TRS-80™ COMPUTING

VOL. 1, NO. 2 24 PAGES $1.50

CONTENTS

CASSETTE:
Without shedding tears, by John Strong 12-14
Faults checklist, by Bob Schuetz 11
Auto-level on Play, by Robert A. Cahill 11
Replace relay in TRS-80 11
Hear Csave, Cload, by Steve Gibson 11
FORT WORTH REPORT, by John Strong:
Tandy death saddens annual meeting,
New manuals on the way,
CTR-80 surfaces 23
LOWERCASE:
Hardware needed, and
Here's software, by Timothy P. Mann 3,4
HARDWARE:
Expansion box schematic, by Patrick J. McMahon 16-19
Convert to Level II, by Thomas B. Perera 15
Clock speedup, by Steve Gibson 10
Power-line glitches, disk woes 10
Key bounce, by Bill Roberts 9
Allows use of 'Brand X' tape, Steve Gibson 7
Keyboard scanning, by William Barden, Jr. 2,3
Disk interfacing, by William Barden, Jr. 8
SOFTWARE:
Level II index 7
Level I & II ROM routines, by Ed Faulk 5-7
Accuracy: inelegant solutions, by Robert M. Richardson . 9
Blackjack fix, by J. P. Rahn 9
PROGRAMS:
Journal/Ledger (Level I), by Frank C. Heinisch 20,21
Star trek (Level I), by George W. Scheil 21,22

Here's how TRS-80 knows you're fingering its 53 keys

By WILLIAM BARDEN, Jr.

Ever wonder how the TRS-80 decodes the keyboard and changes the key depressions to an ASCII code? Well I haven't either, so let's move on to the next topic...

But seriously, to understand the whole process, let's first look at the hardware implementation of the keyboard and associated logic.

Figure 1 shows the electrical arrangement of the keyboard. There are eight rows called A0 through A7 which represent the low eight order bits of the address bus. These, of course, are constantly active when the Z-80 CPU addresses memory. Eight columns, D0 through D7, are outputs to the data bus.

The CPU uses the data bus to read data from memory or I/O devices one byte (eight bits) at a time. An enable signal Kybd* is active only when an instruction in the program addresses the keyboard by specifying address 3800H through 38FFH. We'll get into that later.

Both the 74LS05s and the 74LS368s are inverters. When a high voltage of three to five volts is applied to their inputs, the outputs go to a low value of about zero volts. We'll refer to the high value as "high" and the low value as "low" in the following discussion.

The outputs of the 74LS05s follow the address lines, swinging from low to high and back again as the CPU addresses memory in random fashion. The outputs of the 74LS368s, however must be enabled by signal Kybd* to reflect the opposite of their inputs. When signal Kybd* is not active (high) then the outputs of the 74-LS368s are effectively disconnected from the data bus D0 through D7.

Now let's see what happens when a key is pressed. When the keyboard is being addressed by an instruction, signal Kybd* goes low, enabling the 74LS368s. Outputs D0 through D7 are gated to the data bus D0 through D7. The outputs are the inverse of the inputs on each 74LS368. When no keys are depressed, the inputs on each 74LS368 are high (+5V) and the outputs swing to low or zero. When the keyboard is "read", then, and no keys are depressed, a byte of all zeroes is read.

Suppose key "2" is pressed at the same time the keyboard is read. The key will bounce for about a ten thousandths of a second or so and then settle down to a firm "make". Now one side of the key connects to the output of the A4 inverter. If A4 is low (address not active) then the output of the A4 inverter is high, and the output on D2 is a low as without a key depression. If A4 is high, however, then the inverter output is low and the key shorts the input to the D2 inverter to (effectively) ground, making the input to the D2 74LS368 low and the D2 output high or one.

Therefore, when the keyboard is read during a key depression there will be a one for the output of every data bit D0 through D7 with a corresponding key depression AND address line high. If only one address line A0 through A7 is high, there will be a one for the data bit if, and only if, the key at the junction of the address line and data input line is pressed.

SOFTWARE

Now let's see what happens in the software. The following discussion describes Level I, only because I haven't looked at Level II at this point, but the approach will be the same.

When the CPU performs a "LD A, (387F)" instruction, signal Kybd* goes low as the "38" portion of the address is detected. Address lines 7 through 0 (high to low order) also go high at the same time. The outputs of D0 through D7 remain low unless a key is making. In this case, one of the data lines will go high (or more than one if more than one key is pushed). The data byte corresponding to the data inverter outputs is loaded into the CPU A register by the LD instruction.

Now the program tests the data byte. If it is not zero, at least one key has been pushed (probably only one) and the program calls another subroutine to find which key it is. If the data byte is zero, the program returns a zero, indicating no key is pressed.

In the case of a non-zero byte, the second subroutine tests which key is active. First, however, it delays 2 milliseconds (2 thousandths of a second) for the key to bounce back and forth between making and breaking. After the delay, the subroutine then reads one row at a time by performing an "LD A,(38XXH)" instruction. A series of seven LDs are done with addresses of 3801, 3802, 3804, 3808, 3810, 3820, and 3840. If a zero is read back on all, then a return is made to the higher level subroutine with a "zero" key indication. This could happen if the timing was just right.

If one of the eight LDs results in a non-zero byte then a third level of action is performed. This code performs the following algorithm:

1) A constant value of 3F, 47, 4F, 57, 2F, 37, or 00 is saved, depending upon the address which resulted in the non-zero value—addresses of 3801, 3802, 3804, 3808, 3810, 3820, 3840 respectively.
2) The bit position of the data bit plus one is added to this base constant. For example, if key J was depressed when address 3802 was used, the data byte would be 00000100 and 2 plus 1 would be added to 47H to give 4AH.
3) The resulting value is "pretty close" to ASCII. How some adjustments are made (Level I), depending upon the range of values the number is in. For a value equal to or greater than hex 40 the number is left in uppercase only. For from 3CH to 3FH the value is left alone if the Shift key is active or AND-ed with 2F if the Shift key is not active. For a value between 30H and 3BH the value is left alone if the Shift key

(Continued on page 3)

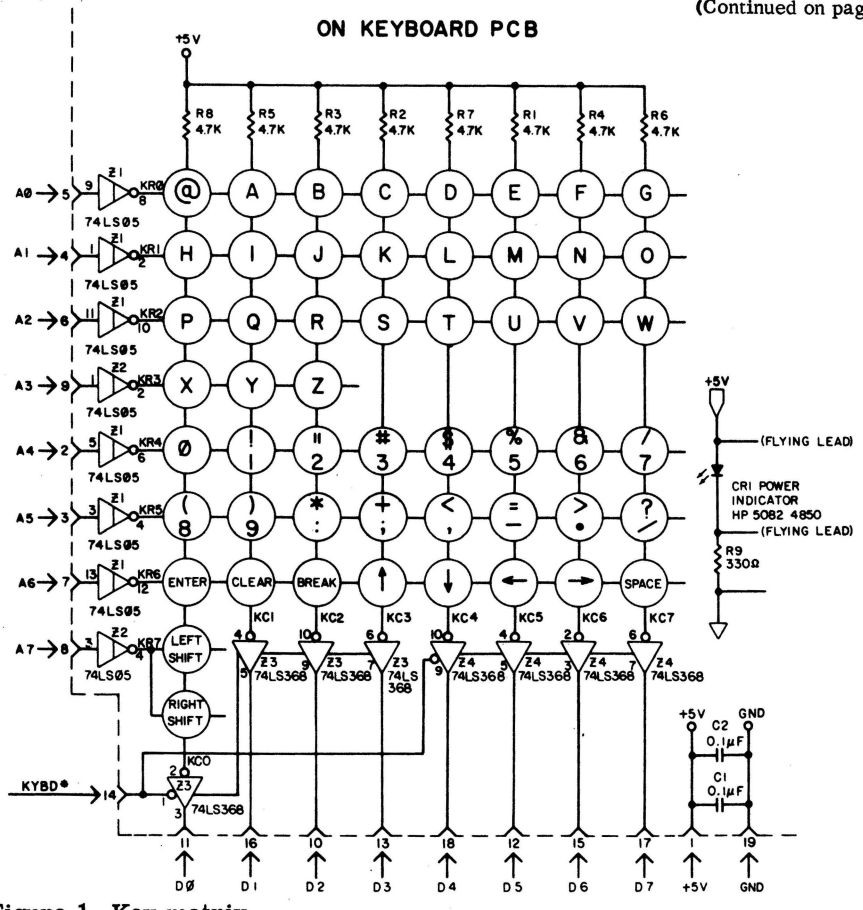


Figure 1—Key matrix.

There's hardware needed for that lowercase software!

I was gratified to see my note on lowercase software in the CIE TRS-80 Bulletin 1:3, byt my happiness didn't last too long after getting the issue.

The very next day I began getting letters from people saying the programs don't give them lowercase, as advised.

Let me first assure you that there were no typos in the listing, as printed, and that the programs worked fine on my computer.

So what was the problem? The letters I got fell into two categories. The first were those who missed the first issue of TRS-80 Computing, or for some other reason failed to get the point that my software is supposed to work together with hardware modification to the TRS-80. This did not come through clearly when my letter was read by itself, and some people evidently thought my program would run on an unmodified computer. It doesn't.

LIKINS' MODIFICATIONS

The second problem came from those readers who used Dan Likins' second version of the hardware mod, the one which is switched on and off by a bit in output port FF.

Now, as published in TRS-80 Computing 1:1, the mod is turned ON by a zero and OFF by a one in bit 4 of the port. If it is actually installed this way, however, Level II Basic will keep turning it on when it should be off. Apparently Likins is now installing the mod the other way around to avoid this problem, but in this case Basic will keep turning the mod off, and my programs won't work.

Fortunately, there is a solution to this. The statement Poke 16445,16 will cause Basic to turn on bit 4 of the output port and keep it on.

All in all, I think the best way of doing the modification is to install a small SPDT switch in the back of the computer to shut it off, rather than fooling around with spare gates and output ports. That is the way my computer is wired, and it has proved most satisfactory.

HOW TO WIRE IT

The way to do this is to piggyback a new 2102 over IC Z45, just as described in TRS-80 Computing 1:1, leaving pins 11 and 12 unconnected.

One wire should go from the new RAM's pin 11 to Z60 pin 5, a second wire should run from Z60 pin 4 to the common terminal of the new switch, and the other two terminals of the switch should go to Z30 pin 13 and the new RAM's pin 12.

Finally, the narrow trace running from Z30 pin 13, under the chip, and out the other side on the component side of the board, should be cut.

Now the computer can be used in either upper or lowercase mode, as desired, and there will be no interference from Radio Shack software turning it on or off at unwanted times.

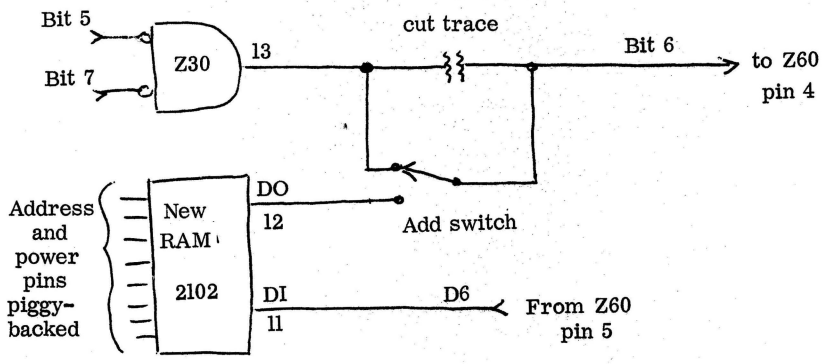


FIGURE 1—Switched lowercase modification for TRS-80.

Figure 1 shows the resulting circuit.

Just as I was about to send this to TRS-80 Computing, I received a call from a man who has the lowercase modification which is sold with the Electric Pencil text formatter. This modificaton works essentially the same way as my version, with a SPDT switch. Just flip the switch to the lowercase position after loading my software routine, and you're ready to go.

— Tim Mann

TRS-80 COMPUTING

TRS-80 Computing is published as often as monthly by Computer Information Exchange, Inc., a nonprofit educational corporation, box 158, San Luis Rey CA 92068.

Subscription rates in the U.S. are $15 for 12 issues. To Canada and Mexico, subscriptions are $18US for 12 issues; all other countries $27US.

No advertising is accepted. Free editorial space is given any commercial product that might be of interest to TRS-80 users. One free page of dollars-off coupons will be made available to product vendors, on a space-available basis.

In keeping with TRS-80 Computing's nonprofit educational aims, any amount of material from each issue may be reproduced by any not-for-profit educational group or institution, without prior request. Tearsheet is requested. Required credit includes: "TRS-80 Computing, box 158, San Luis Rey CA 92068; $10 for 12 issues."

Clubs having regular newsletters may receive TRS-80 Computing in exchange, providing CIE is also granted similar reprint privileges.

Clubs or anyone having audio tapes of speakers that might be of interest to CIE readers should submit them to the editor, Bill McLaughlin. If accepted, they will be typeset free, with a copy returned with the tape. Office phone is (714) 757-4849.

Editorial contributions of all kinds are requested (software, hardware and applications articles, letters, etc.) and may be submitted either written or on cassette.

TRS-80 is a Tandy Corporation trademark licensed to Computer Information Exchange. Computer Information Exchange is solely responsible for the editorial content of this magazine and is not an agent, subsidiary, or otherwise affiliated with Tandy Corporation.

KEY DECODING, Cont.:

is not ANDed with 2F if the Shift key is active. For values less than 30H a second look-up is done to find the result if the Shift key is not active or ANDed with 3 F if the Shift key is active. To find out whether the Shift key is active the program reads row A7 by addressing 3880H. Confusing? Yes.

3 The table at right is a best attempt to show the conversion. Correct me if I'm wrong.

4) The subroutine is now exited with an ASCII-like value as the code.

—Reprinted from Orange County TRS-80 Users' Group Newsletter.

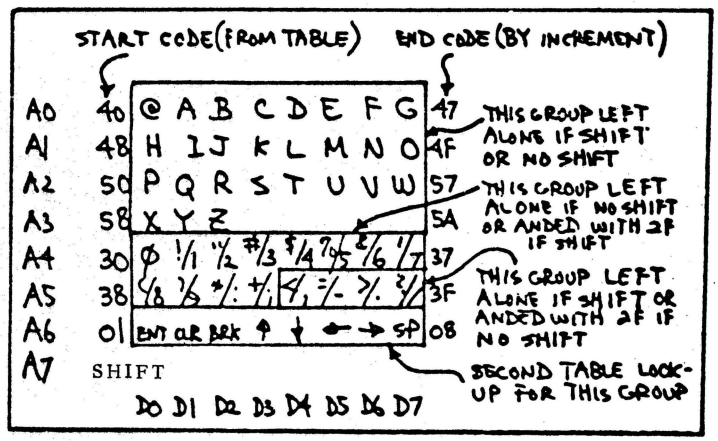


Table shows how TRS-80 converts keystrokes to ASCII.

Modification allows use of 'Brand X' tape

By STEVE GIBSON

The circuit in the cassette record interface for the TRS-80 computer uses a dynamic threshhold detector, which is IC Z4 (see page 27 of TRS-80 Computing 1:1).

Very clever because it's possible to have the data right on top of 60 cycle AC hum and still be able to recover the data.

The way it does this is that R45 and C39 form an RC circuit with the time constant of about 47 milliseconds. We can ignore the impedence presented by CR6 and CR7, the two diodes that look into it, and we can assume that the input impedence of Z4 is darn near zero which gives us a simple RC network.

By changing the value of C39 to a .01 uf capacitor for a Level I machine (hardly worth the effort), but more importantly—changing the value of C39 to a .005 uf capacitor for a Level II machine, it is possible to adjust the time constant of the RC network (ie C39 and R35) to dynamically adjust the gain threshhold of this circuit. With this we can actually perform automatic level adjustment from sync pulse to sync pulse.

In this way it is possible to actually compensate for tape dropouts. It is possible then to recover Level II tapes off of Brand X cassettes—a lot easier than if you used the classic circuit.

There is one "gotcha" in this modification. The designers of this interface method, (this was a "The Computer Hobbyist" interface, so it was Hal Chamberlin's development), by utilizing a .1 uf capacitor, make it possible for the circuit to be very forgiving if you happen to set the cassette volume control to something other than what it should be. As a result, if you make the modification, all you are going to have to do is to be very careful about your volume settings, so that you present to the cassette interface an idealized volume.

HOW TO SET VOLUME

A clever way to do that is to take a simple transistor radio and tune it to about 540KHz, at the lower end of the broadcast band, and set it over the nameplate on the keyboard, where it says TRS-80.

Put the computer in the Cload mode and experiment with this. You'll have to hit Reset afterwards.

Play a tape into the computer and you'll hear a lot of noise—interference—on the radio. When you increase the tape recorder volume, you'll discover that there will be a point where the noise will change. This changing point marks where the computer will no longer accept the data.

As a result, you have a way of finding out what volume setting you should use on your tape recorder.

Find the point where the sound changes, back it off one half notch and reset the computer and go for Cload.

TRS-80 cassettes listed quarterly

Everyone seems to be writing programs for TRS-80. There are so many cassettes for these computers that you need a computerized reference list to keep track of them all.

Robert Elliott Purser has been compiling his list of these computer cassettes for over a year. The list has more than 1,000 TRS-80, Pet and Apple cassettes for sale or trade from over 100 sources. Purser's list is now being published quarterly and every issue seems to double in size.

The February '79 edition will be expanded to include photographs of the computer's display screen for some of the programs so you can see what the program will do on your computer.

To receive this computer cassette reference list every quarter, send $12 for a one year's subscription. Single copies are $4 ppd. Dealers discounts are available.

The November '78 issue was an introductory half-priced edition. While the supply lasts, these are only $2 a copy ppd.

Those having original software for sale or trade should send a list of cassettes to Purser, Box 466, El Dorado, CA 95623.

Software solves lowercase puzzle

By TIMOTHY P. MANN

I have a software solution to the problem of lowercase not working with Level II Basic.

Just last week I spent several evenings disassembling the Level II video display driver routine.

Right near the beginning of this routine are the instructions which convert all upper and lower case ASCII characters to control codes before storing them in video memory.

It takes only a 30-byte machine code program protected at the high end of memory (and a change of the driver address in the video display DCB) to bypass this part of the driver and then patch back to it.

After this change is made, everything in Level II Basic, including prompts, Print statements, Chr$ functions, etc., will work entirely as they should with lowercase installed.

To type lowercase letters on the keyboard, use the shift key with the uppercase keys.

Basic will recognize and properly compress keywords typed in lowercase, but when they are expanded in listings, they will always appear in uppercase. Variable names typed in lowercase will also be converted to uppercase, but comments and strings are left unchanged. As an extra bonus, some of the error messages are automatically printed in lowercase with the first letter capitalized.

Now for the details: If you have Radio Shack's Editor/Assembler (which I highly recommend), type in the following program and assemble it:

```
        ORG 32738D
LCFIX   LD L, (IX+3)    ;Cursor address
        LD H, (IX+4)
        JP C, Ø49AH     ;Go if reading
                         from screen.
        LD A, (IX+5)    ;Is cursor on?
        OR A
        JR Z, RETRV
        LD (HL), A      ;Blank it if so.
RETRV   LD A, C
        CP 2ØH          ;Control?
        JP C, Ø5Ø6H     ;Go if so.
        CP 8ØH          ;Graphic or space comp?
        JP NC, Ø4A6H    ;Go if either.
        JP Ø47DH        ;Patch back to ROM.
;Automatically change address in DCB.
        ORG 4Ø1EH
        DEFW LCFIX
        END
```

Now when you power up your lowercase-equipped TRS-80, use a memory size of 32737 and immediately load the object code output tape from the assembler with the System command. After loading, hit Break to return to Basic; do not branch to the program with the / command. Nothing further is necessary; the program patches itself in automatically.

If you don't have an assembler, run the following Basic program right after power-up. (Here again, memory size must be set at 32737):

```
5   REM—FOR 16K MACHINES
1Ø  FOR I=32738 TO 32767
2Ø  READ BYTE
3Ø  POKE I, BYTE
4Ø  NEXT I
5Ø  POKE 16414,226: POKE 16415,127
6Ø  END
7Ø  DATA 221,11Ø,3,221,1Ø2,4,218,154
8Ø  DATA 4,221,126,5,183,4Ø,1,119
9Ø  DATA 121,254,32,218,6,5,254,128
1ØØ DATA 21Ø,166,4,195,125,4
```

For a 4K machine, set memory size to 2Ø449, and change lines 1Ø and 5Ø in the above program to read:

```
1Ø FOR I=2Ø45Ø TO 2Ø479
5Ø POKE 16414,226:POKE 16415,79
```

One final note: If you would like the keyboard decoding changed so that you get lowercase without shift and uppercase with, or other modifications, this can also be done in software, but requires considerably more effort. And one final plug—I and my friend and associate Glenn Schulz of NDM Designs have all sorts of goodies for the TRS-80 in the works. Many things are possible at a lower cost than you might expect. Watch for us!

4139 N. 78th Court
Milwaukee WI 53222

Level II ROM has routines for assembly programmer

By ED FAULK

One of the most frustrating parts of learning a new assembler language is trying to figure out what has already been written in order to take advantage of it.

In this article (and probably several more) I will describe various routines that are available in Level II ROM which the assembler programmer can take advantage of.

Some of these routines are mentioned in the Editor/Assembler manual (one erroneously) and others have been gleaned from disassembly of Level II Basic.

Still others are my own code developed in the process of writing several programs for the TRS-80. In fact, many of the routines described in this article are available in the program HEXDMP, which is available from me for $12.95 and includes both object and source, as well as a source listing. (Ed's address is 2531 East Commonwealth Fullerton CA 92631).

DISPLAY

Among the first needs of a programmer is the ability to get data into and out of his program. Therefore, the first two routines are concerned with displaying data upon the TRS-80 screen and reading data from the keyboard. A routine to write a byte of data on the screen might look like this:

```
DISPL EQU  $
      PUSH DE      ;save the
      PUSH IY      ;  DE & IY regs
      CALL 33H     ;write routine in
                    ROM
      POP  DE      ;restore the
      POP  IY      ;  DE & IY regs
      RET          ;return to caller
```

This routine could be invoked with the following:

```
      LD   A,x     ;byte to be written
      CALL DISPL   ;write it
```

In this routine, the byte to be written on the screen is loaded into register A and the Displ routine is invoked. Displ in turn saves the DE and IY registers in the stack, and then calls the routine that Basic uses to display the character.

The character is displayed at the current cursor location (this pointer is kept at location 4020H). The cursor location is then incremented by 1 (unless the byte was a carriage return—ØDH). The routine then restores the DE and IY registers to the instruction following the CALL instruction.

KEYBOARD READ

Reading data from the keyboard is not quite as straightforward. To compound this problem the routine given in the Editor/Assembler manual is wrong (used as written, it will exceed the stack limits). The correct routine for reading data from the keyboard might look like the following:

```
READ EQU  $
     PUSH DE      ;save the
     PUSH IY      ;  DE & IY regs
     CALL 2BH     ;scan the keybd.
     POP  1Y      ;restore the
     POP  DE      ;  DE & IY regs
     CP   Ø       ;was a byte entered?
     JP   Z,READ  ;  no—keep scanning
     RET          ;return to caller
```

Again, to invoke this routine the following code could be used:

```
      CALL READ   ;call read routine
```

In this routine, the subroutine (Read) will retain control until a byte is entered from the keyboard. At that time the byte will be returned in the A register. A value of Ø indicates that no byte was entered during the keyboard scan (which takes less than 2 ms.).

Additionally, the manual does not indicate that the Break key will result in the value 1 in Register A. The byte that was entered at the keyboard will not be displayed unless the program echoes it. Therefore the call should probably consist of the following:

```
      CALL READ   ;scan the keybd.
      CALL DISPL  ;display the byte
                   entered
```

This does not allow for the fact that the byte entered might be used by the program as a control character of some sort and might not need to be displayed.

SIGNIFICANCE

These two routines might seem trivial in terms of what you can do using the Z-80 assembler language but they form a core of subroutines which can be gathered together in writing larger, more complex programs.

For example, a collection of routines might include routines to read and write data, data conversion, multiply and divide, printer routines and so on. These routines will then not need to be rewritten and can be saved and merged into a new assembler program, thus simplifying the development cycle and also shortening the test time required since these common routines will have already been tested.

Just a few more tips on programming style. Use the Equ (equate) pseudo-op to define a label rather than putting the label on a statement. This allows you to insert code immediately after the label without having to edit the labeled statement to remove it. Never use code of the following form:

```
      JR   $-n
```

since the insertion of code between the jump and the target of the jump will change the address that is actually being jumped to. Instead, use the following:

```
      JR   tag
```

where "tag" is an equated symbolic address.

Good luck with assembler language, and if you don't like my routines, feel free to not use them. I don't claim that they are the best, just that they work.

—Reprinted from Orange County TRS-80 Users' Group Newsletter, available by donation, $10 a year from user group, 24232 Tahoe ct., Laguna Niguel CA 92677.

Here's more Level II subroutines

I previously discussed routines to read data from the keyboard and display data on the screen. I also promised to provide routines to write to the printer and routines to read and write cassette tapes.

This first routine will take a byte of data and print it on the printer. This routine is similar to the one previously noted for the RMS-1S Line Printer Patch. In keeping with that format, I will present this (and all other routines) as a callable subroutine. This routine is:

```
PRINT EQU  $          ;
      PUSH AF         ;save all
      PUSH BC         ;necessary
      PUSH IX         ;registers
      LD   C,A        ;get byte in rg.C
      LD   IX,4025H   ;printer DCB
      CALL 58DH       ;prntr.drvr. rtn.
      POP  IX         ;restore all
      POP  BC         ;of the saved
      POP  AF         ;registers
      RET             ;return to callr
```

This routine first saves all registers used by the driver routine in ROM (AF, BC, and IX). It then loads the byte to be printed into Register C, loads the address of the line printer control block into the IX index register, and calls the driver routine.

Upon return from the driver, all saved registers are restored, and control returns to the invoking routine. Note that the byte to be printed is passed in register A, just like the screen and keyboard routines even though the driver needs the byte in register C.

CASSETTE ROUTINES

The next routines we will cover are used to access the cassette drives. These first routines will use the normal 500 baud recording density routines in ROM, later we will discuss ways to increase the recording density.

First, we need to have a little understanding of how the TRS-80 records data on the tape. The technique chosen by Radio Shack is very similar to that used in normal telecommunications. At the start of each tape is a synchronization byte which is used to establish a common reference point for the interval timing used on the tape. This is necessary if the tape is to be started and stopped between each record.

Once the timing reference has been established, the computer is then ready to record the actual data. This is accomplished in what seems, at first glance,

LEVEL II ROUTINES, Cont.:

to be a very wasteful manner. Each byte of data is preceded and followed by control bytes. In normal computer networks this type of system of control bytes is called SPACE and MARK bytes. It is normally used to indicate to the computer when a byte starts and ends.

In the normal ASCII method, for instance, there is a one byte start mark followed by a one byte stop character. This discussion is, of necessity, very brief. In fact, if there is no need to speed up the tape rates, it is unnecessary! In any event, here are two routines to read and write tapes, complete with checksum computation and checking.

```
TREADØ EQU  $           ;HL-INPUT STRING
       PUSH AF          ;SAVE WORK REGS
       PUSH IX          ;   IN STACK
       PUSH BC
       PUSH DE
       LD   DE,Ø        ;CLEAR  CHECKSUM
       CALL 212H        ;REG A DEFINES DRIVE
                         WITH Ø OR 1
       CALL 296H        ;FIND LEADER & SYNC
                         BYTE
;*
TREAD1 EQU  $
       CALL 235H        ;READ A BYTE OF DATA
       CP   255         ;END-OF-RECORD?
       JP   Z,TREAD2    ;     YES-GET CHECK-
                         SUM
       LD   (HL),A      ;SAVE THE BYTE
       INC  HL          ;BUMP POINTER
       LD   IX,CHKSUM   ;POINT TO CHECKSUM
                         AREA
       ADD  A,(IX)      ;CALCULATE CHECKSUM
       LD   (IX),A      ;SAVE IT
       JP   TREAD1      ;LOOP 'TIL DONE
;*
TREAD2 EQU  $
       CALL 235H        ;READ CHECKSUM BYTE
       LD   E,A         ;SAVE IT
       CALL 1F8H        ;TURN CASSETTE OFF
       CALL 22CH        ;FLASH RIGHT ASTER-
                         ISK
       LD   A,E         ;GET CHECKSUM
       LD   IX,CHKSUM   ;GET WORKING ADDRESS
       CP   (IX)        ;ARE THEY EQUAL?
       JP   Z,TREAD3    ;   YES-NORMAL RETURN
       LD   HL,(4020H)  ;SAVE CURSOR
                         POSITION
       LD   A,'C'       ;GET CHECKSUM ERROR
                         FLAG
       LD   (3C3EH),A   ;INDICATE CHECKSUM
                         ERROR
       LD   HL,4        ;SET RETURN CODE
       JP   TREAD4      ;GO POP REGS
;*
TREAD3 EQU  $
       LD   HL,Ø        ;SET GOOD RET-
                         URN CODE
;*
TREAD4 EQU  $
       POP  DE          ;RESTORE
       POP  BC          ;ALL OF THE
       POP  IX          ;  SAVED
       POP  AF          ;   REGISTERS
       RET              ;RETURN  TO  CALLER
```

This routine will read a tape and stop after ever read. It verifies the checksum and indicates a checksum error by displaying the letter C in place of the leftmost asterisk. It will flash the right asterisk in keeping with conventions established by Radio Shack. This routine will not read tapes written by either BASIC or the Editor/Assembler. It will only read tapes written by the accompanying routine.

This write routine requires that the HL register point to the record to be written, register B must contain the length (zero indicates that the record is 256 bytes long). The cassette to be used must be indicated in register A (a value of Ø or 1).

```
TRITEØ EQU  $
       CALL 212H        ;SET CASSETTE ADDRESS
       CALL 287H        ;WRITE LEADER & SYNC
                         BYTE
       LD   C,Ø         ;CLEAR CHECKSUM
TRITE1 EQU  $
       LD   A,(HL)      ;GET A BYTE
       ADD  A,C         ;CALCULATE CHECKSUM
       LD   C,A         ;SAVE CHECKSUM
       LD   A,(HL)      ;GET BYTE AGAIN
       INC  HL          ;BUMP POINTER
       CALL 264H        ;WRITE TO TAPE
       DJNZ TRITE1      ;LOOP 'TIL DONE
       LD   A,255       ;END-OF-RECORD MARK
       CALL 264H        ;WRITE IT
       LD   A,C         ;GET CHECKSUM
       CALL 264H        WRITE IT
       CALL 1F8H        ;TURN OFF CASSETTE
       RET              ;RETURN  TO  CALLER
```

NOTE: this and the accompanying routine are dependent upon the record being written not containing a hex "FF" (255) which is used as an end of record mark. If the record is to contain any possible value, the routines may be changed to read and write based solely on a length such as 80 bytes.

TAPE SPEEDUP

Since I said I would show some ways to increase the speed of the tape transfer, I will now pass on some tips that may be of some use. These two routines start and stop the tape for each record and are therefore forced to write the leader (blank tape) and synchronization byte for each record. If this could be eliminated the effective transfer rate would be increased even though the actual packing density is still 500 baud.

One way to accomplish this is to take the start and stop code out of these routines, and make them separate routines. Then, when several records, say four or five, are ready to be written, the sequence could be as follows:

1) Call the start/select routine.
2) Call the write routines for each record (checksums could be taken by record or by "block", where a "block" is a group of records between each start and stop).
3) Call the stop routine.

Using this technique, you could increase the number of records written in a given period of time, as well as increase the number of records written on a tape.

If this technique is used, some thought must be given to the "deblocking" routines that must be used since the block will contain more than one record. One technique that might be used is given below. This routine is called everytime a record is needed.

```
DEBLOC EQU  $
       PUSH DE          ;SAVE NEEDED REG-
                         ISTERS
       PUSH HL          ;FOR LATER RESTORES
       LD   HL,(BUFPTR) ;POINT TO NEXT REG
                         AREA
       LD   DE,ENDBUF   ;END OF BUFFER AD-
                         DRESS
       ADD  HL,BC       ;POINT TO NEXT BUF-
                         FER
       CR   A           ;CLEAR CARRY FLAG
       SBC  HL,DE       ;PAST END OF BUF-
                         FER?
       JP   M,DEREAD    ;YES*READ ANOTHER
                         BLOCK
       LD   (BUFPTR),HL ;SAVE POINTER
       RET              ;RETURN  RECORD
                         ADDRESS
;*
DEREAD EQU  $
       .
       .
       .
;*
BUFFER DEFS nnnn        ;LENGTH OF BUF-
                         FER
BUFPTR DEFW Ø           ;BUFFER POINTER
ENDBUF EQU  BUFPTR-1    ;LAST BYTE OF BUF-
                         ER
```

In this routine, the DEREAD code was not given. It simply calls the read routine that we modified to read an entire block of data after pointing HL to the start of the buffer.

When the read returns to it, we restore HL to the start of the buffer and return to the routine that called DEBLOC. It is assumed that BC contains the length of each record. If we can write variable length records we might consider writing the length of the record as the first byte of the record in order to keep track of it.

To keep the routine simple, I have assumed that all blocks are full blocks, that is, no block contains less than the number of records chosen to fit in a block. One way to do this on the last block is to write as many end-of-file records as needed to fill the block.

WRITE ROUTINE

Of course, to use this routine, we need a corresponding write routine. Rather than provide the code for this, I will simply discuss a technique to do this.

As we saw, the routine to read a block and deblock it was relatively simple. To perform the inverse all we need to do is get the record, put it in the buffer, check to see if the buffer is full and, if it is, write the block.

If there is still room in the buffer we must save the pointer to the next available location and return.

If short blocks are desired, we must be called before the program ends to write the last buffer to tape.

BASIC INTERFACE

Since this type of tape I/O could be beneficial to us even if we are using Basic, I will show a simple interface between Basic and our subroutines. Note that some changes to the routines will be required. To write a Basic record we could use this code:

In Basic:

```
POKE 16526,nn : POKE 16527,nn    'ENTRY POINT
X=USR (VARPTR(I$))               'PASS STRING
```

In subroutine

```
CALL ØA7FH       ;GET VARPTR IN HL
PUSH HL          ;SAVE IT
POP  IY          ;LOAD IN IY
LD   H,(IY+2)    ;STRING ADDRESS MSB
LD   L,(IY+1)    ;STRING ADDRESS LSB
LD   B,(IY)      ;GET STRING LENGTH
```

The rest of the code is the same as that using the normal write routine defined earlier. To read a record and pass it to Basic is more difficult. Using the simple technique we have the following:

In Basic:

```
I$=STRING$ (nn," ")     'BUILD DUMMY RECORD
X=USR (VARPTR(I$))      'PASS STRING ADDRESS
```

In subroutine

```
CALL ØA7FH       ;GET ADDRESS IN HL
PUSH HL          ;SAVE IT
POP  IY          ;RESTORE IT
CALL READEM      ;CALL  READ  ROUTINE  INTO
                  STRING
                 ;AREA IN SUBROUTINE
LD   HL,STRING   ;GET WORK ADDRESS
LD   (IY+2),H    ;SAVE ADDRESS MSB
LD   (IY+1),L    ;SAVE ADDRESS LSB
LD   (IY),B      ;RETURN LENGTH
JP   ØA9AH       ;RETURN TO BASIC
```

A better way to return the string is:

Lots of capability left in old TRS-80 Level I Basic

(Reprinted from Orange County TRS-80 Users' Group Newsletter)

(Level I hasn't been forgotten. Here James tells you how to get more mileage out of it. Ed.)

Let's begin with a few ways to shorten your programs.

1) Omit the space after each line number. The computer supplies a space when you List. If your program is 100 lines long, you will save 100 bytes.
2) Eliminate trailing quotes. If you type the following line: 100P. "Hello the computer knows what you mean.
3) Use the abbreviation "I." for INPUT (the manual says to use "IN."). The machine understands 100I."What is your name" ;A$ and never confuses Input with Int even when both are abbreviated "I." (It tells them apart by context).
4) When preceded by a line number, Cls (clear screen) may be abbreviated "C." (the manual does not list an abbreviation).
5) Consider the following
   100P. "landing velocity=";A. (V)
   where V is a (possibly negative) number which we wish to print without a minus sign. According to the manual, this is a perfectly correct line, however, it won't work (try it!). The computer does not interpret A. as the abbreviation for Abs, even though it is followed by parenthesis. Instead, it interprets A. as AT, because of the P. (Print) with which the line begins! First the bad news: Abs cannot be abbreviated A. in a Print statement. But the good news is that AT can be used anywhere, and any number of times in a single Print statement. And, unlike T. (Tab) AT will "back up" . For example, the following program uses this method (at line 30) to print a message from bot-to top.

```
5C.
10I. "name please" ;A$
20C.
30P. AT596, "Users Group"; AT468;
"TRS-80";AT340;"welcome to";AT212;
40P.A$;AT831
```

   Each such use of At without P. saves at least three bytes (colon, P, period).

'BLACKBOARD'

Most people don't use line 1 in their programs. This is always a good idea, and here's a new reason:

Without disturbing whatever program is currently in the machine, (that is, don't type new), enter the line 1G.1 and RUN. Hit clear and your TRS-80 is now a blackboard, capable of showing 1023 characters. Just type your message, correcting errors with , and skipping lines with Enter. Press clear at any time to start over, or type an answer to a previous message. To return to your original program hit: Break, 1, Enter, Run.

EDITING HINTS

Next time you L. (LIST) your program, press Clear before you press Enter (L., Clear, Enter). This gives a nice clean list of your program without any annoying scrolling.

If your program contains long lines (between 64 and 70 characters) you may lose some lines off the top of the screen. A List command will list twelve logical lines, that is, twelve line numbers and their "contents."

To recover these lost lines so that they may be examined and/or changed, you may insert short lines at unused locations (between the smallest and largest line numbers shown on the screen). Type line numbers, followed by spaces (then Enter). When enough of these "fillers" have been inserted, your top line will sheepishly return for examination.

These "fillers" may be used to set off sections of your program for visibility, saving two bytes over REM statements.)

Another handy "scroll-preventing" trick is:

List, Enter, backspace (←), Enter
or
List, Enter, ↓, ↓, ..., ↓, backspace (←), Enter as many as needed

This allows you to retype the last line, right underneath it or to retype the top line of the current List, without having to renumber it, or List again with a smaller line number.

—JAMES GARON

LEVEL II ROUTINES, Cont.:

```
LD    D,(IY+2)      ;GET STRING ADDRESS MSB
LD    E,(IY+1)      ;GET STRING ADDRESS LSB
LD    C,B           ;GET LENGTH IN C
LD    B,Ø           ;CLEAR B
LDIR                ;MOVE STRING
LD    HL,retcod     ;LOAD RETURN CODE
JP    ØA9AH         ;RETURN TO BASIC
```

I've covered a great deal here that just barely touches on the tape I/O routines. In another article I will discuss techniques for performing I/O at speed greater than 500 baud. This sort of thing is especially useful if you wish to send tapes without fear of their being read by anyone except the person for whom they're intended.

—Reprinted from Orange County TRS-80 Users' Group Newsletter.

(PHOTOCOPY THIS AND INSERT IN YOUR LEVEL II MANUAL)

---TRS-80 LEVEL II INDEX---

ABS	7/1	FIX	7/2	PRINT#-1	3/10
ASC	5/3	FOR..TO..STEP	4/8	PRINT	3/1
ATN	7/1			PRINT@	3/1
AUTO	2/1	FRE	5/5	RANDOM	7/3
CDBL	7/1	GOSUB	4/6	READ	3/9
CHR$	5/4	GOTO	4/5		
CINT	7/2	IF	4/12	RESET	8/2
CLEAR	2/2,4/3	INKEY$	5/5	RESTORE	3/10
CLOAD	2/2	INPUT#-1	3/10	RESUME	4/11
CLOAD?	2/2	INPUT	3/7	RETURN	4/6
CLS	8/2	INT	7/3	RIGHT$	5/7
CONT	2/3	LEFT$	5/6	RND	7/3
COS	7/2	LEN	5/6	SET	8/1
CSAVE	2/3	LET	4/4	SGN	7/4
CSNG	7/2	LIST	2/4	SIN	7/4
DATA	3/8	LLIST	10/2	SQR	7/4
DEFDBL	4/2	LOG	7/3	STOP	4/5
DEFINT	4/1	MEM	8/4	STR$	5/7
DEFSNG	4/2	MID$	5/6	STRING$	5/7
DEFSTR	4/2	NEW	2/4	SYSTEM	2/5
DELETE	2/3	NEXT	4/8	TAB	5/2
DIM	4/3,6/3	ON ERROR GOTO	4/11	TAN	7/4
EDIT	2/4	ON N GOSUB	4/7	THEN	4/13
ELSE	4/13	ON N GOTO	4/6	TROFF	2/5
END	4/4	OUT	8/4	TRON	2/5
ERL	8/2	PEEK	8/5	USING	3/3
ERR/2+1	8/3	POINT	8/2	USR	8/7
ERROR	4/10	POKE	8/5	VAL	5/8
EXP	7/2	POS	8/6	VARPTR	8/8

—Reprinted from Orange County TRS-80 Users' Group Newsletter, available by donation, $10 a year from user group, 24232 Tahoe ct., Laguna Niguel CA 92677.

Here's a sample chapter from Barden's new book, Disk Interfacing Guide

By WILLIAM BARDEN

This text describes the operation of the Shugart SA400 Minifloppy Disk Drive in the Radio Shack TRS-80 Microcomputer System. It is divided into five chapters.

The first chapter, Disk Basics, describes the general operation of minifloppy disks. Chapter 2, Shugart SA400 operation describes the operation of the disk drive itself in terms of interface signals and functions.

The next chapter is concerned with operation of the Western Digital FD1771B-01 Floppy Disk Formatter/Controller Chip used in the TRS-80 Expansion Interface.

Chapter 4 shows how the Expansion Interface decodes disk addressing and commands.

The last chapter shows how Radio Shack software communicates to the disk and how one may do machine language (assembly language) and limited Basic-level programming of disk systems. Appendices provide related material, such as controller commands operation sequence.

A floppy disk system is made up of the disk drive or drives themselves, a controller, and the microcomputer. In our case the microcomputer is the Radio Shack TRS-80, the controller is the Western Digital FD1771B-01, and the disk drive is the Shugart SA400. A block diagram of the TRS-80 disk system is shown in figure 1. As with other units in the TRS-80 system, the cpu communicates over 16 address lines, A15 through A0, eight data lines, D7 through D0, and a set of control lines that specify whether reading or writing and other functions are being performed.

The controller for the disk(s) interprets commands sent to it over the data lines and translates these commands into disk-type commands that the Shugart SA-400 can recognize.

The single chip controller is a 40-pin chip that is effectively a microcomputer in itself, and replaces a hundred chips or so for a TTL design!

Commands are sent to the disk drive by the controller chip to perform functions for head positioning and reading and writing, and the "status" of the drive is returned back to the controller chip.

We'll be talking about the operation of each of these component parts in future chapters, but for the time being let's concern ourselves with how the data is stored on the "diskette" and some of the physical attributes of the diskette and disk drive.

'DISK', DISKETTE

When the disk drive is mentioned in this text, we'll refer to the "disk" or "drive" or "disk drive". When the recording media is mentioned, the term "diskette" will be used.

The diskette used in the Shugart SA-400 is basically a 5 1/4 inch diameter flexible or "floppy" diskette made up of a plastic coated with a magnetic oxide similar to that used for recording tapes. The diskette fits in a square holder for protection and ease of storage.

The square holder fits inside the SA400, which is really only a device that spins the diskette and moves a recording head along a radius while the disk is spinning, along with associated electronics to read and write data.

The recording head reads flux changes or produces flux changes for writing, similar to a tape recording head. Other disk electronics control head positioning, protection of the diskette from writes, and other functions.

The diskette spins at 300 revolutions per minute. As the diskette revolves, the head can be moved along a radius towards the center or back again in small increments.

Each discrete position over the diskette defines a "track" as shown in figure 2. There are 35 tracks for a Shugart SA400, and therefore 35 valid positions along the radius.

When the head is positioned along the radius over a track it can read the data along the concentric circle defined by the track. This circle is divided into ten "sectors" each occupying 1/10 of the circumference of the track. The circumference of the innermost circles or tracks are obviously less than the outermost tracks, but the content of the tracks are the same, although the data is packed a little more tightly into the innermost tracks.

Once positioned over a track, sector 0 can be sensed by a small sector index hole in the diskette which causes a signal to be generated by the disk when the index hole passes five times per second.

Each sector on the disk holds 256 data bytes. The entire track can hold 256*10 bytes, or 2560 bytes of data. As there are 35 tracks, the entire diskette can hold 89,600 bytes of data.

This data is recorded in serial fashion along the track, so that one track holds 2560*8 bits of data, or 20,480 bits along the circumference. Data recorded along every track, then, can be viewed as a long string.

—From TRS-80 Disk Interfacing Guide, 46 pages, $3.75: Micro applications, 2432 Tahoe ct., Laguna Niguel CA 92677.

'Basic Handbook' now covers more

David Lien, author of our Level I manual, reports that sales of "The BASIC Handbook" are booming. Computer stores both in the U.S. and overseas, along with the many readers of TRS-80 Computing who clipped the discount coupon in the last issue, are waiting for the ink to dry.

Instead of covering BASIC as used in 25 computers, as originally planned, The Handbook grew to include over 50. (Dr. Lien tells us the number is actually closer to 80). He didn't know when to stop!

It is done now and is about 350 pages long. CompuSoft says the presses are rolling and deliveries start November 1.

Because so many TRS-80 owners who upgraded from Level I to Level II couldn't make heads or tails out of the instruction booklet that came with the conversion tape, Dave wrote his own version of what the booklet should look like.

CompuSoft has agreed, as another reader discount, to include a free copy of "Converting Level I tapes to Level II---the Easy Way", plus sending "Handbook" orders by UPS instead of postal book rate.

The "Basic Handbook" should be essential to anyone wishing to convert "standard" Basic to TRS-80. See discount coupon page for special offer.

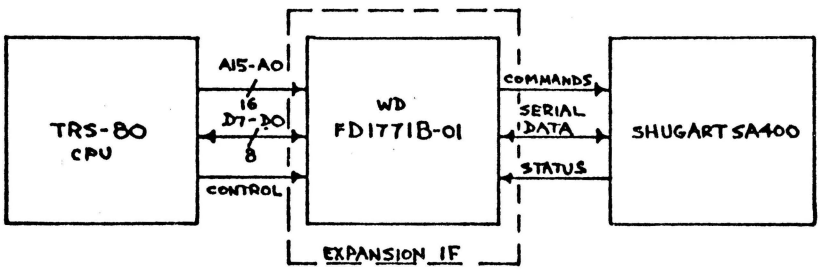


Figure 1, TRS-80 disk.

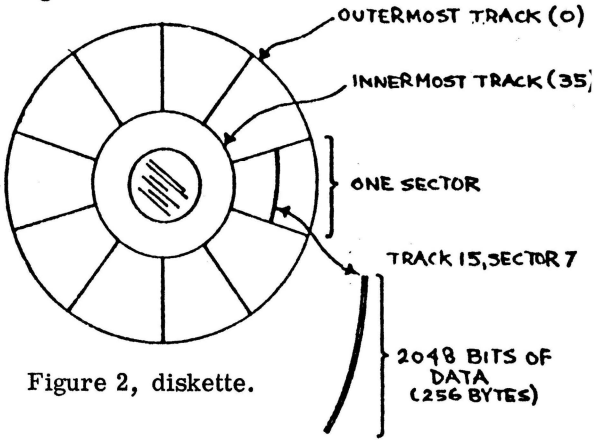


Figure 2, diskette.

Some inelegant solutions improve accuracy

By ROBERT M. RICHARDSON
W4UCH/2, Drawer 1065
Chautauqua Lake, N.Y. 14722

My TRS-80, though a dear friend and true joy to behold and use, is a fairly early model having been ordered during September 1977, and also one of the first to have Level II basic installed, last winter. It is serial no. 010689.

My TRS-80 has two modest glitches that are easily overcome which I would like to share with those that may have experienced the same two problems.

The first problem involves the FOR-NEXT-STEP loop when using small incremental STEP numbers such and .1, .01, or .001, that at times "blows the computer's logic" by picking up or losing a stray digit, whether running in single or double precision.

Try this test to see if your TRS-80 has this problem: 10 FOR X=1 TO 100 STEP .01 : PRINT CDBL (X) : NEXT X If your TRS-80 runs this simple program without one glitch it is healthy and well! Skip the solution and go on to the next test. One solution is to use your INT function, but that is too elegant a solution. A much simpler solution is as follows: 10 FOR X=1 TO 10000 : PRINT CDBL (X) /100;: NEXT X Most TRS-80s will print out this simple program without a flaw. If not, back to the Radio Shack Doctor.

This problem of picking up or losing a stray bit at the end of either a single or double precision operation also occurs at times on my TRS-80 when using the ↑ exponent function. Try this program to see if yours suffers the same malady:
10 FOR X = 1 TO 1000 : PRINT CDBL(X) ↑ 2;: NEXT X

If you have any fractional numbers printed out, you too have a problem. The problem is most like to occur with numbers below 100,000 (print-out), so watch closely. The solution is absurdly simple. Merely multiply X times X the number of times you were using the exponent sign for. 10 FOR X=1 TO 1000 : PRINT CDBL (X*X);: NEXT X

This exponent sign problem has driven at least one TRS-80 user up the proverbial wall. Though it will manifest itself with only a few numbers, Murphy's Law states: "if it can happen, it will." It is especially annoying when this exponent sign problem occurs when using the IF - THEN conditional statement. It will usually happen deep in the heart of your program where an IF - THEN statement is using the = sign for comparing two values. You and I know that the Pythagorean Theorem has been true for over two thousand years. With only a modest assist from us, our TRS-80 will become a believer too!

Here's Level II fix for blackjack tape

I've had some trouble with my Level II blackjack program from Radio Shack. The symptom was that when the dealer would bust he would sometimes win even if neither of the players busted.

The problem seems to lie in the Else statements in line 2670. There are two Else If statements on that line preceeded by an If.

The problem is that if the first If statement is true the two Else If statements are ignored. One of the Else If statements sets the dealer's score to 1 if his actual total score is greater than 21 which (correctly)makes him lose. However, if the first If statement is true the dealer's score will never get checked for busting.

To fix the program add the first If clause to the end of line 2660 and the remainder of line 2670 stays as is except for removal of the first Else.

—J.P. RAHN, 1019 Randall
Ridgecrest CA 935

Contacts too wide on 'bouncy' keys

In regard to the keyboard and the double characters referred to by Le Roy Cooper, the local shop fixed mine and passed on the "secret".

The problem stems from keyboard construction when the contacts are too far apart. Bending the contacts closer together solves the problem.

This is done from outside the sealed case. Merely make a key puller to pop off the key covers.

A paper clip with a hook bent into it is what I use. Put the puller hook under the key and just lift it off.

Once the cover is off you'll see two copper contacts: the one will be solid the other has about 4 small fingers.

With a small screwdriver bend the solid contact toward the fingered one. This is difficult because when you start to push on the contact the key starts to depress leaving you nothing to bend against. I solved this by clamping the side of the exposed key with a hemostat to keep it from depressing as I applied pressure to the contact.

This is all it takes to fix it. Repeat for each "bouncy" key and start enjoying your computer again.

—BILL ROBERTS, 604 Cardamon Dr.
VA Beach VA 23462

Alternatives exist to 'screen printer'

This writer recently unloaded his screen printer at a reasonable price.

The screen printer is not extraordinarily expensive for a computer peripheral and may have been unjustly maligned.

It is fast, and on occasion has been known to print evenly. The alternative to the RS screen printer is the more expensive Centronics, which is a full-fledged computer printer with many of the niceties, including ordinary paper.

For those of us that cannot afford the $1300 for this model, however, there is a further alternative.

Centronics makes a microprinter called the P1 that will interface to the TRS-80 expansion interface unit. I've ordered one after being reassured by Centronics that the unit is compatible with the plug on the expansion interface. For further information, call Centronics.

The P1 prints at 180 lines per minute, uses a 5 by 8 dot matrix, and the aluminized paper comes out from the top rather than the side, as in the screen printer case! You will need the expansion interface, however, at $300.

—WILLLIAM BARDEN, editor, Orange County TRS-80 Newsletter, $10 (yr) donation to Users' Group, 24232 Tahoe ct., Laguna Niguel CA 92677.

Wants circuit for printer interface

Maybe someone out there in TRS-80 computerland can help me. I'm looking for an " el cheapo " printer adapter that will plug directly into my computer. What would be nice, would be a diagram, with instruction, so I could assemble it myself. So far the only practical thing available is a system in a fancy box that goes for $129.00. I don't need anything fancy, just cheap. It's hard enough for the average Joe Blow to scrape up $800.00 to $1800.00 for a printer. I've studied all the available data on a bunch of printers and have come to the conclusion that the Intragul Data IP-125 printer is far and away the best all around value, at $795.00.

If you could run an article on the above, you will have a devoted fan forever!

It wouldn't hurt if you could come up with an " el cheapo " do-it yourself circuit for a TRS-80 to S-100 bus adapter, also. I'll bet the average TRS-80 owner, like me, does not like the idea of being locked into the TRS products.

—J.R. "BOB" MENZIES
1011 Neal Drive
Alexandria VA 22308

(Check out to Electronic Systems' Serial/RS 232 kit, at about $75; address, Box 21638, San Jose CA 95151. ED)

It is not hard to speed up TRS-80's slow ticker

By STEVE GIBSON

(NOTE: in TRS-80 Bulletin 1:3, this story appeared with a wrong IC number in step 6)

TRS-80 does not operate its powerful Z80 processor at full speed, but a speed-up is easily accomplished.

The TRS-80 clock crystal frequency is about 10.64 mc. IC Z56 (see 1:1 edition schematics, p. 19) divides this frequency by 6 to get 1.77 mc which feeds the Z80 chip. All we do is divide the crystal frequency by 4 instead to get 2.66mc and install a switch to allow you to choose which clock frequency you want. Warp 1 or Warp 2!

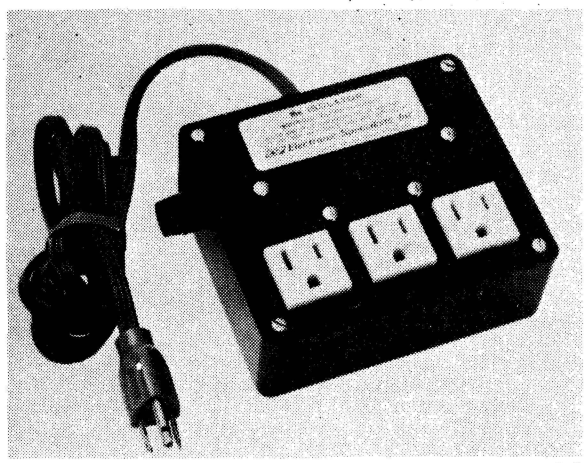
This is a very easy mod...really.

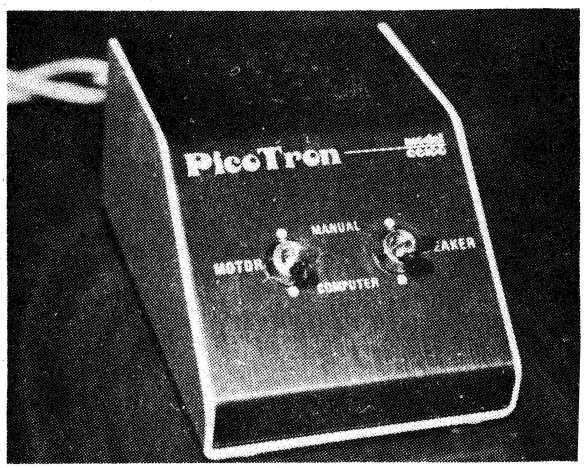
1) Cut the trace leading from pin 8 of IC Z56 to the one and only solder hole next to it.
2) Install a small SPDT switch on the computer bottom plate. Be sure that the location you choose won't cause the switch to touch other parts when you put it all back together.
3) Run a wire from the movable arm of the switch to the solder hole. This point leads to the Z80 clock input.
4) Run another wire from either side of the switch to pin 8 of IC Z56. This is the 1.77 mc output.
5) Run still another wire from the other lug on the switch to pin 12 of IC Z56. This will be the 2.66 mc output.
6) Last...run a short (2 inch) wire from IC Z43 pin 2 to pin 14 of IC Z56. You're done!

What you just did was steal a divide by 2 signal from the clock off of Z43, then divide it again by 2 with an unused flip-flop on Z56 resulting in division by 4. Voila! Despite the fact "you can't run clock signals all over the place" this little mod does work.

One "Gotcha" though with this circuit. Old cassette files and disk won't work. Now you know why you need that switch.

—Gibson Engineering,
547 N. Beachwood dr.,
Los Angeles CA 90004.





Electronic Specialists' power line interference isolator is seen at left, and the Picotron cassette controller is at right.

Power-line glitches cause disk woes

Radio Shack tells TRS-80 Computing that power-line hash is causing severe computer reliability problems.

At the factory, an experiment to determine diskette durability crashed after a disk was reproduced only eight times.

Power-line glitches were suspected, and the technician plugged in a $4-or-so TVI filter (from Radio Shack). The experiment then continued for 80,000 disk reproductions.

A more sophisticated device is available from Electronic Specialists. It provides three individually filtered three-prong AC sockets with integral surge suppression.

Since each socket is isolated from the other sockets, equipment interactions are eliminated, yielding error-free and glitch-free operation.

Isolator is said to be particularly valuable for isolating the MicroProcessor form peripherals as well as isolating disruptive line hash and damaging power line surges.

Connecting to the 120 VAC Line with a standard three-prong plug and 15 amp fuse, Isolator can accommodate an 1875 watt total load, with each socket capable of handling a 1000 watt load.

Price of the device is $50 from Electronic Specialists, Inc., 171 South Main St., Natick MA 01760 (617) 655-1532.

Firm sells Qume, Diablo printers

Computer textile of Los Angeles is offering reconditioned daisywheel printers that hook to TRS-80, allowing high quality word processing and plotting. All that is required to put a daisywheel printer "on line" with the TRS-80 is the TRS-232 interface developed by Small Systems Software, or some other RS232 circuit.

The TRS-232 interface is a software driven transmit-only device that plugs directly into the TRS-80 CPU box or the TRS-80 expansion interface. It gives TRS-80 owners the power to have beautiful low cost hard copy using a reconditioned daisywheel printer.

The TRS-232 interface sells for only $40 as is and may be purchased from Computer Textile, Electric Pencil word processing program for TRS-80, at $100.

Computer Textile sells reconditioned Diablo and Qume based printing terminals.

The Diablo-based terminal has the following features: HyType 1 printer mechanism, 82 key keyboard with numeric pad, RS232 interface with cable, ASCII encoded, 128 character buffer, switch selectable 10 or 12 pitch, complete graphics capability including 1/60th inch horizontal carriage control and 1/48th inch vertical carriage control. The printer is enclosed in a beige cabinet and sits on a black base with a chrome wheeled

The Qume-based terminal is simila but uses the Qume 30 printer mechanism The Qume has extended graphics cap ability in that it has a firmware pack age called " super-plot " which allow 1/120th inch horizontal carriage control The Qume has a 228 character buffe and a few additional operator controls The Diablo terminal sells for $1995 an the Qume is $2150.

Forms tractors and pin feed platen are options available with either printer Both units come with a limited 30 da warranty.

Computer Textile is at 10960 Wilshir blvd., Suite 1504, Los Angeles, CA 9002 telephone (213) 477-2196.

Picotron controls cassettes

Picotron claims its cassette controller eliminates TRS-80 "pull plug-push plug" cassette-to-computer loading.

The Model CC100 controls cassette motor functions, monitors tape location with its internal speaker and requires no power.

Speaker volume is reduced when in data search mode to allow easy listening. Normal volume is maintained when in data load mode. "Saving" data on tape can also be monitored on most recorders.

The $30 unit is ready to plug in and use. Picotron's address is: Box 62076 Sunnyvale CA 94088; (415) 964-4175

Replaces relay in TRS-80, and ends cassette recorder hangups

By DEAN W. CHANDLER
President, Mass Computer Club

From the first day I typed the word "CLOAD", my cassette port on/off relay has had a tendency to stick. Now since I am running Level II, the problem has become more prevalent. I suspected this increased sticking to be caused by several attributes of operating and equipment design;

1) When Level II is used with the expansion interface, the relay "beeps" whenever the NMI reset button is pushed. This must surely cause a slight arcing at the relay contacts.
2) In my area, I was unable to obtain the recommended CTR-41 for use as a second recorder, so I purchased a CTR-21A and pressed it into service. I found that the current to be switched in the remote socket was higher. Bad news in an already shakey area. (By the way, the bias had to be adjusted to get the same level of volume on tape from either recorder.)

In my search to solve this sticky problem, I tried everything from degaussing the relay (this worked for a time) to bypassing the relay contacts with a small (3uf. 15v.) capacitor, but alas with increased use came increased sticking.

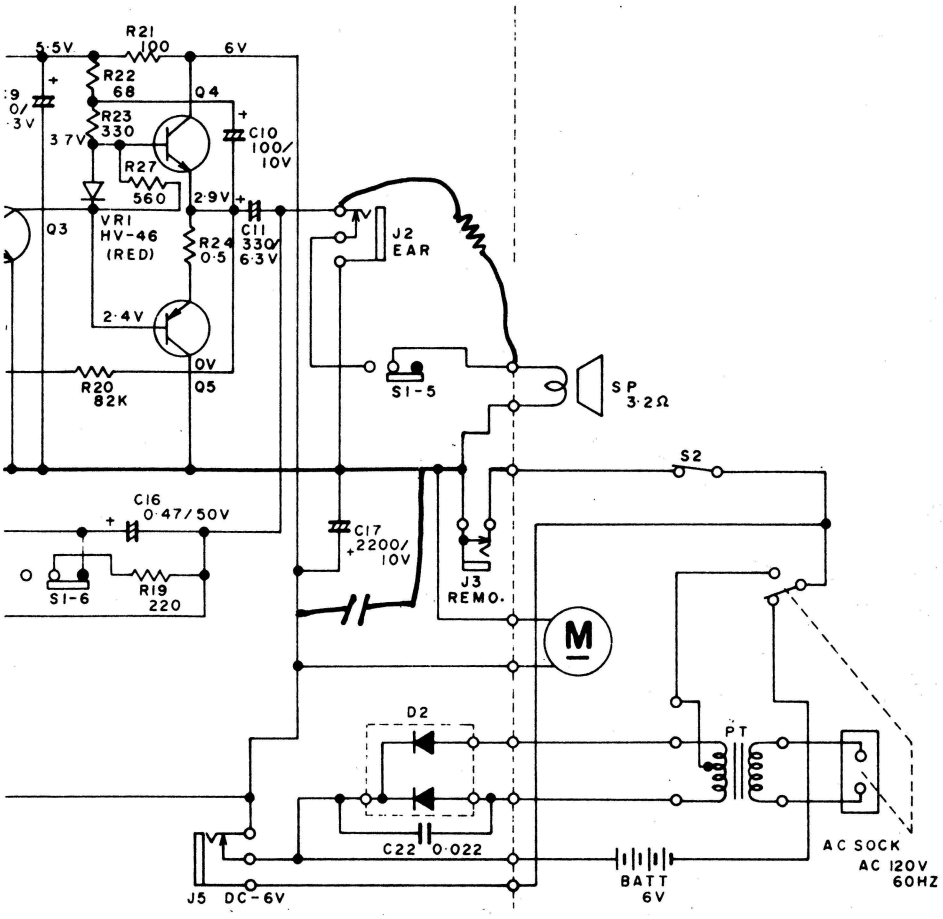
The Solution!!!

Thoroughly irratated, I carefully desoldered and removed the offending relay, purchased a Radio Shack #275-004 sensitive relay ($2.99). Because of it's larger size, I placed it on the foil side of the p.c. and lying on its side and stuck down with a small dab of silicone rubber (Dow "RTV" or equivalent).

I insulated the leads from the relay with shrink tubing. I believe I have finally eliminated the relay problem, and in the bargain, I have a relay that can be controlled by software to operate anything up to 1 amp 125 volts AC or DC.

—134 Breckwood Circle
Springfield MA 01119

(If anyone wants a physically-smaller relay that will do the same thing as the Radio Shack sensitive relay, and doesn't mind putting out the extra bucks, Dan Likins has one for $8. Like the Radio Shack, the terminals do not fit the PC board holes. Dan has an ad on the next page. ED)



Add resistor, hear Csave, Cload; 'cap' improves Level II cassette

CASSETTE MODS—Steve "Mickey Modem" Gibson recommends connecting the 30-90 Ohm resistor from the top speaker terminal to the top ear-jack terminal. This, like the mod diagrammed in TRS-80 Computing 1:1 (p14), allows you to hear a Cload. However, it also lets you hear a Csave. Level II users bothered by high hum levels and poor cassette operation might parallel C17 with another 4K uF, 10 or 12V electrolytic capacitor. Physical size is the limiting factor: you've got to get something small enough that you can close the case.

'Bulletin' is biggest ad buy

Biggest ad bargain in microcomputing is the CIE TRS-80 Bulletin.

Full page is just $295, half page $175, quarter just $90, payable in advance of press time.

Issue 1:4 closes December 1st and bears a January publication date. Write for details.

Here's checklist of cassette faults

The following are some simple things to try when there are problems loading programs from cassette to the TRS-80.

1) Clean heads in the recorder.
2) Demagnetize heads.
3) Break shield connection going to "EAR" by using a short single conductor cable from a 1/8" miniature jack to 1/8" miniature plug (Radio Shack part 274-283).

Item three is a method of solving the ground loop problem without working on the inside of the cassette recorder.

—BOB SCHUETZ, 3327 E. Larkspur Dr.
Phoenix AZ 85032

You can get CTR-41 auto-level on 'play'

This modification used with the ground Loop Modification (TRS-80 Computing 1:1) will cure almost all Level II loading problems.

On CTR-41 short first two used pins of Record/Play Switch, together this will cause the Auto Level Circuit to work in Play mode. Then all programs Level I and II can be loaded at volume setting seven.

This works excellent with Level-I-to-II conversions.

—ROBERT A. CAHILL
RFD 1 Box 31
Kingston NH 03848

Cassette: here's how you can use it without shedding tears

By JOHN STRONG
Author Copyright 1978
3733 Mt. Almagosa pl., San Diego 92111

(Unless things get completely out of hand, this will be the first part of a two-part article. The second part will appear in the next issue.)

The vast majority of TRS-80 users rely on the cassette tape recorder for long-term storage of their programs or data. But you don't have to do it that way. If your rich uncle just died, the one that liked you anyway, you can buy a mini-disk system. Floppies are a computerist's nirvana but definitely not available on a tight budget.

A few strange TRS-80 types insist on typing in each program as they use it—a very poor way to develop a meaningful relationship with a personal computer! Such people are either in their first few hours of TRS-80 use (having not yet reached Chapter 9 in Dr. Lien's famous best seller) or are frustrated by problems in trying to get that %*=!&#! cassette recorder to work properly.

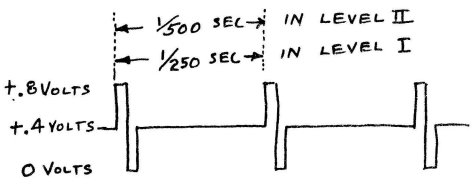
The cassette recording system was chosen by Radio Shack (and others) because of its refreshingly low cost, equipment-wise and especially tape-wise. Without it the TRS-80 line would have a big hole in the middle, leaving a super-elegant programmable calculator at one end and a multi-kilo-buck disk system at the other end. Cassettes provide a middle ground in both cost and storage convenience.

Cost advantages tend to be ephemeral. If the system designer doesn't exercise heroic self restraint, system costs skyrocket as the designer adds 'just one more' little special feature after another. In the case of cassette recorders used for personal computers, the cost advantage of cheap mass-produced standard audio cassette recorders requires that users live with a few inconveniences. Into each sun-drenched hobby a little rain must fall.

This discussion of TRS-80 cassette techniques is intended to help the user by informing him about common problems and how to avoid them. We aim to keep glitch-frustration down and up-time up. First let's discuss how data is recorded on tape.

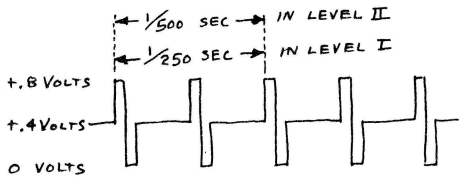
CASSETTE TAPE BYTE FORMAT

Program or data information is recorded by the cassette as a series of bi-directional rectangular pulses. The computer output is shown below for a series of 0 bits being recorded on tape:



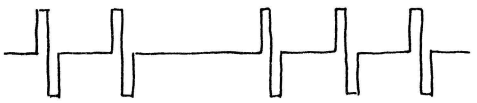
Series of Ø Pulses

The system sends 1 bits by adding another pulse between the 0 pulses:



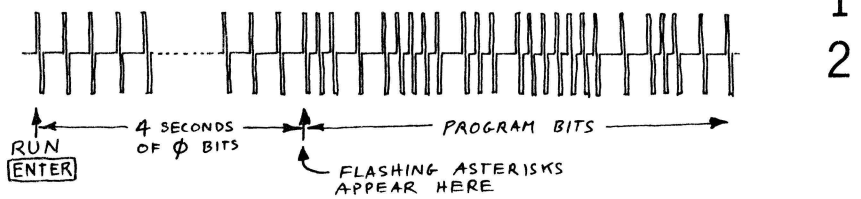
Series of 1 Pulses

Pulses for 0 bits and 1 bits are mixed in actual programs, as follows:



Series of Mixed Pulses: 1,Ø,1,1,Ø

Before sending a program to the cassette to be recorded (CSAVE), the computer proceeds it with four seconds of 0 bits:



This is necessary so that the automatic gain controls in the tape recorder will have time to settle down before the program pulses start arriving. If the recorder is not recording properly at the start of the program, some bits may be lost immediately and the program will not Cload successfully. The first few bytes in the program as recorded on the tape are control characters that must be recorded properly or the program can never be loaded successfully.

PULSE WAVEFORM DISTORTION

The waveform of the computer output becomes distorted in the recording and playback process, all of which has been taken into account in the design of the computer. The "fundamental" frequency of the computer output for a string of 0 bits is 500 Hz in Level II (250 Hz in Level I). Because of the duty cycle and waveforms of these signals they require enough bandwidth to transmit a square wave (50% on — 50% off) of 3500 Hz. This is much more difficult than transmitting a sine wave of 3500 Hz. To record pulse waveforms faithfully one would have to use a recorder with a frequency response flat to around 35,000 Hz. With a frequency response flat to 8000 Hz, the CTR-41 audio cassette produces somewhat distorted wave forms.

Fortunately, although distorted, these wave forms work very well indeed with the TRS-80. To record pulses more accurately is not possible in a recorder designed for audio applications. In the Floppy Disk System—designed only for digital signals—waveforms are better and the system works very well, at a rate of 100,000 bits/sec (or 12,500 bytes per second).

FLASHING '**'s

During Cload, as soon as the fou second train of 0 bits is completed, tw *,s will appear in the upper right corne of the screen as the program bits sta being received by the computer. Immed ately thereafter the right * will start flashing—once for each line in the program being Cloaded. In Level II the flashes occur at a rate depending on the leng of the program line. For a very sho line (10 x = 23) the rate would be flashes/sec (3 flashes/sec in Level

For a 64 character line which fil completely one line on the screen, the rate is close to one flash per second (1/2 flash/sec in Level I). for the longe possible program line, 255 characters 4 lines on the screen, the rate is one flash per 4 seconds (1 flash/8 sec in Level I).

So, by watching the flash rate one can get a good idea of how the Cload is pr ceeding. If the flashes are faster th one every second or two, fine--all's w (unless you are addicted to extremely long program lines). If the * stops flashing completely or flashes very slowly, you are in trouble. Drop-out of a single bit or character can cause it. So can too a playback volume level—or too hig volume level—or dirt on the cassette tape or head. Almost inevitably you will not get a READY>_ and you'll have to out of Cload by pushing the reset bu before trying again.

FILE NAMES

In Level II, each program must be signed a file name as it is Csave tape. The proper syntax is Csave file name". If you do not specify a file name in Csave, a MO (missing operand) e message will occur. The file nam optional in Cload or in Cload? bu used, the syntax is Cload "file name or Cload? "file name". Only number

CASSETTE UNPAIN, Cont.:

lower case letters should ordinarily be used for file names and usually they are only one character long.

All symbols and uppercase letters will cause problems of one sort or another if used as file names. They will produce an error message, cause a BAD message after Cload? or in some manner fail to operate properly with at least one of the three commands Csave, Cload or Cload?.

FILE NAMES WITH CLOAD

The file name feature permits one to specify a particular file on a cassette tape, such as Cload "D", and the computer will ignore any programs it comes to which have other file names. When it finally reaches the one specified, "D" in this case, it will load that one into the computer and stop. A very handy feature. And to further help the intrepid computerist, the file name of a rejected file is displayed in the upper right corner of the screen (where the first asterisk usually appears) while the cassette is running through the unwanted file. The second asterisk appears but does not flash. When the desired program finally comes up on the tape, the left asterisk finally appears (wiping out the previously displayed unwanted file name) along with the flashing right asterisk to indicate that the desired program is being properly recorded.

FILE NAMES—MULTI-CHARACTER

Although one letter is all that will ever appear in the upper right corner of the screen, much longer file names (at least 9 characters—sometimes more) can be used. Unfortunately, the computer only uses the first character to descriminate between programs. Csave "art" and you'll get it back into the computer with Cload "andy". The first letter is all the computer cares about. And it had better be lowercase too.

UPPERCASE FILE NAMES

If you Csave "f" using a lowercase "f" and then use either the command Cload "f" or Cload? "f" using an uppercase "F", you've got trouble. The left asterisk will be replaced by the letter F, no Ready message appears and Cload goes on forever. You'll have to press the reset button to get it out of Cload! Aggravating. It acts like you have tried to Cload a program with a completely different file name—which is really what you have done. The computer considers uppercase (shifted) "F" a completely different letter than the lowercase "f" so it refuses to load a lowercase "f" program when you ask for an uppercase F program. They may look the same on the screen, but inside the computer they are very different.

Another way to get into trouble is to Csave "F" using an uppercase F and then Cload "f" or Cload? "f" using a lowercase "f". This time the left asterisk is replaced by an ! and again, Cload goes on forever and no asterisk appears. This time there is a solution of sorts. Type in a Cload "F" or Cload? "F" using an uppercase F and every thing works fine. So you can use an uppercase letter for a file name after all—at least if you are willing to put up with a little confusion in Cload? and Cload. None the less, the best procedure is always avoid using uppercase letters in file names.

CLOAD ERASES OLD PROGRAM

The first thing that happens when a new program is Cloaded is that the old program is completely erased. The volume setting on the tape recorder may not be high enough to Cload the new program properly but if it is high enough for those double asterisks to appear, the old program is immediately gone with the wind. Which can give rise to the following sad scenario:

After 8 hours of typing and debugging and sweat your wonderful new program works like a dream! To preserve it for posterity you Csave and then verify the Csave was OK by Cloading it again. ZOUNDS! Only then do you discover the Csave was a huge failure and that in the process of trying to Cload you erased the original program in memory. Maybe you forgot to press the red Record on the recorder during Csave. Maybe the volume setting was too low during Cload. But no matter how it happened, the result is eight hours of time and creativity down the drain. There is an important lesson there some place! Do you suppose you should have used Cload? to verify that the Csave was OK? Of course you should have, dummy! If you aren't routinely using Cload? after every Csave you are shooting craps with destiny. I do it all the time and lose frequently. I'll never learn!

On the other hand, if you Cloaded and the double asterisks didn't appear at all (playback volume too low? or MUCH too high? or tape completely blank?) you came out smelling like a rose. The old program in the computer was not destroyed. Just Csave the program again, using Cload? to verify it and Cload to load it.

'LIST' VERIFYING OF CLOAD

If you have just turned on the computer and Cloaded a tape you want to use, what do you do next? Run it or Cload? it? The need to Cload? is not so great here. If the CLOAD was goofed up it is no big deal. You still have the program on cassette and you can always Cload it again if you have to.

The surefire way to check your Cload is to use Cload?. If you get a READY message you're in clover. If you get a BAD message, Csave it again and then try Cload? once more. This does take an irritating amount of time though.

If the program is long, and you're in a hurry and fairly sure there is no problem, there is a quicker way to check it than Cload?. Just LIST it. Most bad loads are quickly apparent that way. If the screen doesn't fill with gibberish before the end of the program, it is probably OK. Not absolutely, just probably. The danger is that one or more bits or bytes may be recorded wrong, enough to mess up a Run but not enough to be noticeable as the program zooms across the face of the scope. Quickest way to find such errors (which seem to occur about 1 or 2 per cent of the time) is Run the program. Even then you can't be 100 per cent sure. There may be an error in a branch or program line that your test Run didn't use. In the ultimate case (which a person doesn't see too often these days) you would have to devise a test run or test program which uses every single line in the program. A nice idea but very time consuming and seldom used. Just Run it and if the answer looks OK, hope for the best.

CAUSES OF MISSING BITS

More should be said about "missing a bit". It is really a very important and dangerous phenomenon. First, let's consider how it can occur in Csave:

(a) One way is by a dirt particle on the surface of the tape. That isn't too far fetched. In Level II, bits are transmitted at a rate of 500 baud or 500 bits per second. One seventh of the total time used to transmit a "Ø" bit is used for the bidirectional pulse. (See diagrams.) The rest is used for waiting. If "1"s are transmitted, the waiting time is shorter because of the second bidirectional pulse. Thus one pulse lasts 1/7 of 1/500 sec or 1/3500 sec. In a cassette the tape is moving at 1 7/8" per sec so one pulse occupies 1.875 * 1/3500 or .00054". That's small! No wonder dust is such a problem. So keep those cassettes in boxes and out of the nasty night air when they aren't being used.

(b) Another way that you can "miss a bit" occurs when there is a small defect on the tape due to a hole or void in the magnetic coating, usually invisible to the naked eye. Experts call this revolting development a "dropout" and make a big effort to avoid it. They don't always succeed.

(c) Transients on the 117V line caused by turning on a heavy-drain load (such as a soldering iron) near the computer might be expected to goof up a Csave—but they don't seem to. I have tried it several times, filling the screen with dots every time I turned on my soldering iron but the program I was Csaving was perfect. Transients on the line don't seem to cause bad Cloads either although they are reported to goof up mini-disk systems.

(d) Tape recorder gain fluctuations lasting several seconds or more are caused by "weaving" or sideways wandering back and forth of the tape over the recorder playback head. On a particularly severe low gain or high gain excursion, bits may be lost. Later a method is described for quickly detecting such problems—otherwise you are at the mercy of the tape recorder. Only Cload? can reliably protect you from a disaster due to weaving. In Tim Hensler's article the paragraph LEVEL II TAPE PROBLEM mentions a tape weaving problem and then he describes what may be the only possible way to Cload an important program from a weaving cassette. (Hensler, Tim; "For so long, repairmen...."; TRS-80 COMPUTING; v.1, n.1, August 78, p. 11-13)

(e) One way to absolutely guarantee a BAD Cload or Csave is to use a cassette tape with wrinkles or splices. In order to read the information from a tape, that tape must make very close contact with the recorder head—an impossibility if the tape isn't perfectly smooth.

CASSETTE UNPAIN, Cont.:

If a particular cassette turns out to have a bad spot then the usual solution is to discard the cassette or use it only for music henceforth. Stingy or frugal (stingy if it's the other guy frugal if it's you) computerists will prefer to cut out and discard the short piece of tape between the mylar leader and the bad spot. The end of the long piece should then be spliced to the leader and you're back in business. It is wise to re-label the 7-minute program on what is now a six, but used to be a 10-minute tape.

Fortunately, it turns out to be easy to locate a bad spot on tape. First Cload into the computer any long program which you know from experience will load and run properly Then Csave that same program onto the defective tape. Now rewind the tape and Cload? it. If there is a defect on the tape, it will stop in the middle of Cload? with a BAD message. Press Stop and then Eject and inspect the tape through the little windows at the edge of the cassette. The tape will usually stop within one inch after the defect passes the read head in the middle of the center window --making the defect visible in the right window.

If there is a visible defect you can cut it out as described above. If no defect is visible the trouble may have been due to a dust particle or a bad gain setting. Repeat the process a few times. If the Cload repeatedly stops at the same point (as indicated by the counter reading) but no defect is visible, it is probable that the defect is due to the magnetic coating and that spot should be cut out. Or only the other side of the cassette used--after appropriate tests, of course.

LOADING THE FRONT HALF

One trick you can do if you wish is Cload only the front half of a program. This seldom has any practical use but if you do it accidentally you will want to be able to recognize the symptoms. To do it on purpose, Cload a program, then press the Stop key on the CTR-41 before Cload is finished. You'll of course not get a Ready message and will have to press the Reset key to terminate the Cload command. Now List the new resident program (that means the new program you just put in RAM). Note that it starts out perfectly but ends in the middle where you stopped the tape recorder.

LOADING THE END

Loading the back half of a program is unfortunately a rather common problem and as far as I know produces only despair. Usually, it happens because a person forgets the mylar leader on practically all cassettes (ugh!) and cleverly Csaves the first part of the program on the leader. Very unfortunate! It can also happen if you turn the recorder on after Csave has already started. Without each of those all-important bytes at the beginning of every Csave, the computer can't possibly deal with the rest of the program. The really evil part of this problem is that when you try to Cload the back end of a program into the computer you always get the double asterisks (second asterisk not flashing though--that's helpful) and the resident program is always erased. This happens whether you used Cload with a file name or without. If you used Cload with a file name corresponding to a later program on the tape (after the half program), the computer will continue searching through the tape until it finds the right one and then Cload it.

BULK ERASING

Buying a bulk eraser isn't absolutely necessary, but it sure saves a lot of aggravation. If you try to record a new program on top an an old program or series of old programs, you will find that the new one will be recorded properly, completely erasing the old program underneath it.

Unfortunately, depending on where the new program was put, the front end and maybe the rear end of a couple of old programs will still be on the tape. Bad news. All the problems of dangling front ends and rear ends will plague you. It is easier to invest $12 and use a bulk eraser to get rid of all those old programs you don't want.

DOUBLE DUMP

One clever procedure to minimize the problem of Csaving programs on cassettes is known by the unappetizing name of the "double dump". Rather than record a program once, you record it twice, and all in one operation. Type Csave "a": Csave "b" then Enter and sit back and wait. It is wise to always use different file names on the two Csaves and to mark the cassette with tape recorder counter reading at (1) the beginning of "a" , (2) the end of "a" and the beginning of 'b" and (3) the end of "b". Occasionally the first two or three feet of a cassette tape will be of poor quality. If so, the double dump gives you a second recording nearer the center of the tape where the quality is likely to be better. And the chance of losing the program by missing a bit in both recordings is considerably reduced. If you are super cautious, a triple dump works fine too.

SYNTAX TRICKS

The manual says that the proper syntax is Csave "file name". Actually, if the command is used alone, the closing quotes are unnecessary. The command Csave "a works fine. But when there is a second command immediately following (on the same line), then closing quotes must be used before the colon. Csave "a": Csave "b works fine but Csave "a: Csave "b does not.

If you try Cload"a": Cload "b" you will find that it won't work. The computer will Cload "a" properly but then will stop and say Ready, ignoring Cload "b". That's pretty smart of the computer--after all, when you think about it, a double Cload doesn't make sense anyhow.

The computer does the same with a double Cload?, that is, it will do the first one and ignore the second. 1

You can use single or multiple Csaves inside the body of the program to good effect. But you must always use those closing quotes. Just make sure the tape recorder is all set up and ready to receive the Csave when the computer sends it. Using Cload inside the body of a program doesn't sound too useful but can be done if you wish. Besides remembering to set up the tape recorder before you start the run, also remember that the resident program will be erased before the new program is Cloaded. 4

On completing the Cload iside the program, the computer will stop and say Ready. It will not run the new program until you command it to. I suppose that using Cload? inside a program works too but I can't figure out a way to test it! Even if it does work--practical applications of such a trick will sure be few and far between.

STICKING RELAY

There is a relay contact in the keyboard unit that automatically starts and stops the tape recorder on computer command. Occasionally this contact will fail—usually due to using the wrong tape recorder—and cause problems in executing Csave and Cload commands. This problem is described at length in Tim Hensler's article. Should your relay contact be defective, have it repaired as soon as possible. If you have questions, comments or corrections on the above please send them to TRS-80 Computing as soon as possible and we will try to print the answers in the next issue along with the second half of the article. Courage.

ALTERNATIVES EXIST TO SCREEN PRINTER

This writer recently unloaded his screen printer at a reasonable price.

The screen printer is not extraordinarily expensive for a computer peripheral and may have been unjustly maligned.

It is fast, and on occasion has been known to print evenly. The alternative to the RS screen printer is the more expensive Centronics, which is a full-fledged computer printer with many of the niceties, including ordinary paper.

For those of us that cannot afford the $1300 for this model, however, there is a further alternative.

Centronics makes a microprinter called the P1 that will interface to the TRS-80 expansion interface unit. I've ordered one after being reassured by Centronics that the unit is compatible with the plug on the expansion interface. For further information, call Centronics.

The P1 is available at Hamilton-Avnet and other places for $395 in single-unit quantities and $375 for lots of 25.

The P1 prints at 180 lines per minute, uses a 5 by 8 dot matrix, and the aluminized paper comes out from the top rather than the side as in the screen printer case! You will need the expansioninterface, however. —William Barden

You can convert own TRS-80 with Level II ROM kit

By Dr. THOMAS B. PERERA
Squire Hill rd., N. Caldwell NJ 07006

There are several reasons why a person might want to convert the TRS-80 to Level II at home rather than sending it to the repair depot. He may not wish to do without the computer for a week or so, or may want to keep the Level I ROM.

The procedure is relatively simple but should only be attempted by people who are experienced with fine circuit board soldering techniques. Proceed at your own risk! So far, these instructions have only been used with "D" and "G" suffix boards.

The conversion kit contains a printed circuit board which has the Level II ROMs and associated circuits as well as four colored wires attached to it. You also receive a flat cable jumper and a 100 ohm resistor and a small 16 pin jumper.

HERE'S HOW

To convert your computer, proceed as follows:

First: unscrew the case screws, being certain to remember which screws go in which holes. Remove the case and carefully lay the keyboard face down toward you with the etch side of the main circuit board facing up. Be careful not to stress the small flat wire which connects keyboard to computer.

In the upper right corner of the board you will see the etches, drawn in figure 1. Solder the 100 Ohm resistor carefully to the two locations shown in the figure. Note: if your board has a 100 Ohm resistor on the component side of the board between the two pots (G-boards) don't install the 100 or resistor.

Figure 2 shows the 40 pin expansion connector. As indicated in the figure, cut the heavy etch coming from the leftmost pin and install a jumper so that the leftmost pin and the pin just to the right of it are connected together.

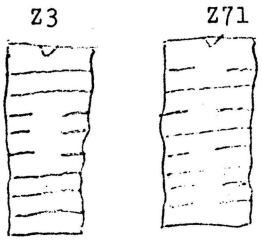
Now peel the paper covering from the adhesive attachments on the bottom of the small circuit board that holds the Level II ROMs. Stick the circuit board to the lower right corner of the computer board so that there is no more than 1/16 inch of space between the right edge of the computer board and the small circuit board (see Figure 3.) Be sure that the empty socket is to the right and the loose wires are to the left.

Carefully solder the four colored wires to the small feed through spots indicated in Figure 4. Be certain that they do not touch anything else on the board. If it is not already done on your machine, cut the etches indicated in Figure 5 and install the two jumpers as shown. The G board does not seem to require this modification. Turn the main board over and remove the Level I ROMs or ROM from socket Z33 and/or Z34. Carefully push the connector from one end of your flat jumper cable into socket Z33 and, keeping it from twisting, run the other end around the main board and plug it into the empty socket on the Level II ROM board. The G board requires that the cable be plugged into the Z34 socket. If you have another suffix on your board, you might try plugging the cable into Z34 if it doesn't work in Z33 Also try with and without the modification shown in Figure 5.

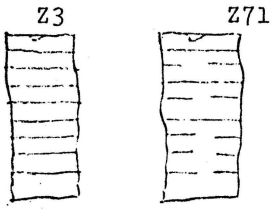
Finally, install the 16-pin jumper plug into socket Z3 so that you will have the proper connections as indicated in the chart below. The chart shows the necessary jumper connections for 4K, 16K, Level I, and Level II on the later boards (boards that use 16 pin sockets for Z3 and Z71).

I have not attempted the conversion from Level I to Level II on any of the earlier boards yet, and cannot be sure that these procedures will work with them. I have successfully converted three TRS-80s to Level II using these procedures and each machine works perfectly.

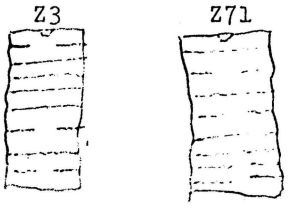
4K LEVEL II

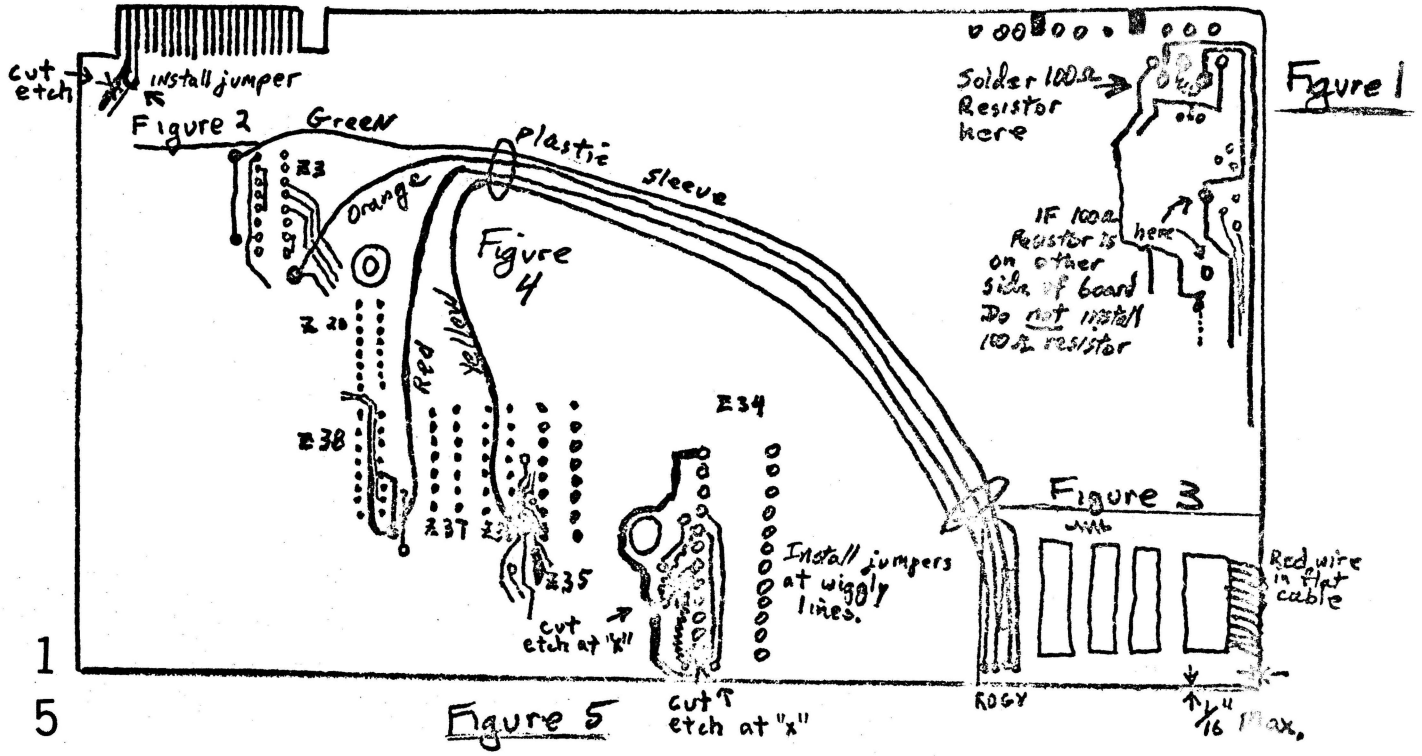


16K LEVEL II



16K LEVEL I

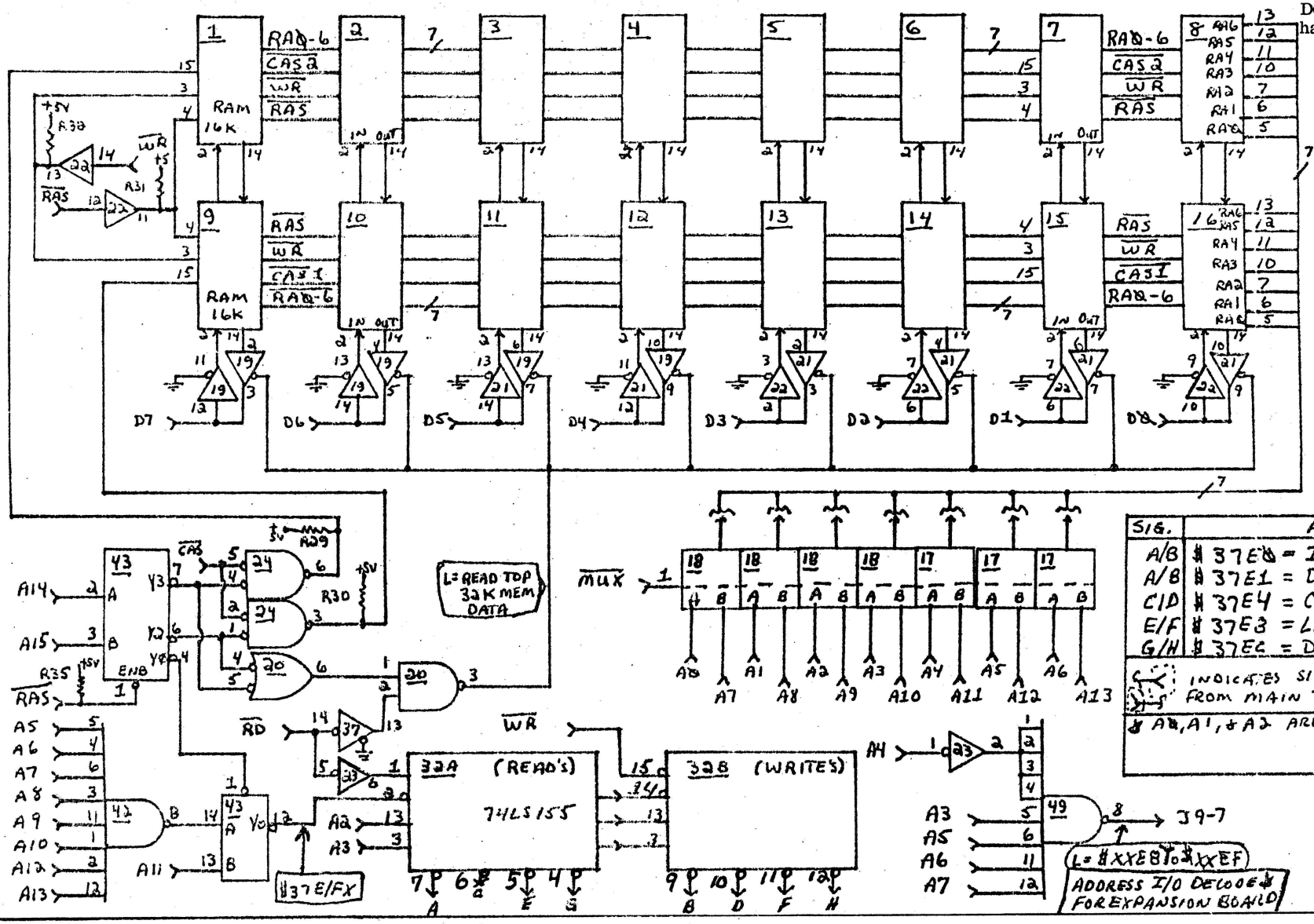




EXPANSION INTERFACE SCHEMATICS COURTESY OF PATRICK J. MCMAHON

TRS-80	RAM AND ADDRESS DECODES	EXPANSION INTERFACE		PG.1

NOTE:
Decoupling capacitors have been omitted.



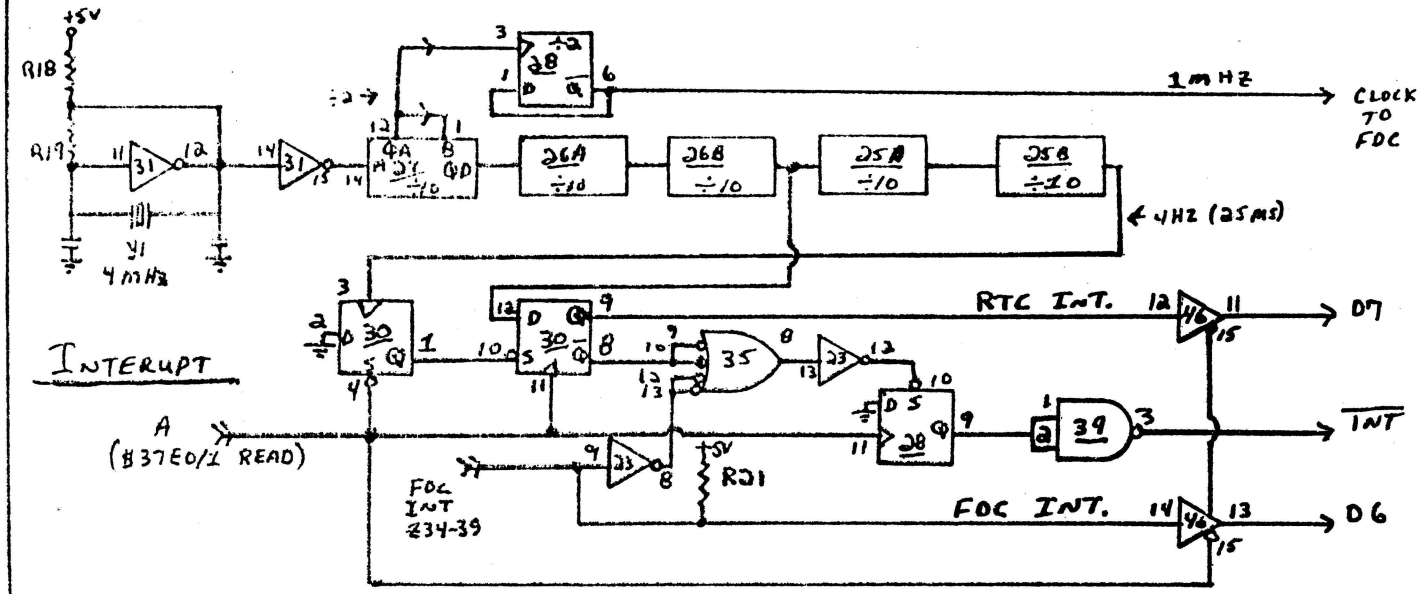
SIG.	ADDRESS DECODES
A/B	# 37E0 = INT. LATCH ADDRESS
A/B	# 37E1 = DISK SELECT LATCH ADD.
C/D	# 37E4 = CASSETTE SELECT ADD.
E/F	# 37E8 = LIN PRINTER PORT
G/H	# 37EC = DISK DATA I/O

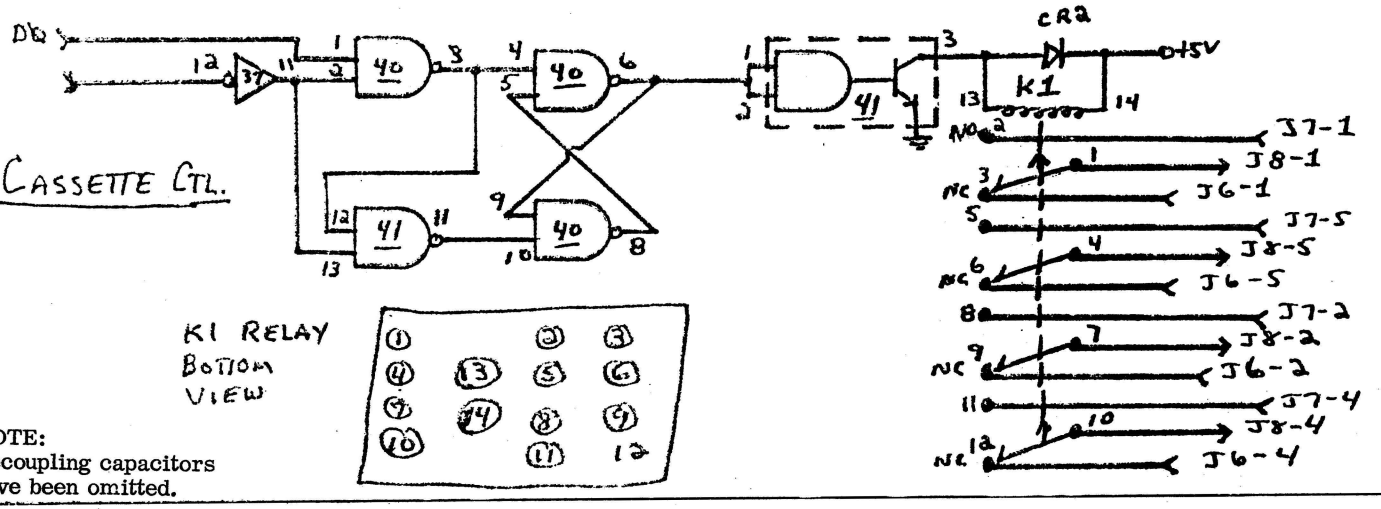
INDICATES SIGNAL DIRECT FROM MAIN TRS-80 JACK "J2"

A0, A1, & A2 ARE BROUGHT TO J9

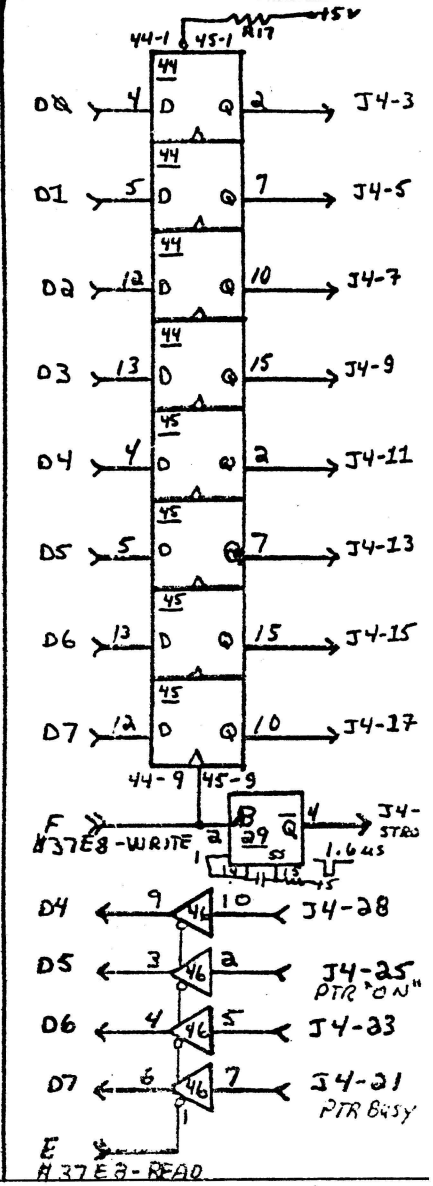
EXPANSION INTERFACE SCHEMATICS COURTESY OF PATRICK J. MCMAHON

TRS-80	INTERRUPT/CASSETTE/& LINE PRINTER	EXPANSION INTERFACE	LINE PTR. PORT ↓	PG.2

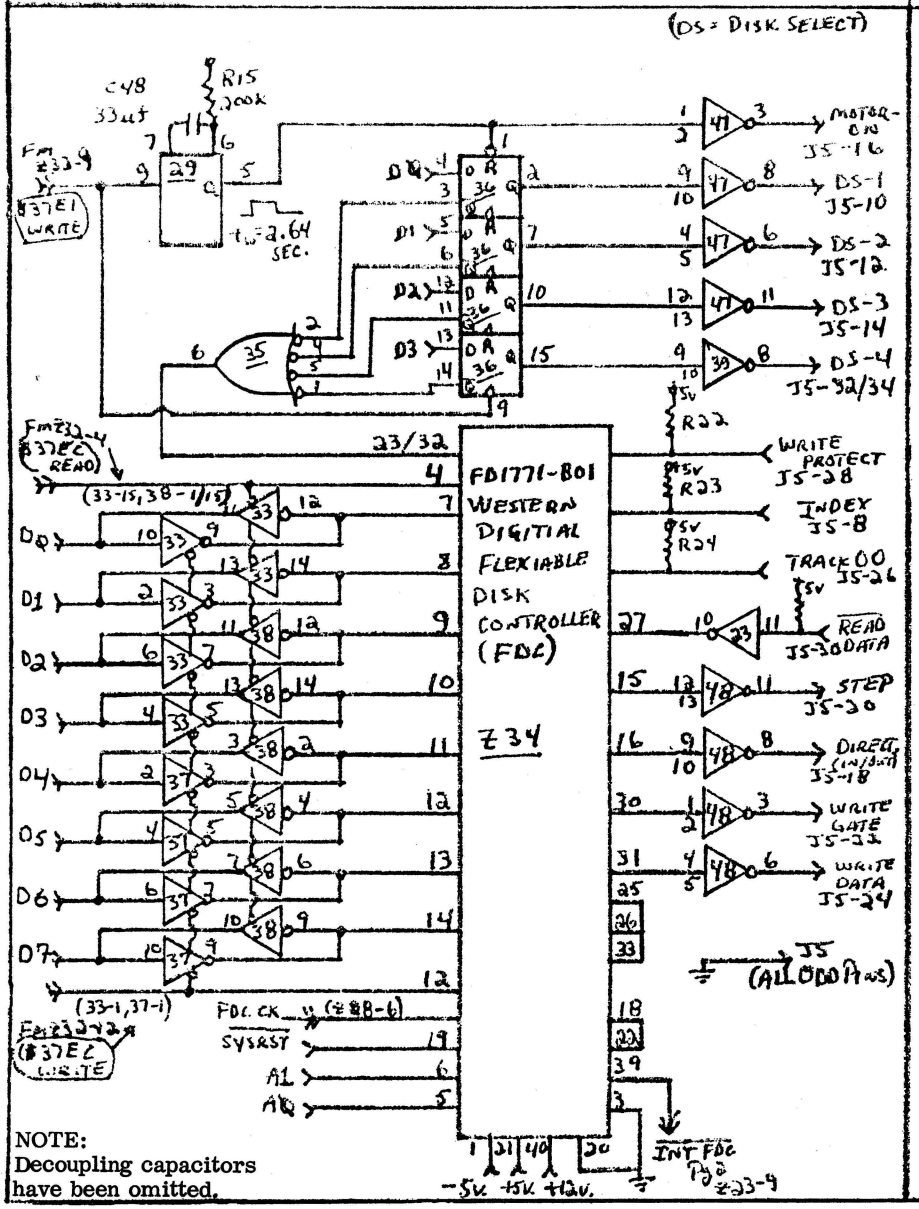




NOTE:
Decoupling capacitors have been omitted.



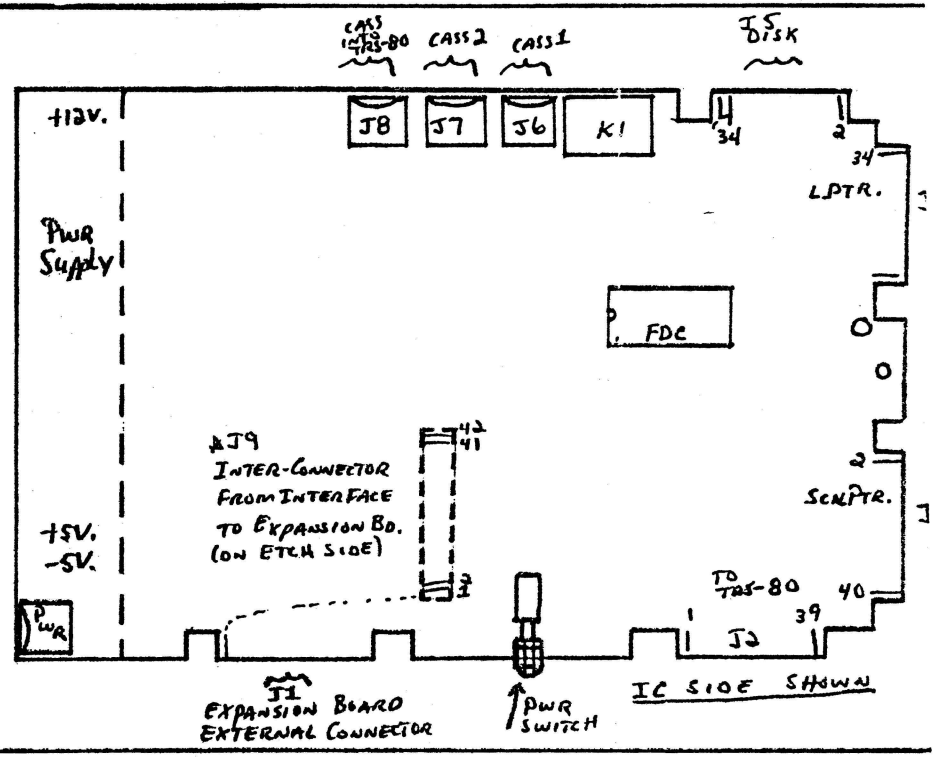
FLOPPY DISK CONTROLLER (FDC)



IC TYPES & BOARD LAYOUT

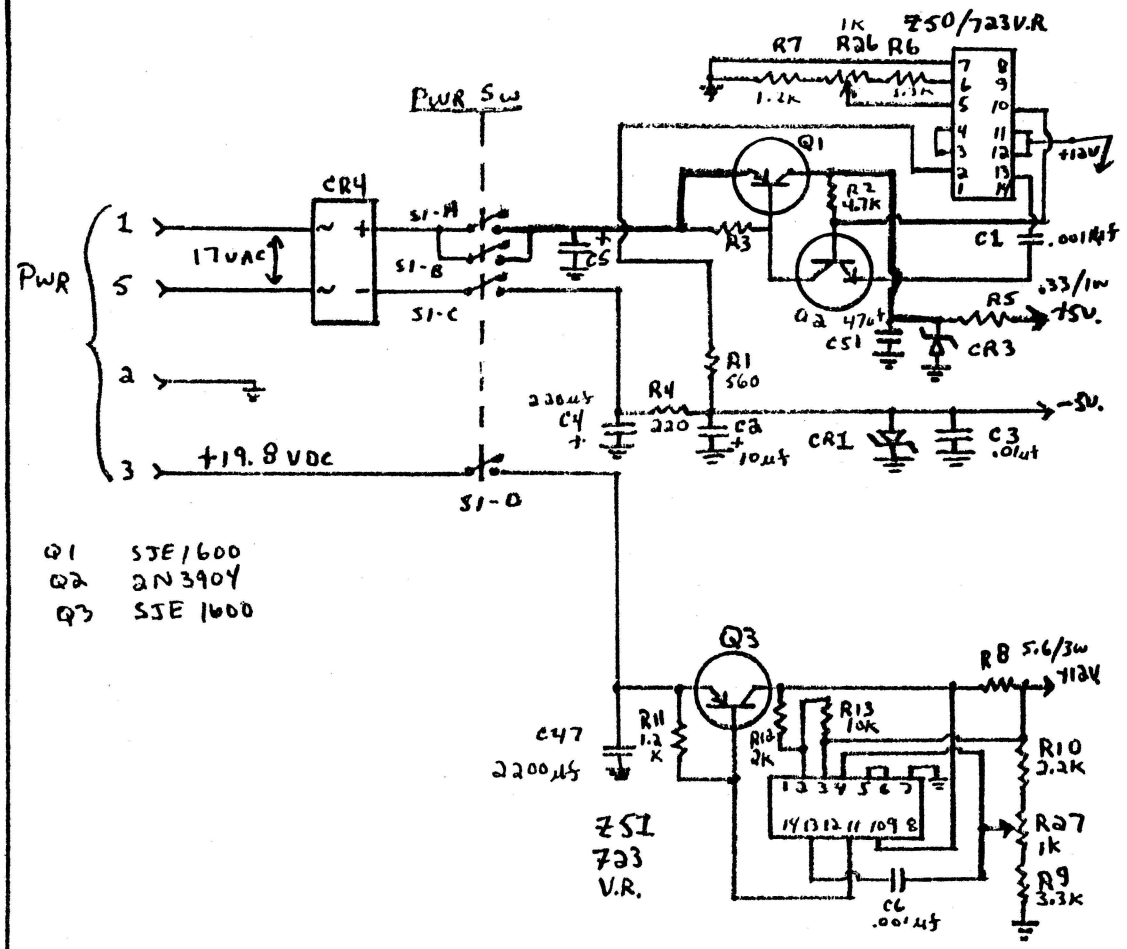
TRS-80 EXP/INTERFACE | PG. 3

IC Z#	IC TYPE	IC Z#	IC TYPE
1 - 16	16 RAM, Dy.	32	74LS155
17 & 18	74LS157	33, 37, & 38	74LS368
19, 21, 22, & 46	74LS367	34	FD1771-B01
20 & 40	74LS00	35	74LS20
23	74LS04	36, 44, & 45	74LS175
24	74LS32	39, 47, & 48	7438
25 & 26	MC 14518B	41	767427
27	74LS90	43	74LS139
28 & 42	74LS30	49 & 50	723 V.R.
29 & 31	74LS74		
30	74LS123		



EXPANSION INTERFACE SCHEMATICS COURTESY OF PATRICK J. MCMAHON

ON-BOARD POWER SUPPLY



Q1 SJE/600
Q2 2N3904
Q3 SJE 1600

NOTE:
Decoupling capacitors have been omitted.

TRS-80 EXPANSION INTERFACE

PG. 4

J-1 EXPANSION BOARD I/O
J-2 TRS-80 BUS
J-3 TRS-80 BUS
J-4 LINE PTR PORT
J-5 DISK PORT
J-6 CASS 1
J-7 CASS 2
J-8 CASS TO TRS-80
J-9 TAKE J-1 &TRS-80 I/O SIGNALS TO EXPANSION BOX
J-10 POWER SUPPLY

TELEDYNE KINETICS (B2046-42) EXPANSION BOARD CONNECTOR:

PIN	GOES TOO/FROM	PIN	FROM
* 1	J1-40	22	A2/J2-40
2	J1-38	23	$\overline{IN}$/J2-19
3	J1-36	24	$\overline{INT}$/J2-21
4	J1-34	25	D1/J2-22
5	J1-32	26	D2/J2-32
6	J1-30	27	D3/J2-26
7	J1-28	28	D0/J2-30
8	J1-26	29	A0/J2-25
9	J1-24	30	A1/J2-27
10	J1-22	31	D5/J2-28
11	J1-20	32	D4/J2-18
12	J1-18	33	$\overline{SYSRST}$/J2-2
13	J1-16	34	D7/J2-20
14	J1-14	35	D6/J2-24
15	J1-12	36	#XXE8-8EF/Z49
16	J1-10	37	**
17	**	38	**
18	**	39	+5V
19	**	40	
20	**	41	GND
21	$\overline{OUT}$/J2-12	42	

* PIN 1 IS TOWARD "POWER-SWITCH"
** NO CONNECTION - BUT, NOTE SIGNALS (i.e. $\overline{RD}$, $\overline{WR}$) CAN BE BROUGHT UP ON THESE PINS.

journal /ledger

NOTE: This 4K Journal/Ledger by Frank C. Heinisch is available on CIE's People's Software Tape I, in Level I, see current CIE TRS-80 Bulletin.

```
10  REM "J-L" PROGRAM :: JOURNAL - LEDGER PROGRAM
20 REM FRANK C. HEINISCH,  BOX 311, GENEVA, NEBR. 68361
21 REM APRIL 16, 1978
24 Q=0
25  REM L IS THE # OF LINES DISPLAYED
26  L=10
27 V=0
30 N=0:M=0:C=0:Y=0:I=0:S=0:X=0:D=0
31 U=0:A=0
50 DATAJAN.,FEB.,MAR.,APR.,MAY.,JUNE,JULY,AUG.,SEP.,OCT.,
      NOV.,DEC.
51 DATANOT LISTED
80 GOTO250
89 CLS
92 INPUT"WHAT IS THE FIRST CK # ";N
93 N=N-1
95 INPUT"ENTER MONTH -- NUMERICAL ^^^  ";A
97 IF A <13 THEN 100
98 STPRINT"YOU MISSED TRY A NUMBER LESS THAN 12 FOR A
       MONTH !!!":GOTO95
100 PRINT"ENTER >> 0 << TO ENTER MONTH; & >> 111 << WHEN
       ALL DATA IS IN."
102 Q=0
103 PRINT"ENTER >> 222 << TO CORRECT ENTRY"
109 PRINT"ENTER ALL THREE NUMBERS WITH ^^^^ COMMAS ]]]]
       IE. >> 1,13,1.45"
110 PRINT"#","DAY,","LEDGER CODE #,","AMOUNT"
114 N=N+1:V=N:PRINT"#  ";N;" ";
115 FOR X=1 TO A: READ B$:NEXTX:RESTORE:PRINTB$;" ";
116 INPUTD,C,M
119 IF(D=0)+(D=111)+(D=222)THENGOSUB4900
120 IF D=0THEN95
130 IF D=111THEN250
140 IF D=222THEN5500
150 IF D<32THEN210
160 PRINT"RE ENTER THE DAY -- IT MUST BE LESS THAN 32":
       GOTO114
170 IF A>.9THEN200
175 PRINT"YOU FORGOT TO ENTER A MONTH !!!";GOTO95
200 IF C<99THEN210
203 PRINT"LEDGER CODE CANNOT EXCEED 99,  LAST ENTRIES NOT
       USED":GOTO114
210 REM ^^^  ASSIGNS ARRAY NUMBERS ]]]
220 A(N)=N:A(N+750)=(A*10000)+(D*100)+C:A(N+1400)=M
230 Q=Q+1:IF Q=L THEN 100
240  GOTO114
250 CLS:PRINT"        >>>>>>   MENUE   <<<<<<"
251  Q=0
260 PRINT"LIST ALL DATA WITH TOTAL ----------------  1"
270 PRINT"LIST ALL DATA FOR ONE MONTH -------------  2"
280 PRINT"LIST ALL DATA PER ONE CODE NUMBER --------  3"
290 PRINT"LIST ALL DATA PER ONE CODE PER ONE MONTH -  4"
320 PRINT"LIST TOTALS FOR DESIGNATED CODE NUMBERS   5"
330 PRINT"LIST TOTALS FOR MONTH OF CODE NUMBERS  -  6"
340 PRINT"TO  STORE DATA ON TAPE  --------------- 7"
360 PRINT"TO ENTER DATA ON TAPE STORAGE -------- 8"
362 PRINT"TO ENTER DATA ------------------------- 9"
364 PRINT"TO CORRECT DATA ----------------------- 10"
366 PRINT"TO ENTER THE LOWEST & HIGHEST CK # ------ 11"
370 PRINT:INPUT"ENTER MENUE I.D. NUMBER --------------";I
371 CLS
399  IF I=11 THENGOSUB5600:GOTO250
400 IF I=7 THEN 10000
401  IF I=8 THEN 12000
402 IF I=9 THEN 89
403 IF I=10 THEN5500
404 IF V<1 THENGOSUB5600:GOTO250
409  A=0:Q=0:S=0:T=0
410  IF I=1 THEN 800
420 IF I=2 THEN 1200
430  IF I=3 THEN 1000
440  IF I=4 THEN 900
450 IF I=5 THEN 2200
460 IF I=6 THEN 2400
550 PRINT"YOU MISSED !!! ENTER 1 THROUGH 11":GOTO251
600 REM SUB TO RECONSTITUTE ARRAYS TO DATA
612 X=0:O=0:D=0:M=0:C=0
618 IF Q>0  THEN625
620 PRINT"DATE OF CK","ASSIGNED CK #","CK AMOUNT","LEDGER
       CODE #"
621 Q=.0001
625 A=INT(A(N+750)/10000)
628 IF A<1 THENA=13
635 FOR X=1 TO A: READ B$ : NEXTX
660 D=(A(N+750)/100)-(INT(A(N+750)/10000)*100)
662 D=INT(D+.1)
680 C=((A(N+750)*.01)-INT(A(N+750)*.01))*100
682  C= INT(C+.2)
690 RESTORE:RETURN
699 REM SUB TO DISPLAY DATA
700 IF A(N)<1 THEN 735
705 PRINTB$;" ";D,A(N),"$ ";A(N+1400),C
710 Q=Q+1
715  IF (A=13)+(N<.9) THEN 5500
716 IF(N>649)+(C>99)THEN5500
718 IF C>69 THEN725
720 S=S+A(N+1400):GOTO730
725 T=T+A(N+1400)
730 IF Q>L THEN738
735  RETURN
738 PRINT"SUBTOTAL OF CKS $ ";S;"  &  OF  DEPOSTIT  $ ";T
739 PRINT"RUNNING BALANCE: $"; T-S
740 INPUT "PRESS ENTER TO CONTINUE ";A$
750 Q=0:RETURN
800 CLS:PRINT"JOURNAL OF ALL DATA ENTERED"
801 Q=0
825 FORN=U TO V:GOSUB600:GOSUB700:NEXTN
850 PRINT"TOTAL JOURNAL FOR ";V;" ENTERIES"
852 PRINT"  TOTAL CHECKS $ ";S;"  AND  DEPOSITS  $ ";T;
       " BALANCE $ ";T-S
860  GOSUB5000
900 CLS:PRINT"ALL DATA FOR ONE MONTH, PER ONE CODE #,
       DISPLAYED WITH SUM"
901  Q=0
910 INPUT"ENTER MONTH, (A NUMBER) & LEDGER CODE # >> USE
       COMMAS <<";F,G
930 FORN=U TO V:GOSUB600
940  IF (A=F)*(C=G) THEN 960
950 NEXTN:GOTO985
960  IF A(N)<1 THEN 985
980  GOSUB 700:NEXTN
985 FOR X=1 TO F:READ B$:NEXTX
990 PRINT"TOTAL FOR LEDGER CODE ";G;" IN ";B$;" IS $ ";S
995  GOSUB5000
1000 CLS:PRINT"ALL DATA FOR ONE CODE IS DISPLAYED AND
       TOTALED"
1010 INPUT"ENTER CODE NUMBER ^^^^^^^  ";G
1020 FORN=U TO V:GOSUB600
1030 IF C=G THEN 1050
1040 NEXTN:GOTO1070
1050 IF A(N)<1 THEN 1070
1060 GOSUB700:NEXTN
1070 PRINT"TOTAL OF ALL CKS UNDER CODE # ";G;" IS $ ";S
1073 PRINT"AND ALL DEPOSITS IS $ ";T;" A BALANCE OF $";T-S
1080 GOSUB 5000
1200 CLS:PRINT"JOURNAL OF ALL DATA FOR ONE MONTH"
1201  Q=0
1220 INPUT"ENTER MONTH ----- NUMERICAL";F
1230 FORN=U TO V:GOSUB600
1240  IF A=F THEN 1260
1250 NEXTN
1260 IFA(N)<1 THEN 1275
1270 GOSUB 700:NEXTN
1275 FOR X=1 TO F:READ B$:NEXTX
1280 PRINT"TOTAL FOR JOURNAL IN MONTH OF ";B$;" IS:"
1282 PRINT" CKS = $ ";S;" ** DEP = $";T;" ** BAL = $";T-S
1290 GOSUB5000
2000  CLS:PRINT"TOTAL FOR ALL UNDER ONE CODE NUMBER"
2010 INPUT " ENTER CODE NUMBER  ^^^^  ";C
2020 FOR X = 1 TO V
2030 IF A(N*10+3) = C THEN 2050
2040  NEXT X
2050  S = A(N*10+2)+S : NEXT X
2070 GOSUB 5000
2100 CLS:PRINT"TOTAL FOR MONTH UNDER ONE CODE NUMBER"
2110 INPUT"ENTER MONTH -- A NUMBER AS APRIL IS 4 ^^^^  ";D
2120 INPUT"ENTER CODE NUMBER ^^^^  ";C
2130  FOR X = 1 TO V
2140  IF INT((A(N*10+1))1000/) = D THEN 2160
2150  NEXTX
2160 IF A(N*10+3) = C THEN 2180
2170 NEXTX
2180 S = A(N*10+2)+S : NEXTX
2190 PRINT"TOTAL IN ";D;" MONTH UNDER CODE ";C;" IS  $ ";S
2195  GOSUB 5000
2200  CLS:PRINT "TOTALS OF EVERY CODE NUMBER"
2210 INPUT"WHAT IS THE LOWEST & HIGHEST CODE # >> USE A
       COMMA";B,H
2220 GOSUB4000:Q=1:F=0
2230 FORN=U TO V:GOSUB600:GOSUB4100
2232 PRINT@522;">>> THINKING <<<  ^^^ NOW AT # ";N;" ]]]"
```

```
2234 NEXTN
2235 Q=0:A(H+651)=0
2240 FOR C=B TO H STEP 2
2250 IFQ>0 THEN 2270
2260 Q=0:PRINT"CODE #", "TOTAL","CODE #","TOTAL"
2270 PRINTC,"$ ";A(C+650),C+1,"$ ";A(C+651)
2271 IF C<>69 THEN 2273
2272 S=S-A(C+651):T=T+A(C+651)
2273 IF C<70 THEN2280
2274 T=T+A(C+650)+A(C+651):GOTO2290
2280 S=S+A(C+650)+A(C+651)
2290 Q=Q+1:IFQ>L THEN2305
2300 NEXTC:IF Q<=L THEN 2330
2305  Q=0
2310 PRINT"SUB-TOTAL OF CK CODES IS $ ";S;"  & OF DEPOSIT
       CODES IS $ ";T
2320 INPUT"PRESS ENTER TO CONTINUE";A$:NEXTC
2330 PRINT"TOTAL OF CODE # ";B;" TO ";H;" IS :"
2332 PRINT"CKS = $ ";S;" :::  DEP = $ ";T;" ::: BAL = $ ";
       T-S
2338 T=0:S=0
2340 IF F=0 THEN 2365
2350 FORX=1 TO F:READ B$:NEXTX
2360 PRINT"THESE TOTALS ARE ONLY FOR THE MONTH OF ";B$
2365 PRINT"ENTER .1 TO REPEAT THE DISPLAYED TOTALS"
2370 GOTO5000
2400 CLS:PRINT"TOTALS OF SPECIFIED CODE #S FOR DESIGNATED
       MONTH"
2410 INPUT"ENTER MONTH ---- NUMERICAL ";F
2420 INPUT"WHAT IS THE LOWEST AND HIGHEST CODE # >> USE A
       COMMA";B,H
2430 GOSUB4000:Q=1:S=0
2440 FORN=U TO V:GOSUB600:IF A=F THENGOSUB4100
2450 GOTO2232
2500  PRINT"TOTAL FOR ONE MONTH OF ALL CODE NUMBERS"
2510 INPUT"ENTER MONTH -- USE NUMBER, AS APRIL IS 4 ^^^ ";D
2520 FORX=1TO V
2530  IF INT((A(N*10+1))/1000) = D THEN GOSUB 4000
2540 NEXTX
2550 GOTO2240
2960 PRINT"TOTAL UNDER CODE NUMBER ";C;" IS  $ ";S
4000  REM SUB FOR TOTALING ALL CODE NUMBERS
4010 FORZ=(650+B)TO(H+650)
4020 A(Z)=0:NEXTZ:RETURN
4030  S=0:T=0
4100 A(C+650)=A(C+650)+A(N+1400):RETURN
4900 N=N-1:V=N:RETURN
5000 REM SUB TO END PROGRAM
5004 PRINT"THE LOWEST CK # IS ";U;" & HIGHEST IS ";V;" TO
       CHANGE ENTER .5"
5005 PRINT"TO ENTER MORE DATA TYPE  1  :::::  FOR MENUE
       TYPE  2"
5007 INPUT"TO CORRECT ENTRY TYPE 3     :::::  TO END TYPE
       4";Y
5008 IF Y=.1 THEN2235
5009 IF Y=.5 THENGOSUB5600:GOTO250
5010 IF Y=1 THEN5100
5020 IF Y=2 THEN250
5030 IF Y=3 THEN5500
5040 IF Y=4 THENEND
5050 PRINT"YOU MISSED !!!  TRY 1 THROUGH 4 ":GOTO5005
5100 N=V:GOTO95
5500 PRINT"TO CORRECT AN ENTRY:"
5510 INPUT"ENTER THE CORRECT MONTH, DAY, CK #, CK AMOUNT,
       CODE";A,D,N,M,C
5520 A(N)=N:A(N+1400)=M
5522 A(N+750)=A*10000+D*100+C
5530  GOTO5000
5600 INPUT"ENTER THE LOWEST AND HIGHEST CK #,(USE COMMA) "
       ;U,V:RETURN
10000 PRINT"TO STORE DATA ON THE CASSETTE TAPE:"
10010 PRINT"   FIRST, REWIND TAPE THE END OF LAST DATA,
       IF ANY"
10020 PRINT"    SECOND, PRESS RECORD AND PLAY KEYS"
10030 PRINT"    THIRD, ENTER THE BEGINNING AND ENDING CK #
       TO BE SAVED"
10040 INPUT"        USE A COMMA ^^^^";B,E
10060 PRINT"   FOURTH, KEEP A RECORD OF WHICH CK #S ARE
       ON DATA TAPE"
10065 INPUT"    FIFTH, PRESS ENTER WHEN READY";A$
10070  FOR N=B TO E
10080 PRINT #-1,N;",";A(N+750);",";A(N+1400)
10090 PRINT@ 520;"CK # ";N;" HAS JUST BEEN RECORDED"
10100 NEXTN:PRINT"ALL DATA IS NOW ON TAPE, PRESS STOP KEY":
       GOTO5000
12000 PRINT"TO RECOVER DATA STORED ON TAPE ."
12010 PRINT" FIRST, REWIND TAPE & TURN OFF YELLOW SWITCH"
12020 PRINT"  SECOND, PRESS RECORDER'S PLAY KEY"
12030 INPUT   THIRD, ENTER THE LOWEST & HIGHEST CK # TO BE
       RECOVERED ";L,H
12040 INPUT"  FORTH, PRESS ENTER WHEN READY";A$
12045 V=.001
12050 FORA=L TO H: INPUT #-1, X,Y,Z
12055 IF L=X THEN 12070
12057 IF V=.001THEN12060
12059 IF V<X THEN 12062
12060 V=X
12062 IF X>=L THEN 12070
12064 IF L-V<0 THEN 12150
12068 PRINT@520;"CK # ";X;" HAS ^^ NOT ]] BEEN ENTERED ":
       NEXTA
12070 N=X:A(N)=N:A(N+750)=Y:A(N+1400)=Z
12080 PRINT@520;"CK # ";N;" HAS JUST BEEN ENTERED"
12085 NEXTA:IF L=V THEN 12093
12087 FORA=1 TO L-V:INPUT #-1,X,Y,Z
12088 N=X:A(N)=N:A(N+750)=Y:A(N+1400)=Z
12090 PRINT@520;"CK # ";N;" HAS JUST BEEN ENTERED"
12092 NEXTA
12093 PRINT"ALL DATA HAS NOW BEEN ENTERED, TURN OFF
       RECORDER"
12094 U=L:V=H:L=10
12095 A=0:X=0:Y=0:Z=0:GOTO5000
12150 PRINT"RE-ENTER LOWEST & HIGHEST CK #, SINCE ";L;"
       IS LESS THAN THE"
12160 PRINT"FIRST CK # ON TAPE OF ";X:GOTO12030
```


*trek**

George W. Scheil contributed this 4K Level I copyrighted Startrek, for which he sells tapes at just $5 postpaid.

This has been reproduced with his permission; no reproduction in any form is allowed without written permission of the author.

To win Startrek you must destroy all of the klingon ships, which are sometimes very treacherous. Four levels of difficulty set the maximum number of klingons. The following commands are available to players:

- S—Gives ship status and a map of the local quadrant. Coordinates are by row, column.
- E—Transfers energy to or from the shields. If the shields are down enemy fire can be fatal.
- G—Gives location of the klingons and your starbase.
- P—Fires phasors, which are not blocked by obstacles and are self-targeting, but attenuated by distance. Power is evenly divided among all klingons in the local quadrant. Phasor energy fired is removed from Enterprise's main energy banks.
- T—Fires a torpedo, and requires 250 energy units.
- J—Moves the ship. Warp drive changes quadrant, and impulse power is used for moving within a quadrant. Warp drive uses a warp tunnel, adds one stardate, and the jump is made directly to the destination. Enter the destination coordinates separated by a comma. Impulse power requires a vector and distance. A ship cannot leave the local quadrant on impulse power. Warp drive uses 150 units of energy.

Klingons have from one to 1,000 energy units upon entry of the Enterprise into their quadrant. They always strike the Enterprise with half their shield power. Klingons in the vicinity of the Enterprise will always retaliate if the Enterprise attacks and may also attack on their own.

To replenish supplies dock left or right of the starbase. The Enterprise, starbases and stars are shown with asterisks ("*"), and klingons are three pluses ("+++").

Each game starts with 5,000 energy units and four torpedoes, maximum load for the Enterprise. Everything takes energy, at least 10 units. Don't run out. Coordinates for impulse power and torpedoes are:

```
4   3   2
5  ENT  1
6   7   8
```

Klingons do not move but are set to random locations whenever the Enterprise enters their quadrant. The game is lost if a player: collides with any object, runs out of energy, or reaches stardate 3050.

The game is supplied on a cassette, with the playing rules and the game itself recorded once on each side of the cassette. For a tape of 4K Level I Startrek, send $5 to Scheil at: #118-1638 Bushman dr., Kansas City MO 64110.

(NOTE: To reduce memory requirements, George has divided his startrek into two programs: the instructions, below, and the game, which follows.)

```
10  REM COPYRIGHT 1978 BY GEORGE W. SCHEIL
20  CLS
350  PRINT"TO WIN STARTREK YOU MUST DESTROY ALL OF THE"
360  PRINT"KLINGON SHIPS (WHICH ARE SOMETIMES VERY DANGEROUS)"
370  PRINT"4 LEVELS OF DIFFICULTY SET THE MAXIMUM # OF KLINGONS"
400  PRINT"'S' GIVES SHIP STATUS AND MAP OF THE LOCAL QUADRANT"
405  PRINT"  COORDINATES ARE BY ROW, COLUMN"
410  PRINT"'E' TRANSFERS ENERGY TO OR FROM THE SHIELDS"
420  PRINT"  IF THE SHIELDS ARE DOWN ENEMY FIRE CAN BE FATAL"
425  PRINT"'G' GIVES THE LOCATION OF THE KLINGONS AND YOUR STARBASE"
430  PRINT"'P' FIRES YOUR PHASORS.  PHASORS ARE NOT BLOCKED BY"
435  PRINT"  OBSTACLES AND ARE SELF TARGETING"
440  PRINT"  BUT THEY ARE ATTENUATED WITH DISTANCE"
445  PRINT"  POWER IS EVENLY DIVIDED AMONG ALL KLINGONS IN THE"
450  PRINT"  LOCAL QUADRANT."
452  INPUT"PRESS ENTER TO CONTINUE";A$
453  CLS
455  PRINT"KLINGONS HAVE FROM 1-1000 ENERGY UNITS UPON ENTRY"
460  PRINT"OF THE ENTERPRISE INTO THEIR QUADRANT."
465  PRINT"THEY ALWAYS FIRE WITH HALF THEIR SHIELD POWER"
470  PRINT"'T' FIRES A TORPEDO"
475  PRINT"'J' MOVES THE SHIP, WARP DRIVE CHANGES QUADRANT,"
480  PRINT"AND IMPULSE POWER IS USED FOR MOVING WITHIN A QUADRANT"
482  PRINT"WARP DRIVE USES A WARP TUNNEL, ADDS 1 STARDATE,"
483  PRINT"AND THE JUMP IS MADE DIRECTLY TO THE DESTINATION."
484  PR     NTER THE COORDINATES SEPARATED BY A COMMA."
485  PRINT"IMPULSE POWER REQUIRES A VECTOR AND DISTANCE."
486  PRINT"A SHIP CANNOT LEAVE THE LOCAL QUADRANT ON IMPULSE POWER"
488  PRINT
490  PRINT"TO REPLENISH SUPPLIES DOCK LEFT OR RIGHT OF THE STARBASE"
495  PRINT"<*>=ENTERPRISE,>*<=STARBASE,+++=KLINGON,*=STAR"
496  PRINT"EVERYTHING TAKES ENERGY.  DON'T RUN OUT."
500  INPUT"PRESS ENTER TO CONTINUE";A$
510  CLS
520  PRINT"COORDINATES FOR IMPULSE POWER AND TORPEDOES:"
530  PRINT"  4   3   2"
540  PRINT:PRINT"  5  ENT  1":PRINT
550  PRINT"  6   7   8"
560  PRINT:PRINT"KLINGONS MAY FIRE AFTER ANY ENTRY IF THE "
570  PRINT"  ENTERPRISE IS IN AN ENEMY QUADRANT."
580  PRINT"KLINGONS DO NOT MOVE BUT ARE SET TO RANDOM LOCATIONS"
590  PRINT"WHENEVER THE ENTERPRISE ENTERS THEIR QUADRANT."
600  PRINT"THE GAME IS LOST BY:  COLLISION WITH ANY OBJECT,"
610  PRINT"  RUNNING OUT OF ENERGY, OR REACHING STARDATE 3050."
```

*trek**

```
5  REM COPYRIGHT 1978 BY GEORGE W. SCHEIL
10  CLS:FORI=1TO89:A(I)=0:NEXTI
12  DATA1,-7,-8,-9,-1,7,8,9
20  INPUT"PICK A #(10 TO 100)";X:FORI=1TOX:Z=RND(0):NEXTI
40  T=4:E=5000:F=0:FORI=81TO83:A(I)=RND(64):NEXTI:K=0:L=0
60  Y=8*RND(7):X=RND(5):FORI=1TO3:A(83+I)=Y+X+I:NEXTI
70  INPUT"LEVEL(1 TO 4)";Z:IF(Z<1)+(Z>4)THEN70
80  FORI=87TO89:A(I)=RND(Z+1):K=K+A(I):NEXTI
90  D=3045-K+Z
110  X=A(81):GOSUB890:U=M:V=N:GOSUB900:GOSUB500
120  A(65)=T:A(66)=E:G=1:S=3:J=5:T=6:P=2:E=4
140  INPUT"COMMAND";X:B=10:T=A(65):E=A(66)
150  GOSUB1850:IF(X<1)+(X>6)THEN170
160  ON X GOSUB1110,730,500,410,1410,1190
170  GOSUB1050:IF D<3050 THEN190
190  IF K>0 THEN120
200  PRINT:PRINT"CONGRATULATIONS! KLINGON FORCE WIPED
       OUT":END
410  INPUT"ENERGY TRANSFER=";X:IF X<0THEN450
420  IF E>=XTHEN440
430  PRINT"NOT ENOUGH ENERGY":RETURN
440  E=E-X:F=F+X:RETURN
450  IF F<X THEN430
460  GOTO440
500  FORI=1TO8:PRINTTAB(3*I);I;:NEXTI:PRINT
505  FORI=1TO8:PRINTI;:FORJ=1TO8
510  ON A((I-1)*8+J)+1GOTO540,550,560,570,580
540  PRINT"   ";:GOTO590
550  PRINT"<*>";:GOTO590
560  PRINT">*<";:GOTO590
570  PRINT"+++";:GOTO590
580  PRINT" * ";:
590  NEXTJ:PRINT:NEXTI
600  PRINT"STARDATE";D;" TORPEDOES=";T
630  PRINT"QUADRANT";U;V;"SECTOR";W;Y;"ENERGY=";INT(E);
640  PRINT"SHIELDS=";INT(F):RETURN
730  INPUT"PHASOR ENERGY TO FIRE";X:IF X<=0 THENRETURN
740  IF X>ETHEN430
750  P=X
760  B=X:GOSUB1850:IFL>0THEN780
770  PRINT"NO KLINGONS":RETURN
780  FORI=71TO75:IF A(I+5)<=0THEN860
785  X=A(I):GOSUB890:A=(:ABS(W-M)+ABS(Y-N)+3/4
790  A(I+5)=A(I+5)-P/L/A:PRINT"KLINGON AT";M;N;
820  IF A(I+5)<=0THENZ=A(I):GOSUB1950:GOTO860
830  PRINT"HAS ENERGY OF";INT(A(I+5));" RETURN SHOT=";
835  B=A(I+5)/2:PRINTINT(B):GOSUB1900:GOTO860
850  PRINT"TOTAL JOURNAL FOR " V;" ENTRIES"
860  NEXT: RETURN
890  M=INT((X-1)/8+1):N=X-(M-1)*8:RETURN
900  FORI=1TO80:A(I)=0:NEXTI:A(A(82))=1
910  FORI=84TO86:IF(A(I)=A(81))*(A(I+3)>0)THENC=I:GOTO920
915  NEXTI:L=0:GOTO930
920  L=A(I+3):FORJ=1TOL
924  X=RND(64):IF A(X)>0 THEN924
927  A(X)=3:A(70+J)=X:A(75+J)=RND(1000):NEXTJ
930  X=A(82):GOSUB890:W=M:Y=N
940  IF A(83)<>A(81)THEN960
950  X=RND(64):IF A(X)=0THENA(X)=2:GOTO960
955  GOTO950
960  FORI=1TO12
970  X=RND(64):IF A(X)=0THENA(X)=4:GOTO1000
980  GOTO970
1000  NEXTI
1050  IF L=0THENRETURN
1060  FORI=1TO5:IF(RND(0)>.5)+(A(75+I)<=0)THEN1100
1070  X=A(70+I):GOSUB890:PRINT"KLINGON AT";M;N;"ATTACKS";
1080  B=A(75+I)/2:PRINT" ENEMY SHOT=";INT(B):GOSUB1900
1100  NEXTI:RETURN
1110  X=A(83):GOSUB890:PRINT"STARBASE AT";M;N
1120  FORI=84TO86:IF A(I)=0THEN1150
1130  X=A(I):GOSUB890:PRINTA(I+3);"KLINGONS AT";M;N
1150  NEXTI:RETURN
1190  T=T-1:IF T<0THENPRINT"NO TORPEDOES":RETURN
1200  B=250:GOSUB1850:GOSUB1940:Z=A(82)
1220  Z=Z+X:IF(Z<1)+(Z>64)THEN1260
1235  IF(A(Z)=4)+(A(Z)=2)THEN1260
1240  IF A(Z)=0THEN1220
1250  X=Z:GOSUB890:PRINT"KLINGON AT";M;N;:GOSUB1950:
       GOTO1050
1260  PRINT"MISSED":GOTO1050
1410  INPUT"WARP(0) OR IMPULSE(1)";X
1420  IF X=0THENINPUT" 0";X,Z:D=D+1:GOTO1470
1425  GOSUB1940
1430  INPUT"DISTANCE(1 TO 7)";Z:Z=INT(Z):IF(Z<1)+(Z>7)
       THEN1430
1440  FORI=1TOZ:A(A(82))=0:A(82)=A(82)+X
1442  IF(A(82)<1)+(A(82)>64)THENA(82)=RND(64)
1445  IF A(A(82))>0THENPRINT"COLLISION!! SHIP DESTROYED":
       END
1450  A(A(82))=1:NEXTI
1455  X=A(82):GOSUB890:W=M:Y=N:B=10:GOSUB1850
1460  IF(A(A(82)-1)=2+(A(A(82)+1)=2)THENT=4:E=5000:F=0
1465  GOTO500
1470  B=150:GOSUB1850:IF(X<1)+(X>8)THENRETURN
1480  IF(Z<1)+(Z>8)THENRETURN
1485  U=X:V=Z:A(81)=(U-1)*8+V:GOSUB900:GOSUB500:RETURN
1850  IF B<E THEN1870
1860  E=E+F:F=0
1870  E=E-B:IF E>0THENRETURN
1880  PRINT"ABANDON SHIP!  NO ENERGY LEFT":END
1900  IF B<F THEN1920
1910  E=E+F:F=0:B=2*B:PRINT"DANGER--SHIELDS DOWN":
       GOTO1870
1920  F=F-B:IF F<=0THEN1880
1930  RETURN
1940  INPUT"COURSE";X:IF(X<1)+(X>8)THEN1940
1942  RESTORE:FORI=1TOINT(X):READ A:NEXTI:X=A:RETURN
1950  PRINT"DESTROYED":K=K-1:L=L-1:A(A(+3)=A(A(+3)-1:A(Y)=0
1960  FORI=71TO75:IF A(I)=7THENA(I)=0:NEXTI
```

FT. WORTH REPORT

Tandy death saddens annual meeting

--ASSOCIATE EDITOR TRS-80 COMPUTING

Charles Tandy, 60, chairman of Tandy Corporation died in his sleep during an afternoon nap Nov. 4 in Ft. Worth.

Although Tandy was reported 10 years ago to have suffered a heart attack, his friends and associates were shocked by his sudden passing. He is survived by his wife, Anne Burnett Tandy, and her three children. He and his wife were both active in Ft. Worth civic affairs and supported many organizations including Texas Christian University and Southern Baptist Radio-Television Commission.

Initial fears that his death might trigger a selling wave of Tandy stock proved groundless. Stock price held relatively firm throughout the following week. At Tandy's annual stockholders meeting in Ft. Worth Nov. 9, just five days after his death, the mood was restrained but optimistic. There was standing-room only in the elegant new Ft. Worth Public Library. Almost as many people were standin aisles or sitting on the thickly-carpeted floor as were sitting in seats.

The man in the center of the speakers' table opened the meeting with the words "My name is Phil North and I have been chairman of Tandy Corporation for 16 hours. We will now stand for one minute of silence in honor of our departed friend." During the brief meeting some 25 or so Tandy executives were introduced, of which only one was born in Texas! The yankees have arrived in Ft. Worth in force!

Lewis F. Kornfeld , Jr., Radio Shack president, gave a review of various Tandy product activities during the year—and not surprisingly, the TRS-80 came in for a large share of his attention. He made a comment that the thing about the TRS-80 project that was most frightening to management was the fact that it would have no competition in its price range. "And if there is anything that frightens management more than too much competition it is no competition at all." His gamble on a price of $599 and Steve Leininger's design for the TRS-80 have certainly paid off.

Although overlooked in the flood of introductions and kudos for the past year's successes, Steve Leininger seemed happy and understandably proud. He was quick to give credit for the encouragement of Charles Tandy and the courage of Lewis Kornfeld who had the "guts to say GO" when it came time for a decision on the TRS-80 project.

New manuals on the way

When a new product line like the TRS-80 comes out, everyone is sympathetic about the engineering and production problems that can delay the equipment.

Not many people, however, are aware of the parallel problems in getting out the manuals. You can't very well write an acceptable manual until the design is final and a few working systems exist. And you don't like to delay delivery of systems until the final printed manual is available. Compromises are inevitable.

Radio Shack uses a system in which a temporary manual is delivered with early systems, to be followed later by a more extensive, prettied up and complete manual. In the long run this system proves ideal although a few early users who got temporary manuals may gripe a bit. That is a minor penalty for being the first kid on your block with the new equipment.

Dave Gunzel, technical publications manager and editor, is in charge of producing the TRS-80 manuals. He is to be congratulated for the extremely high quality and usefulness of his final manuals—although one can't be very complimentary about some of those temporary manuals (like DOS 2.0 — Ugh!) except to say that they were delivered with the equipment — a fact not to be sneered at!

Gunzel and his crew have been busy churning out some elegant new manuals, some just now available and others to be seen soon:

TRS-80 MICRO COMPUTER Technical Reference Handbook, #26-2103 — $9.95

This is now available at the Fort Worth Radio Shack Computer Center and can be ordered soon if not now available at your local dealer. A gem, this answers all those questions about how the computer hardware works, in language suitable for experienced digital circuit tinkerers. If you would have difficulty repairing an abacus forget about this manual.

T-BUG MANUAL

Is due in late November or early December, and will discuss T-Bug use in considerable detail.

DOS 2.2/DISK BASIC USERS' MANUAL, #26-2104 — $5.95

Also due in late November or early December, thank goodness!

RS232-C BOARD, #26-1145

Available in November, will be shipped with what is described as a very complete manual.

EXPANSION INTERFACE TECHNICAL MANUAL

Is on the way and with luck will be available in January or February, 1979. It is said to contain extensive information on hooking up the expansion interface with outside equipment — a use which up to now Radio Shack has not been of much assist.

Dave Gunzel's group of writers is expanding as new TRS-80 projects and equipment proliferate. He has put out a call for technical writers and is interested in hearing from anyone with experience in that field. Call him at (817) 390-3011.

CTR-80 surfaces in Fort Worth

--By JOHN STRONG

Have you been wondering what the CTR-80 is going to look like and what features it will have?

Rumors have been floating for a couple of months about a new cassette recorder to replace the CTR-41 in TRS-80 systems. Its features would make it better suited for computing, they said. But no one knew what those features would be. Now the new unit is in use at the Fort Worth Radio Shack Computer Center, where anyone can look at it.

Essentially, the CTR-80 is a CTR-40 with tape counter added, plus some internal changes. (The CTR-40 is the same size and shape as the CTR-41, but the decorations and keys are substantially different and it has no footage counter.)

2 The CTR-80 looks exactly like a CTR-40 at first glance until you notice the added

3 footage counter.

Internal improvements include modifying the mode key switch circuits so the computer can start and stop the tape motor only in Play (Cload) or Record (Csave). The Rewind and Fast Forward keys will operate independently of whatever the TRS-80 may be doing. This makes it much easier to rewind tapes any time you wish.

Further, the ground loop problem has been solved in the TRS-80. The fact that recently-delivered CTR-41s have been free from ground loop problems hints strongly that there was a mid-production-run ground loop fix on the CTR-41 at the Hong Kong factory.

The CTR-41 pipeline was turned off and the CTR-80 pipeline turned on recently so the switchover in recorders delivered with TRS-80 systems could occur in November. Unfortunately, the CTR-80 production rate hasn't built up as fast as hoped. To avoid holding up delivery of TRS-80 systems for lack of a recorder, Radio Shack is being forced to supply CTR-40s with a few systems.

If you got a CTR-80 with your new system, hooray! You got a free bonus. If you got a CTR-41 — well, you got what you ordered — and it is a darn good recorder. You can always buy a CTR-80 when they become available independently of computer systems — probably not soon.

If you got a CTR-40, that is unfortunate, but no big disaster. The CTR-40 will do everything a CTR-41 will do except permit you to find programs quickly on a multi-program cassette, via the footage counter. A lot of users never use cassettes that way anyhow — putting just one program on a tape and sometimes following that up with a second protection recording of the same program.

When buying a computer, you may be able to persuade your Radio Shack dealer to take your new CTR-40, sell it to someone else, and replace it with a CTR-80 when he gets them in stock eventually. One can do a lot of learning on a CTR-80 without a cassette recorder, although you will certainly need one eventually.

Here's source of tapes, disks

Jeff Lasman's PRACTICAL APPLICATIONS is now offering low-cost TRS-80 compatible mini-diskettes and program-tape. C-10 Tapes with leaders, sell for $1.25 in single lots to $1.00 in quantity, and are quaranteed.

Fully guaranteed and certified, mini-diskettes are either Memorex or Verbatum (no choice), the same disks Lasman uses for program development. Selling price is $3.50 in lots of one; $3.00 in lots of ten.

C-10 cassettes and mini-diskettes for the TRS-80 are available directly from Jeff Lasman, Box 4139, Foster City, CA 94404 (California residents please add proper sale tax) Minimum order is $10, or add $1.00 for postage and handling.

Publication schedule?

I received my first issue of "TRS-80 Computing" in August and I was very pleased with its content.

But now the first week in October is drawing to a close and I still haven't received the September issue. What's the problem?

I was led to believe that this would be a monthly publication and your first issue was so good that it whet my appetite for more of the same—I'm particularly interested in the Ledger/Journal program you advertised to appear in the next issue. I am impatiently waiting.

—EDWIN SCHWARTZ, 410 Pinewood ln., Los Gatos CA 95030

(Let's make it monthly from now on. How about everyone sending lots of editorial contributions. Those who have, and do not see them in this issue, expect them in the next. Thanks, editor.)

Write on both sides of disk

From some experimentation I have discovered how to write on both sides of a diskette using a standard RS disk drive. All that is necessary is to cut another write protect slot on the opposite side of the diskette and another sector slot in the diskette case opposite the present sector slot. A full sized diagram on the other side will show exactly what I mean.

—BRUCE M. TAYLOR
118 So. Mill St.
Pryor, OK

PS:
I ordered a second disk drive from Apparat Inc. Denver, Co.—Delivery was good and the product works great.

TRS-80 COMPUTING

BOX 158
SAN LUIS REY CA 92068

Nonprofit Org.
U.S. POSTAGE
PAID
San Luis Rey CA 92068
PERMIT 4