.

2. a. Write code starting at location 000: LBL A (sequence) RTN
   b. Write code starting at location 085: R/S R/S R/S
   c. Write code starting at location 100: LBL B GTO 8 5 A
   d. Go out of LRN and press B. The sequence of subroutine A will be executed, and the return will be to the R/S at location 86 (85 + 1 per rule 7 above). The calculator will stop at location 087 after executing the R/S at location 086.

3. a. Write code starting at location 000: 00 GTO 00 GTO , or
      0 GTO 0 GTO
   b. Press RST and then R/S. The calculator will crash (rule 8).

4. a. Be sure there is no R/S code at location 045.
   b. Write code starting at location 100: LBL C GTO 4 5 R/S
   c. Go out of LRN and press C. A halt will occur with the program counter at location 046, as if the R/S at location 105 had been at location 045.

PROGRAMMING CHALLENGES FOR THE TI-66 - With only 512 program locations, no magnetic card reader, and no Solid State Software (TM) modules program space will be at a premium on the TI-66. Therefore, it seems that specialized routines, optimized for minimum memory requirements, should be particularly useful. The challenge for TI-66 users is to begin accumulating a library of such routines. Three that I suggest are:

1. Print code converters which change integers to print code. A primary TI-59 example is the Robert Snow converter (V5N6P10).

2. An alpha code converter like those on page 18 of this issue.

3. Some sort of 13 digit register list routine.

ADVANCED ELECTRICAL ENGINEERING module for the the CC-40.

Review by Maurice E.T. Swinnen.

This is the fourth module for the CC-40 I have seen so far, and all prove to be of an extraordinary quality and usefulness. Although I feel a little at home with the Mathematics module, I certainly feel unqualified to review either the Statistics or the Finance module. But Electrical Engineering is a field I eat, drink, and sleep at least ten hours a day, and I have been doing this for the last fourty years. Boy do I wish I had this CC-40 and this EE-module when I started, eons ago! The closest I ever came to it was a slide rule or a Monroe mechanical calculating machine.

The module contains the following programs:

1. Active second-order multiple-feedback (one op-amp) low-pass, high-pass and band-pass filters.
2. Bode-Nyquist calculations.
3. Roots of a polynomial. (Finds all real and complex roots of up to a 20th degree polynomial in one variable with real coefficients)
4. Discrete Fourier transform. (Transforms a sampling of the time domain to the frequency domain and also performs the inverse transform from the frequency domain to the time domain. Six windowing techniques are available for sidelobe suppression.)
5. Passive low-pass filters. (Very handy in very-high frequency computations. Allows design of both Tchebycheff and Butterworth low-pass filters.)
6. Phase-lock loop calculations.(Complete! For both active and passive types)
7. Series/Parallel impedance conversions.(I am not so crazy about this one. Bill Beebe wrote a simpler and more useful one for the TI-59.)
8. Signal detection. (Calculates signal-to-noise ratio, probability of false alarm, probability of detection given any two of the three, and the ratio of the standard deviation of the two signals.)
9. S to and from Y, H, and Z parameter conversion. (This program has Gary Morella written all over it. Gary is no longer working at TI, although the manual of this module names him in the credits list. In my opinion this alone is worth the price of the module. I have seen several attempts to write a program of this magnitude for the TI-59, but they all had serious shortcomings, mostly due to the limited memory available. The only program that did things satisfactorily is contained in the EE-module for the TI-88 and it was written by, you guessed it, Gary Morella. Unfortunately TI made only twenty samples of the TI-88 EE-module, which makes them even rarer than hen's teeth.)

Besides these programs there are several subprograms. They are shared by the main programs, in about the same manner as subroutines are. But they may also be called from a user-written program in RAM. As an example of this technique, I have enclosed at the end a program that uses two subprograms: PR and RP. They do the conversion of Rectangular to Polar and vice-versa for you. They are, of course, built into the firmware of the TI-59, but not in the CC-40. The program is fully prompting, which makes mistakes almost a thing of the past. I admit, with some editing, one could write it on fewer lines, combining several statements on one line each time. But for the gain of a few bytes, readablity would suffer in the process. In this program a technique is used, unique to the TI-99/4A (the home computer) and the CC-40: one-key response. Most computers require you to place your answer in the display, followed by pressing the ENTER key. Here it is possible that simply pressing Y or N allows you to select program sequence. See, for example, line 120. It displays the message"Rectangular to Polar? Y/N", and assigns A$ to KEY$. It waits for your response. If you press the N-key, either in lower or upper case, line 130 sends you to line 320. If you press the Y-key (or any other key for that matter) the

program continues with line 140. It is very similar to the user-defined keys in the TI-59, except that all the keys can be used and that the user can select which ones and their effect.

To conclude, this module is well worth investing in, if your game is electrical engineering. If TI would just see fit to finally produce some peripherals for this portable machine I, and a lot of my friends, would be very happy to clear out the nine programs we all have stored in permanent memory. And we finally would be able to sleep tightly again, free of nightmares that someone might type the dreaded word NEW on the keyboard.(For those not familiar with Basic, we will let you in on the joke: NEW, followed by ENTER, wipes out everything in RAM, program and variables, and the mere mention of the word is enough to give me apoplexy.)

```
100  DISPLAY AT(2)"EE module in place? Y/N":A$=KEY$
110  IF A$="N" OR A$="n" THEN 360
120  DISPLAY AT(2)"Rectangular to Polar? Y/N":A$=KEY$
130  IF A$="N" OR A$="n" THEN 320
140  DISPLAY AT(2)"X-coordinate?";
150  ACCEPT AT(18)VALIDATE(NUMERIC)BEEP,X
160  DISPLAY AT(2)"Y-coordinate?";
170  ACCEPT AT(18)VALIDATE(NUMERIC)BEEP,Y
180  CALL RP(X,Y,M,A)
190  DISPLAY AT(2)"Magnitude=";M:PAUSE
200  DISPLAY AT(2)"Angle=";A;"degrees":PAUSE
210  GOTO 120
220  DISPLAY AT(2)"Polar to Rectangular? Y/N":A$=KEY$
230  IF A$="N" OR A$="n" THEN 350
240  DISPLAY AT(2)"Magnitude?";
250  ACCEPT AT(18)VALIDATE(NUMERIC)BEEP,M
260  DISPLAY AT(2)"Angle in degrees?";
270  ACCEPT AT(20)VALIDATE(NUMERIC)BEEP,A
280  CALL PR(M,A,X,Y)
290  DISPLAY AT(2)"X-coordinate=";X:PAUSE
300  DISPLAY AT(2)"Y-coordinate=";Y:PAUSE
310  GOTO 220
320  DISPLAY AT(5)"Exit program? Y/N":A$=KEY$
330  IF A$="N" OR A$="n" THEN 220 ELSE END
340  DISPLAY AT(5)"Exit program? Y/N":A$=KEY$
350  IF A$="N" OR A$="n" THEN 120 ELSE END
360  DISPLAY AT(4)"Insert EE module, please!":PAUSE 4
370  END
```

EDITOR'S NOTE - My sentiments about the lack of peripherals are the same as Maurice's. I am using the Mathematics module and have a set of interacting programs which perform polynomial regressions, compute residuals, solve sets of linear equations by various methods, and the like. One inadvertent NEW would be a disaster. The CC-40 is beginning to get some favorable press. In the article "Choosing a Notebook Computer" in the January 1984 issue of Creative Computing author David Ahl discusses price versus performance:

> "... But perhaps most interesting are the five machines that fall below the curve, and thus represent relative bargains. At the low end is the TI CC-40. For professionals, students, and engineers, this is an unbeatable machine at only $250, frequently discounted to well under $200. ... "

DECIMAL/OCTAL CONVERSIONS - David Mah of Vancouver, B.C.. This fast mode program for the TI-58C/59 should come in handy for computer science professionals and hobbiests. The program uses Patrick Acosta's h12 method of fast mode entry (V6N8P4) combined with Palmer Hanson's transparent fast mode (V7N1/2P23). It also demonstrates selection of the fast mode entry constant to control flag 4 to indicate whether decimal-to-octal or octal-to-decimal is desired (V6N8P3, lines 045 and 052 of the program).

Correct conversions are provided over the range from 7,777,777,777,777 (octal) to 549,755,813,887 (decimal). Illegal inputs such as negative numbers, or octal numbers containing an 8 or a 9 will produce a flashing 9's error indication. Decimal inputs outside the input range will not indicate an error. There is no provision to print out all digits.

Program Listing:

The first through fourth columns show the listing before fast mode initialization. That is the code you enter in LRN mode and save on a magnetic card. Program locations from 064 to the end are altered or shifted down two locations by the fast mode initialization process. The fifth and sixth columns show the listing after initialization. You cannot save the h12 command at location 064 on a magnetic card. The h12 command will survive turnoff and turnon on the TI-58C.

000	92	RTN	032	82	HIR	064	70	70	096	75	-	064	12	12	096	00	00
001	25	CLR	033	16	16	065	12	B	097	53	(	065	68	NOP	097	05	05
002	35	1/X	034	29	CP	066	38	SIN	098	46	INS	066	61	GTO	098	75	-
003	99	PRT	035	22	INV	067	23	LNX	099	55	÷	067	00	00	099	53	(
004	81	RST	036	67	EQ	068	53	(	100	08	8	068	74	74	100	46	INS
005	55	÷	037	00	00	069	00	0	101	54	)	069	78	Σ+	101	55	÷
006	08	8	038	05	05	070	00	0	102	59	INT	070	98	ADV	102	08	8
007	32	X:T	039	61	GTO	071	00	0	103	65	×	071	22	INV	103	54	)
008	01	1	040	01	01	072	01	1	104	32	X:T	072	94	+/-	104	59	INT
009	00	0	041	25	25	073	82	HIR	105	08	8	073	14	D	105	65	×
010	95	=	042	76	LBL	074	08	08	106	95	=	074	01	1	106	32	X:T
011	82	HIR	043	12	B	075	25	CLR	107	65	×	075	82	HIR	107	08	8
012	06	06	044	32	X:T	076	82	HIR	108	82	HIR	076	08	08	108	95	=
013	22	INV	045	04	4	077	07	07	109	18	18	077	25	CLR	109	65	×
014	59	INT	046	61	GTO	078	32	X:T	110	95	=	078	82	HIR	110	82	HIR
015	82	HIR	047	00	00	079	75	-	111	82	HIR	079	07	07	111	18	18
016	56	56	048	53	53	080	59	INT	112	37	37	080	32	X:T	112	95	=
017	65	×	049	76	LBL	081	22	INV	113	01	1	081	75	-	113	82	HIR
018	01	1	050	11	A	082	77	GE	114	00	0	082	59	INT	114	37	37
019	00	0	051	32	X:T	083	00	00	115	82	HIR	083	22	INV	115	01	1
020	65	×	052	01	1	084	01	01	116	48	48	084	77	GE	116	00	0
021	77	GE	053	85	+	085	95	=	117	32	X:T	085	00	00	117	82	HIR
022	00	00	054	02	2	086	22	INV	118	29	CP	086	01	01	118	48	48
023	01	01	055	52	EE	087	67	EQ	119	22	INV	087	95	=	119	32	X:T
024	82	HIR	056	01	1	088	00	00	120	67	EQ	088	22	INV	120	29	CP
025	18	18	057	02	2	089	01	01	121	00	00	089	67	EQ	121	22	INV
026	95	=	058	94	+/-	090	82	HIR	122	98	98	090	00	00	122	67	EQ
027	82	HIR	059	95	=	091	11	11	123	82	HIR	091	01	01	123	00	00
028	37	37	060	60	DEG	092	87	IFF	124	17	17	092	82	HIR	124	98	98
029	32	X:T	061	22	INV	093	04	04	125	99	PRT	093	11	11	125	82	HIR
030	82	HIR	062	58	FIX	094	00	00	126	81	RST	094	87	IFF	126	17	17
031	48	48	063	86	STF	095	05	05				095	04	04	127	99	PRT
															128	81	RST

Decimal/Octal Conversions (cont)

User Instructions:

1. Key in the program, and record the card if you are using a TI-59.

2. Initialize for fast mode with the following sequence:

   9 Op 17 CLR Cms GTO 064 Pgm 19 SBR 045 P/R LRN

   Ignore flashing displays. If you have done things correctly so far you wil see 064 32 in the display. Continue the initialization with the sequence:

   Ins Ins LRN RST CLR

   Note that there are two inserts in this particular initialization.

3. To convert from octal to decimal, enter the octal value and press A. The calculator prints the decimal value and stops with the decimal equivalent in the display.

4. To convert from decimal to octal, enter the decimal value and press B. The calculator prints the octal value and stops with the octal value in the display.

CLR OR CE AFTER FAST MODE ENTRY FROM THE END OF PARTITION - G. Thomson

Where I used STF IND at the end of the partition followed by 7 and EE for fast mode entry I found that the printed answers were correct but had an added ? . I discovered that if I added a CLR anywhere before the printing all would be well.

Editor's Note: This is an idiosyncrasy which is unique to this method of fast mode entry. The problem seems to come from the error state which exists when the calculator reaches the end of the current partition, and which cannot be cleared at that time or the calculator will not enter fast mode. The effect was described in Patrick Acosta's classic discussion of fast mode entry techniques. The third paragraph on V6N8P4 states:

> "... put LBL E Fix 0 Deg RCL NN STF IND in your program such that the IND is at the last step of the partition. Then you only need to press E (and then in response to the flashing display) press 7 INV (to enter fast mode). In that case, just put a CE instruction as the first step in your fast mode program to clear the error condition caused by the aforementioned procedure."

In the above entry sequence it is assumed that register NN contains the required fast mode entry constant. A discussion of the need to clear the error condition was also buried in the program description for Jovan Puzovic's "1188 Digits of Pi" program (V8N1P21). The CLR or CE required to clear the error state appeared in other programs which used this fast mode entry technique, for example the CLR at location 122 of Peter Messer's "Exact Factorials in Fast Mode" (V8N4P5/6).

INDEX FOR TI PPC NOTES - Bob Fruit. A combined index for TI PPC Notes is available again this year. With a little work the index has been kept to 15 pages so that the cost will be the same as last year, five dollars ($5.00).

Every article appearing in TI PPC Notes for the past four years is in the index. The index is easy to use. If you remember the name of an article just look it up by its name. However, if you are trying to find something about using graphics mode while in fast mode, check the articles under the category Fast Mode and you will quickly find what you are looking for. Also, when there are changes to a program or an article those later references are included in the listing about the article. This means that you would never have a program without also having the corrections for it. The categories in the index are:

April Fools	Club Challenges	Mathematics	Speed Programs
Articles Vol.5	Club Module	Modules	SR-52 Articles
Articles Vol.6	Diagnostics	Newcomer's	SR-56 Articles
Articles Vol.7	Fast Mode	Other Clubs	TI-55 Articles
Articles Vol.8	Firmware	Patent Info.	TI-57 Articles
Authors	Games	Pictures	TI-59 Articles
Benchmark Pgm.	Graphics Mode	Plotting	TI-66 Articles
Brain Teasers	Help Requests	PPX Programs	TI-88 Articles
CC-40 Articles	HIR Codes	Sources of Info.	Utility Pgms.

There are four new categories this year: 1) PPX Programs (so that there will be a concise listing for what's available), 2) CC-40 Articles, 3) TI-66 Articles, and 4) TI-88 Articles (this category will be good for trivia buffs). The category "Programs" was dropped this year. It added a couple of pages to the index and seemed yo be of little use. With that sacrifice, the index was able to be kept to last years's price. A sample segment of the index follows:

TIPPC NOTES SUBJECT INDEX PAGE 10

TITLE	AUTHOR	LOCATIONS
FAST MODE NOT FOR STAT FUNC	HANSON,P	07N3P10
FAST MODE R15 SOFT DISPLAY	SNOW,RICH	05N8P2
FAST MODE SFF,11 OR 12 DIGIT	PRINS,ROB	08N1P10
FAST MODE WHAT IT IS	ARENDT,B	06N6-7P16
GRAPHICS MODE	LEISING,D	06N6-7P11
INV LIST FAST MODE	WORTHING†	05N9-10P15 08N1P12 08N5P2
MODULE SELECTOR	THOMAS,J	08N4P8 08N5P5
PGM-MM-R/S QUIRK, A		08N6P12
SBR IN FAST MODE	LEISING,D	06N1P7
STF IND FAST MODE STEP 239	PRINS,R	08N6P21
FIRMWARE		
512 PGM LOCATIONS IN FIRMWR	HANSON,P	06N9-10P20
BREACKING INTO PROTEC. CARD		05N3P2
CALCULATOR STATUS ROUTINE	BEEBE,B	06N3P9
CARD READER BEWARE GRAPHITE		06N3P10
CIRCULAR STEPPING LIST FIRM.	HANSON,P	07N1-2P11
CODE 27 INV	ALLEN,J	07N1-2P27
CODES 21 AND 26	SWINNEN	06N9-10P4
COMPLETE ANALYSIS OF SELFPGM	BLAYNEY,D	05N1P7
CREATING HEX KEYCODES	ACOSTA,P	07N7-8P18
TI-58 EXTRA 32 STEPS		06N1P15
TRACE QUIRK	MIRANDA,J	06N4-5P17
TRACE,STRANGE	LEWIS,J	05N6P4
TRACE,UNIQUE	MAIRS,J	05N4-5P18
Y^X Y=0	PRINS,R	08N6P3
GAMES		
A-MAZE-ING LABYRINTH	BIEK,A	06N9-10P17 07N1-2P5 05N4-5P22 07N1-2P32
ANOTHER GAME NO RULES	LANE,DAVE	08N1P19
ARITHMETIC EXERCISES SR52	ATHANS,D	05N7P10
BACKGAMMON PPX#918217	SLADEN,B	06N2P7 06N8P11
BATTLESHIP	SNOW	07N1-2P7
BLACKJACK TUTOR	SNOW	07N6P3 07N7-8P28
BOWLING	SNOW	05N6P15 05N7P1
CHESS 2.1	SPERBER,M	07N7-8P20
DRAW POKER TI59 BYTE MAGIZ.	BOYLE	07N7-8P28
DUNGEONS AND DRAGONS	LEISING,D	07N9P3
GAME, ANOTHER DOW JONES AVE	LANE,D	08N3P20
JIVE TURKEY TI-88	SWINNEN,M	07N7-8P13
MAKE UP YOUR MIND	MEUSCH	05N2P10
MAN,FOX,CHICKEN,CORN,RIVER	SKILLMAN	05N1P10

Those people interested in ordering the index should send five dollars ($5.00) U.S. Currency to: Robert Fruit
100 Fuller Road
Hinsdale IL 60521
The indexes will be mailed first class mail.

THE USE OF RTN TO LEAVE FAST MODE - Laurance Leeds. The decimal/octal conversion program by David Mah uses the RST-R/S method to leave fast mode. This technique does provide a "transparent" characteristic for fast mode: that is, once the initialization is complete there is no indication to the user that fast mode is even being used. The question is: "Is there a program sequence which, under program control, will both transfer to normal mode and stop at a location other than 000. I desire to use the 000 region for other purposes, avoiding the RST exit."

The answer is yes. An example appeared in David Lobbestael's "Profile Plot" program (V8N1P24/25). The RTN at location 237 exits fast mode and sends the calculator back to the MU-05 module. TI-59 users may have missed that illustration since the instructions for David's program were for the TI-58C only. Patrick Acosta defined the technique for the TI-59 in V6N8P4:

> "...Assuming that the 13 digit constant needed for fast mode entry is stored in register NN, ... (then with the sequence) ...
>
> LBL A FIX 0 DEG RCL NN STF IND h12 Nop GTO mmm
>
> ... you may use library programs or statistics and conversion functions, then switch to fast mode under program control. Another nice feature, for calculators-alone programs, is that you may call the above LBL A as a subroutine in your normal mode program. Then, when your fast mode segment arrives at the RTN instruction, you may return to normal mode at the point from which you called subroutine A. However, with the printer attached, the calculator seems to return to TRACE mode (At least it did for Palmer Hanson in one of the programs in which this was tried.) Note that the fast mode segment must be the lowest level subroutine. You still cannnot call subroutines in fast mode."

The problem of dropping into TRACE mode at fast mode exit was eventually found to be the use of a fast mode constant which set Flag 9 (again, see Lobbestael's plot program, paragraph 2 on page V8N1P24). The use of only STF rather than STF IND was also established later (V8N6P21).

Laurance wrote a short demonstration program for all of this:

```
050   LBL A Cms SBR 469 CLR Pause RCL 36 R/S
400   25 STO 36 RTN
469   4 9 7 2 + 4 EE 12 = STF
```

The 4 at location 474 avoids setting any flags. Press A to begin. The program stops with a flashing 4. 12 in the display. Press 7 then EE to enter fast mode at location 400 (49*8 + 7 + 1 , from V8N4P15). You see a flashed zero (the CLR at 056 followed by the pause at 057) followed by a steady 25 (recalled from register 36). This technique provides a complete and proper return from fast mode to any desired location.

Editor's Note: This discussion, and CE/CLR discussion on page 13 by George Thomson, illustrate how thorough Patrick Acosta's fast mode article in V6N8P3/4 was. When in doubt on fast mode techniques, study those pages again.

REVISED SIMULTANEOUS EQUATIONS PROGRAM - R. Prins

V8N6P15-17 presented Henrik Ohlsson's program for solution of systems of linear equations, together with modifications by Dejan Ristanovic and Robert Prins which replaced the Pgm-02-SBR-239 method of fast mode entry with the STF at the end of partition method. But the easier fast mode entry was at the expense of printout of the input data, and printout of the input data is important for programs such as these where there is no storage of the input data such that it can be retrieved or corrected later. Robert Prins has provided an additional revision of the Ohlsson program which not only which not provides printout of the input data, but also provides an example of audible prompting.

In this kind of program it is difficult for the user to establish any sort of rhythm for data input since the computation time required before the calculator is ready for the next input varies greatly. Where only visual prompts are available, as with the original Ohlsson program, the user must continually switch his attention from the list of data to be entered to the display to determine when the calculator is ready for the next input. Robert's revision prints the input value, proceeds with the necessary calculations, and then displays and prints the prompt for the next input. The sound of the printing of the prompt provides an audible cue that the calculator is ready for the next input. Of course, if you have never learned the TI-59 keyboard well enough to enter data without looking at the keyboard then this feature is of little interest.

The user instructions are essentially the same as in V8N6P16. Enter the order of the system (n) and press A. The calculator will show a flashing "10".

Press 7 and then EE. See a prompt of "1.01", also printed. Enter the matrix element A and press R/S. The element will be printed, some calculations will take place, and "1.02" will be printed and displayed as the prompt for the next matrix element. When the decimal portion of the prompt is (n+1) you enter the vector element corresponding to the row defined by the integer portion of the prompt. When the last vector element has been entered the calculator will run for a short time and then print and display the first element of the solution. Press R/S again and again to see the remainder of the solution. A sample printout appears at the right. The program listing appears on page 17.

```
          1.01
           0.5
          1.02
    .3333333333
          1.03
          0.25
          1.04
           0.2
          1.05
            1.
          2.01
    .3333333333
          2.02
          0.25
          2.03
           0.2
          2.04
    .1666666667
          2.05
            1.
          3.01
          0.25
          3.02
           0.2
          3.03
    .1666666667
          3.04
    .1428571429
          3.05
            1.
          4.01
           0.2
          4.02
    .1666666667
          4.03
    .1428571429
          4.04
         0.125
          4.05
            1.
-20.00000007
180.0000005
-420.000001
280.0000006
```

Revised Simultaneous Equations Program (cont)

Program Listing:

```
000  92 RTN    040  06  06    080  22  22    120  05  05    160  03  03    200  06  06
001  24 CE     041  54  )     081  93  .     121  01  01    161  69 DP     201  97 DSZ
002  69 DP     042  74 SM*    082  00  0     122  96  96    162  26  26    202  01  01
003  37  37    043  03  03    083  01  1     123  08  8     163  97 DSZ    203  00  00
004  09  9     044  69 DP     084  44 SUM    124  42 STO    164  08  08    204  09  09
005  42 STO    045  23  23    085  04  04    125  03  03    165  01  01    205  69 DP
006  03  03    046  69 DP     086  43 RCL    126  43 RCL    166  38  38    206  34  34
007  42 STO    047  26  26    087  04  04    127  01  01    167  43 RCL    207  73 RC*
008  06  06    048  32 X:T    088  99 PRT    128  42 STO    168  01  01    208  06  06
009  43 RCL    049  97 DSZ    089  91 R/S    129  08  08    169  44 SUM    209  99 PRT
010  07  07    050  02  02    090  99 PRT    130  69 DP     170  06  06    210  91 R/S
011  42 STO    051  00  00    091  72 ST*    131  23  23    171  42 STO    211  69 DP
012  02  02    052  37  37    092  03  03    132  69 DP     172  08  08    212  26  26
013  43 RCL    053  43 RCL    093  69 DP     133  32  32    173  97 DSZ    213  97 DSZ
014  04  04    054  07  07    094  23  23    134  00  0     174  05  05    214  04  04
015  59 INT    055  42 STO    095  00  0     135  63 EX*    175  01  01    215  02  02
016  42 STO    056  02  02    096  63 EX*    136  03  03    176  30  30    216  07  07
017  04  04    057  44 SUM    097  03  03    137  32 X:T    177  43 RCL    217  00  0
018  42 STO    058  03  03    098  69 DP     138  69 DP     178  02  02    218  81 RST
019  05  05    059  97 DSZ    099  33  33    139  23  23    179  44 SUM    219  76 LBL
020  69 DP     060  05  05    100  74 SM*    140  00  0     180  03  03    220  11  A
021  24  24    061  00  00    101  03  03    141  63 EX*    181  69 DP     221  22 INV
022  29 CP     062  26  26    102  43 RCL    142  03  03    182  23  23    222  58 FIX
023  67  EQ    063  93  .     103  05  05    143  75  -     183  00  0     223  94 +/-
024  00  00    064  00  0     104  22 INV    144  43 RCL    184  63 EX*    224  32 X:T
025  63  63    065  01  1     105  64 PD*    145  02  02    185  06  06    225  09  9
026  93  .     066  44 SUM    106  03  03    146  44 SUM    186  69 DP     226  69 DP
027  00  0     067  04  04    107  69 DP     147  03  03    187  26  26    227  17  17
028  01  1     068  43 RCL    108  23  23    148  32 X:T    188  72 ST*    228  47 CMS
029  44 SUM    069  04  04    109  97 DSZ    149  65  x     189  03  03    229  32 X:T
030  04  04    070  99 PRT    110  02  02    150  32 X:T    190  69 DP     230  42 STO
031  43 RCL    071  91 R/S    111  00  00    151  73 RC*    191  23  23    231  01  01
032  04  04    072  99 PRT    112  81  81    152  06  06    192  97 DSZ    232  42 STO
033  99 PRT    073  42 STO    113  43 RCL    153  54  )     193  08  08    233  07  07
034  91 R/S    074  05  05    114  04  04    154  72 ST*    194  01  01    234  93  .
035  99 PRT    075  73 RC*    115  59 INT    155  03  03    195  83  83    235  01  1
036  94 +/-    076  03  03    116  42 STO    156  43 RCL    196  69 DP     236  34 FX
037  65  x     077  44 SUM    117  05  05    157  02  02    197  27  27    237  33 X²
038  32 X:T    078  05  05    118  22 INV    158  22 INV    198  09  9     238  35 1/X
039  73 RC*    079  69 DP     119  97 DSZ    159  44 SUM    199  42 STO    239  86 STF
```

DISTANCE TO THE NEAREST INTEGER - This is another programming challenge from Charlie Williamson who proposed th powers of minus one problem which elicited a large response. The new problem is: "Find the distance from X to its nearest integer". Two classes of solutions are desired, those which use the t-register and those which do not.

RTN IS NOT LIKE R/S IF NOT IN A SUBROUTINE - Don Graham. Page V-58 of Personal Programming states "(INV)(SBR) acts like a Run/Stop if not used within a subroutine." This is not exactly so. If the display happens to be soft when the RTN instruction is encountered, RTN will harden it. R/S will not.

ALPHA CODE CHALLENGE - V8N6P24 carried the following challenge from the Swedish newsletter Programbiten: provide a routine which will change the numbers 1 through 26 into the letters A through Z. Five routines have been received to date, using from 26 through 38 steps, and not counting:the printing commands.

George Thomson and Clyde Durbin submitted straightforward solutions involving t register comparisons. Clyde's second solution is longer, uses one memory register, but no tests or transfers. The empirically tuned solutions of Robert Prins and Jan Bjorklund (forwarded by Robert from the Programbiten challenge) seem to be best, requiring no transfers, no tests, and no memory registers.

Prins

000	76	LBL
001	11	A
002	53	(
003	46	INS
004	85	+
005	53	(
006	53	(
007	46	INS
008	85	+
009	04	4
010	02	2
011	54	)
012	55	÷
013	07	7
014	93	.
015	08	8
016	05	5
017	54	)
018	59	INT
019	65	×
020	02	2
021	93	.
022	05	5
023	54	)
024	59	INT
025	92	RTN

Bjorklund

000	76	LBL
001	11	A
002	85	+
003	53	(
004	53	(
005	24	CE
006	85	+
007	02	2
008	93	.
009	08	8
010	54	)
011	55	÷
012	07	7
013	93	.
014	09	9
015	54	)
016	59	INT
017	65	×
018	02	2
019	93	.
020	05	5
021	85	+
022	01	1
023	02	2
024	93	.
025	05	5
026	95	=
027	59	INT
028	92	RTN

Durbin 1

000	76	LBL
001	11	A
002	85	+
003	32	X:T
004	05	5
005	77	GE
006	38	SIN
007	01	1
008	03	3
009	77	GE
010	39	COS
011	02	2
012	00	0
013	77	GE
014	30	TAN
015	03	3
016	85	+
017	76	LBL
018	30	TAN
019	02	2
020	85	+
021	76	LBL
022	39	COS
023	03	3
024	85	+
025	76	LBL
026	38	SIN
027	01	1
028	02	2
029	95	=
030	92	RTN

Durbin 2

000	76	LBL
001	11	A
002	75	-
003	01	1
004	02	2
005	93	.
006	09	9
007	54	)
008	65	×
009	42	STO
010	00	00
011	69	DP
012	10	10
013	65	×
014	01	1
015	93	.
016	04	4
017	95	=
018	28	LOG
019	59	INT
020	65	×
021	03	3
022	85	+
023	01	1
024	54	)
025	65	×
026	43	RCL
027	00	00
028	69	DP
029	10	10
030	85	+
031	43	RCL
032	00	00
033	85	+
034	02	2
035	09	9
036	95	=
037	92	RTN

Thomson

000	76	LBL
001	11	A
002	42	STO
003	01	01
004	32	X:T
005	01	1
006	02	2
007	44	SUM
008	01	01
009	05	5
010	77	GE
011	30	TAN
012	03	3
013	44	SUM
014	01	01
015	01	1
016	04	4
017	77	GE
018	30	TAN
019	02	2
020	44	SUM
021	01	01
022	02	2
023	00	0
024	77	GE
025	30	TAN
026	03	3
027	44	SUM
028	01	01
029	76	LBL
030	30	TAN
031	43	RCL
032	01	01
033	92	RTN

)MS, INV DMS CHALLENGE - Robert Prins suggests that optimized DMS and INV DMS routines could be useful for fast mode ırograms where the statistics and conversions routines cannot be used. One ıpproach might be to simply use downloaded versions of the firmware. But lobert suggests that we should be able to do better. So the challenge is :o develop more efficient DMS and INV DMS routines.

MEMORY PROTECTION - The TI-58C will not lose memory during the battery removal needed to mount or dismount the calculator from the PC-100. Page A-1 of the 1979 edition of Personal Programming which covers the TI-58C states:

> "If the symptoms of a discharged battery pack are observed on the TI Programmable 58C, turning off the calculator immediately may prevent losing program and data memory contents. Install a fresh battery pack or connect the charger as quickly as possible. ..."

I don't know how long a TI-58C would hold memory without a battery installed. I have verified that mine will hold at least forty-eight hours. Further testing is in progress.

Appendix A of the TI-66 manual states:

> "NOTE: The calculator cannot retain data in its user data memories or program memory when the batteries are removed or become discharged".

Dave Leising reports: Notwithstanding what is said in the manual, there is some small amount of time to replace the batteries before program or data is lost. With one TI-66 no data was lost or altered if the batteries could be replaced in about 30 seconds. Between one and four minutes without batteries, data starts to decay, and an increasing alteration up to about ten per cent of the data occurs by four minutes. But a second TI-66 provided about half of that hold-up time. Beyond about five or six minutes without batteries, some condition is set in the machine causing a special power-on clear to be executed upon battery reinstallation. Unlike the case when the "ON" key is normally pressed, this battery-installation clear causes the partition to be set at 255.31, all memory to be cleared, and the machine comes up in the "on" condition without the "ON" key having been pressed. Dave has found that it is possible to generate unusual program codes (mnemonics) by re-installing the batteries while memory decay is in progress.

What is the memory protection for the CC-40? Can I safely bridge a battery removal by having the AC adapter connected? You will recall that we were cautioned that having the Adapter/Charger connected to a TI-58C or TI=59 with the battery pack removed could damage the calculator. The CC-40 manual provides no information. I did not want to do a test with my CC-40 since I run the risk of destroying all my accumulated programs. Maurice Swinnen says that he has changed batteries without losing his programs. He thinks it took about a minute to make the change. As soon as I have some sort of recording device for the CC-40 I will run the appropriate tests. In the meantime I have asked TI for clarification.

BRAIN TEASER - Laurance Leeds proposes this test of our programming skills: "Find the lowest integer which when divided by each of the primes from 2 through 31 gives a remainder equal to the prime next lower in value than the dividing prime. What are the factors of this number?"

Laurance's program has 177 steps and has a run time of 4 minutes in normal mode.

LOAN SCHEDULE FOR ANY CALENDAR YEAR - Carl Rabe

This program will give any desired calendar year data printout of a portion of a loan repayment schedule. It calculates what the loan status should be as of the beginning of the requested calendar year, and then calculates and prints the monthly data and total interest for that year. The primary input data is the basic information for the beginning of a loan. If that data is not available, the data from the latest available loan statement may be entered using an alternate data entry procedure. It must be expected that some small differences will show up between the results from this program and the statement received from a lender. This is probably due to diffferent truncation and rounding methods, and also from the method of calculating the data for the beginning of the year. However, this program provides a quick cross-check of the statement from a lender. One figure with quite high correlation is annual interest. The program runs in fast mode. Once the input data has been entered and the calculations have been started, the program cannot be stopped until the end of the year is reached. Run time is about 56 seconds. There is no provision for operation without a printer. After running the program to get the printout for a given year, the data for a different year may be requested without reentering the basic inputs.

User Instructions - Primary Mode:

1. Enter the magnetic cards (recorded 6-Op-17) sides 1, 2, and 3.
2. Press SBR CLR to initialize.
3. Enter Interest Rate in percent and press A.
4. Enter the years in the loan and press B.
5. Enter the month and year of the first payment (MM.YYYY) and press C.
6. Enter the loan amount and press D.
7. Enter the year for which data is desired and press E.
8. When a flashing "2" appears press 7 and then EE.

User Instructions - Alternate Procedure:

1. Enter magnetic cards. Press SBR CLR to initialize.
2. Enter the interest rate in percent. Press 2nd A'.
3. Enter the monthly payment. Press 2nd B'.
4. Enter the year of the last available statement. Press 2nd C'.
5. Enter the remaining balance at the end of the last available statement. Press 2nd D'.
6. Enter the year for which data is desired and press 2nd E'.
7. When a flashing "2" appears press 7 and then EE.

A sample printout appears at the right. The program listing is on page 22 for banks 1 and 2. The printer code for the printout annotation for bank 3 appears at the right.

```
XXXXXXXXXXXXXXXXXXX
      12.25    RATE
        30.     YRS
     4.1978    MNYR
      65000.    LOAN
     681.13    PYMT

     1982.00    YEAR
    64001.57    RBAL

        1.00     MTH
      653.35     INT
       27.78     PRC
    63373.79     BAL

        2.       MTH
      653.       INT
       28.06     PRC
    63945.73     BAL

        3.00     MTH
      652.78     INT
       28.35     PRC
          38     BAL

       63801.    MTH
                 INT
                 PRC

        8.00
      651.30
       29.83     PR
    63771.22     BAL

        9.00     MTH
      651.00     INT
       30.13     PRC
    63741.09     BAL

       10.00     MTH
      650.69     INT
       30.44     PRC
    63710.65     BAL

       11.00     MTH
      650.38     INT
       30.75     PRC
    63679.90     BAL

       12.00     MTH
      650.07     INT
       31.06     PRC
    63648.84     BAL


     7820.83    ΣINT
XXXXXXXXXXXXXXXXXXX
```

```
6666666666.    30
  35133717.    31
    453536.    32
  30314535.    33
  27321331.    34
  33453037.    35
  45171335.    36
  35141327.    37
    303723.    38
    243137.    39
    333515.    40
    141327.    41
  77243137.    42
```

Loan Schedule for Any Calendar Year - (cont)

Program Listing:

```
000 25 CLR   080 54 )   160 61 GTO   240 44 SUM   320 09 09   400 01 01
001 22 INV   081 42 STO   161 04 04   241 14 14   321 01 1   401 94 +/-
002 58 FIX   082 03 03   162 26 26   242 43 RCL   322 00 0   402 85 +
003 29 CP   083 92 RTN   163 68 NOP   243 37 37   323 68 NOP   403 43 RCL
004 69 OP   084 76 LBL   164 75 -   244 61 GTO   324 32 X:T   404 12 12
005 00 00   085 13 C   165 01 1   245 00 00   325 43 RCL   405 54 )
006 91 R/S   086 58 FIX   166 75 -   246 09 09   326 10 10   406 23 LNX
007 76 LBL   087 04 04   167 43 RCL   247 98 ADV   327 22 INV   407 94 +/-
008 99 PRT   088 32 X:T   168 07 07   248 00 0   328 77 GE   408 85 +
009 55 +   089 43 RCL   169 54 )   249 32 X:T   329 03 03   409 43 RCL
010 01 1   090 33 33   170 65 ×   250 43 RCL   330 36 36   410 13 13
011 02 2   091 71 SBR   171 01 1   251 09 09   331 98 ADV   411 54 )
012 22 INV   092 99 PRT   172 02 2   252 94 +/-   332 97 DSZ   412 55 ÷
013 28 LOG   093 45 YX   173 85 +   253 85 +   333 09 09   413 43 RCL
014 85 +   094 32 X:T   174 43 RCL   254 01 1   334 02 02   414 02 02
015 01 1   095 01 1   175 06 06   255 03 3   335 48 48   415 23 LNX
016 54 )   096 54 )   176 75 -   256 54 )   336 98 ADV   416 54 )
017 82 HIR   097 59 INT   177 43 RCL   257 32 X:T   337 43 RCL   417 55 ÷
018 08 08   098 94 +/-   178 03 03   258 02 2   338 11 11   418 01 1
019 32 X:T   099 85 +   179 54 )   259 06 6   339 32 X:T   419 02 2
020 69 OP   100 01 1   180 94 +/-   260 08 8   340 03 3   420 54 )
021 06 06   101 03 3   181 42 STO   261 42 STO   341 05 5   421 71 SBR
022 87 IFF   102 54 )   182 00 00   262 14 14   342 00 0   422 12 B
023 01 01   103 42 STO   183 04 4   263 43 RCL   343 42 STO   423 61 GTO
024 00 00   104 06 06   184 05 5   264 38 38   344 14 14   424 04 04
025 27 27   105 32 X:T   185 61 GTO   265 61 GTO   345 43 RCL   425 62 62
026 92 RTN   106 22 INV   186 00 00   266 00 00   346 42 42   426 42 STO
027 83 GD*   107 59 INT   187 29 29   267 09 09   347 61 GTO   427 05 05
028 14 14   108 65 ×   188 35 1/X   268 43 RCL   348 00 00   428 52 EE
029 44 SUM   109 04 4   189 65 ×   269 10 10   349 09 09   429 22 INV
030 15 15   110 22 INV   190 43 RCL   270 65 ×   350 76 LBL   430 52 EE
031 43 RCL   111 28 LOG   191 05 05   271 43 RCL   351 25 CLR   431 42 STO
032 02 02   112 54 )   192 54 )   272 01 01   352 29 CP   432 16 16
033 45 YX   113 42 STO   193 42 STO   273 54 )   353 43 RCL   433 43 RCL
034 43 RCL   114 07 07   194 10 10   274 32 X:T   354 30 30   434 08 08
035 00 00   115 81 RST   195 98 ADV   275 01 1   355 69 OP   435 61 GTO
036 94 +/-   116 76 LBL   196 43 RCL   276 06 6   356 01 01   436 01 01
037 54 )   117 14 D   197 08 08   277 44 SUM   357 69 OP   437 64 64
038 94 +/-   118 32 X:T   198 32 X:T   278 14 14   358 02 02   438 76 LBL
039 85 +   119 43 RCL   199 04 4   279 43 RCL   359 69 OP   439 17 B'
040 01 1   120 34 34   200 08 8   280 39 39   360 03 03   440 99 PRT
041 54 )   121 71 SBR   201 44 SUM   281 61 GTO   361 69 OP   441 42 STO
042 55 ÷   122 99 PRT   202 14 14   282 00 00   362 04 04   442 12 12
043 43 RCL   123 42 STO   203 43 RCL   283 09 09   363 69 OP   443 23 LNX
044 01 01   124 04 04   204 36 36   284 52 EE   364 05 05   444 42 STO
045 54 )   125 81 RST   205 61 GTO   285 22 INV   365 25 CLR   445 13 13
046 35 1/X   126 86 STF   206 00 00   286 52 EE   366 22 INV   446 81 RST
047 83 GD*   127 01 01   207 09 09   287 44 SUM   367 58 FIX   447 00 0
048 15 15   128 00 0   208 43 RCL   288 11 11   368 66 PAU   448 00 0
049 76 LBL   129 42 STO   209 07 07   289 94 +/-   369 81 RST   449 00 0
050 11 A   130 15 15   210 32 X:T   290 85 +   370 76 LBL   450 00 0
051 32 X:T   131 42 STO   211 43 RCL   291 43 RCL   371 16 A'   451 00 0
052 43 RCL   132 11 11   212 08 08   292 16 16   372 11 A   452 00 0
053 31 31   133 43 RCL   213 67 EQ   293 54 )   373 76 LBL   453 00 0
054 71 SBR   134 03 03   214 02 02   294 32 X:T   374 19 D'   454 25 CLR
055 99 PRT   135 42 STO   215 29 29   295 02 2   375 14 D   455 61 GTO
056 55 +   136 00 00   216 01 1   296 00 0   376 81 RST   456 01 01
057 01 1   137 01 1   217 02 2   297 44 SUM   377 68 NOP   457 26 26
058 02 2   138 04 4   218 42 STO   298 14 14   378 76 LBL   458 76 LBL
059 00 0   139 03 3   219 09 09   299 43 RCL   379 18 C'   459 15 E
060 00 0   140 61 GTO   220 43 RCL   300 40 40   380 85 +   460 42 STO
061 54 )   141 00 00   221 10 10   301 61 GTO   381 01 1   461 08 08
062 42 STO   142 29 29   222 61 GTO   302 00 00   382 54 )   462 05 5
063 01 01   143 65 ×   223 02 02   303 09 09   383 55 ÷   463 06 6
064 85 +   144 43 RCL   224 37 37   304 94 +/-   384 04 4   464 05 5
065 01 1   145 04 04   225 98 ADV   305 85 +   385 22 INV   465 02 2
066 54 )   146 54 )   226 61 GTO   306 43 RCL   386 28 LOG   466 52 EE
067 42 STO   147 58 FIX   227 02 02   307 10 10   387 54 )   467 01 1
068 02 02   148 02 02   228 87 87   308 54 )   388 85 +   468 02 2
069 81 RST   149 32 X:T   229 43 RCL   309 42 STO   389 01 1   469 94 +/-
070 76 LBL   150 01 1   230 06 06   310 10 10   390 95 =   470 85 +
071 12 B   151 06 6   231 42 STO   311 32 X:T   391 13 C   471 02 2
072 32 X:T   152 00 0   232 09 09   312 01 1   392 76 LBL   472 95 =
073 43 RCL   153 42 STO   233 43 RCL   313 07 7   393 10 E'   473 22 INV
074 32 32   154 14 14   234 04 04   314 44 SUM   394 42 STO   474 52 EE
075 71 SBR   155 43 RCL   235 42 STO   315 14 14   395 08 08   475 58 FIX
076 99 PRT   156 35 35   236 10 10   316 43 RCL   396 43 RCL   476 00 00
077 65 ×   157 61 GTO   237 32 X:T   317 41 41   397 04 04   477 60 DEG
078 01 1   158 00 00   238 03 3   318 61 GTO   398 65 ×   478 86 STF
079 02 2   159 09 09   239 09 9   319 00 00   399 43 RCL   479 40 IND
```

EXACT SOLUTIONS FOR TEST PROBLEMS - George Thomson

Editor's Note: V8N6P18/19/20 discussed the measurement of the accuracy of solutions for systems of linear equations. A proposed test case was a 7x7 matrix with the sub-Hilbert form and a unity vector on the right hand side. George had provided an exact solution for that case. but it would be convenient to have the exact solutions for systems of other orders. George writes:

I deduced the general term for a matrix of any size n using your Radio Shack Model 100 results which are almost exact up to n = 7, and of the general magnitude for n = 8, 9, and 10. Let t be the term number in the solution of n equations, then each term equals:

$$(-1)^{n+t} \cdot \frac{(n+t)!}{t!\ (t-1)!\ (n-t)!}$$

Of course, either t or (t-1) factorial need to be calculated. It is much more rapid to calculate the terms for a given n using the relationship between successive terms:

$$term_t = (-1) \cdot \frac{(n+t)(n-t+1)}{t(t-1)} \cdot term_{t-1}$$

where it is known that term 1 is always

$$Term_1 = (-1)^{n+1}(n)(n+1)$$

I wrote two ad-hoc programs which calculate the terms either starting with n = 2 or with any other n. The first program used MU-11 for the three factorials and was very slow. The second used the successive term idea and I let it go up to an n of 100 (about five hours). It did not overflow. The largest term was -1.3777514E76 at n = 100, t = 71. I checked this using the table of log factorials to seven digits in Biometrika Tables for statisticians (1 - 1000), and got the same result within rounding error: 1.377751E76. Of course these high values are just a game for the TI-59.

A sample printout appears at the right. The program listing is on the next page. To start at n = 2 simply press A. To start at any other n, enter n and press A'.

```
         2.  SIZE

        -6.
        12.

         3.  SIZE

        12.
       -60.
        60.

         4.  SIZE

       -20.
       180.
      -120.
       280.

         5.  SIZE

        30.
      -420.
      1680.
     -2520.
      1260.

         6.  SIZE

       -42.
       840.
     -5040.
     12600.
    -13860.
      5544.

         7.  SIZE

        56.
     -1512.
     12600.
    -46200.
     83160.
    -72072.
     24024.

         8.  SIZE

       -72.
      2520.
    -27720.
    138600.
   -360360.
    504504.
   -360360.
    102960.

         9.  SIZE

        90.
     -3960.
     55440.
   -360360.
   1261260.
  -2522520.
   2882880.
  -1750320.
    437580.
```

Exact Solutions for Test Problems - (cont)

```
000  76 LBL    025  00  00    050  42 STO    075  43 RCL    100  99 PRT
001  11  A      026  43 RCL    051  13  13    076  03  03    101  69 OP
002  02  2      027  11  11    052  65  ×     077  54  )     102  23  23
003  76 LBL    028  69 OP     053  71 SBR    078  42 STO    103  97 DSZ
004  16 A'     029  04  04    054  88 DMS    079  05  05    104  04  04
005  42 STO    030  43 RCL    055  95  =     080  55  ÷     105  39 COS
006  01  01    031  01  01    056  99 PRT    081  43 RCL    106  69 OP
007  70 RAD    032  42 STO    057  02  2     082  03  03    107  21  21
008  03  3      033  04  04    058  42 STO    083  55  ÷     108  98 ADV
009  06  6      034  69 OP     059  03  03    084  53  (     109  97 DSZ
010  02  2      035  34  34    060  76 LBL    085  43 RCL    110  02  02
011  04  4      036  69 OP     061  39 COS    086  03  03    111  38 SIN
012  04  4      037  06  06    062  43 RCL    087  75  -     112  76 LBL
013  06  6      038  98 ADV    063  01  01    088  01  1     113  88 DMS
014  01  1      039  43 RCL    064  75  -     089  54  )     114  53  (
015  07  7      040  01  01    065  43 RCL    090  65  ×     115  53  (
016  42 STO    041  85  +     066  03  03    091  43 RCL    116  43 RCL
017  11  11    042  01  1     067  85  +     092  13  13    117  05  05
018  09  9      043  95  =     068  01  1     093  95  =     118  65  ×
019  09  9      044  42 STO    069  95  =     094  42 STO    119  89  π
020  42 STO    045  05  05    070  65  ×     095  13  13    120  54  )
021  02  02    046  65  ×     071  53  (     096  65  ×     121  39 COS
022  76 LBL    047  43 RCL    072  43 RCL    097  71 SBR    122  54  )
023  38 SIN    048  01  01    073  01  01    098  88 DMS    123  92 RTN
024  69 OP     049  95  =     074  85  +     099  95  =     124  00  0
```

MAILBAG - There were many comments returned with interest section of the subscription form for 1984. The following representative sample shows that there is a wide range of opinions as to the material that should be covered in TI PPC Notes. Not presented are many comments expressing general satisfaction with the content in 1983. I tend to cover subjects for which there is member response, and TI-59 related subjects continue to dominate reader response. Therefore, my intent for the remainder of the year is to continue to emphasize TI-59 useage, and to cover the newer developments such as the CC-40, the BA-55 and the TI-66 in proportion to indicated interest. I will also continue to present comparisons with non-TI produced products. Here are some reader comments:

"I would like to have some kind of program exchange network for learning and exchanging programs in either BASIC or FORTRAN or both. ... I do not think PPC should restrict itself to only TI products ..." R.B.

"There are a million magazines on the market that cover BASIC, etc., but there is only one that covers TI programmable calculators (TI PPC Notes). I would prefer to see TI PPC Notes just cover the traditional types of PPC's." K.B.

"I am not a mathematician and programs on square roots, sines, etc., leave me cold. I am interested in practical programs, such as a 1983 IRS Income Tax Program." F.G

"Its time to consider other programmable calculators since we never know when TI will drop their support . How about letting manufacturers know of the need for scientific programmable calculators and a next generation replacement for the 59?" S.H.

"Particularly interested in new products, and in scientific software developments (e.g., Fortran compilers) for home computers. Would like to see a comparison of extended precision features available." B.E.

GENERATING 64 - In V8N6P11 Myer Boland proposed this brain-teaser, which was to obtain the number 64 in the display in a minimum number of steps, but without using the number keys. You were not allowed to assume the start-up condition for the calculator. Myer provided a 14 step solution, and challenged readers to do better.

Laurance Leeds submitted three ten step solutions each of which generated a four in the display and then used y^x pi INT to raise the four to the third power. One of his solutions was:

CLR Cos + CE) x^2 y^x Pi INT =

Gregory Hoen submitted a similar ten step solution

CLR Cos + CE = y^x Pi INT = x^2

Former editor Maurice Swinnen submitted seven solutions starting with one of twelve steps and ending with the following 8 step solution:

CLR GRAD Cos + Sin 1/x INT)

Peter Stromgren submitted an eight step routine

DEG Pi Tan 1/x Lnx x^2 INT x^2

W. J. Widmer submitted an eight step routine

= Pi - cos) INV tan INT

and also submitted a seven step routine

Cms $\Sigma+$ $\Sigma+$ x^2 y^x $\Sigma+$)

Peter Stromgren also submitted a seven step program

Cms $\Sigma+$ $\Sigma+$ y^x $\Sigma+$ = x^2

but noted that routines based on the use of the statistics routines do not meet the criteria that we were not allowed to assume the calculator condition. This routine will not run in O-Op-17. The same comment applies to W. J. Widmer's seven step routine.

W. J. Widmer also investigated the 64 problem for calculators ranging from the TI-55, TI-57 and TI-57LCD. He found a curious quirk with the TI-55 (not the TI-55II). A routine similar to his seven step routine above which first clears the statistics registers and then uses the sequence

$\Sigma+$ $\Sigma+$ x^2 y^x $\Sigma+$)

yields 125, not the anticipated 64. The same quirk occurs with the TI-MBA. The quirk occurs because the y^x function and the automatic incrementing for trend line analysis share the use of the y register. (That a feature was not carried forward to later calculators such as the TI-59, where the y^x function moves the display value to the first hierarchy register) The four which is in the display of the MBA after the x^2 in the sequence is transferred to the y register by the y^x. Then, then at the next $\Sigma+$, automatic incrementing changes the contents of the y register to a five, and brings a three to the display. The) completes the y^x calculation yielding five to the third power or 125.