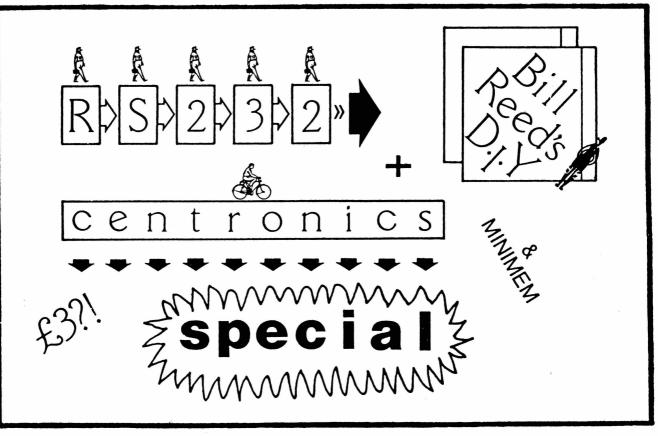


Volume 4, Issue 6

November 1st., 1987



Formerly OXON TI USERS

R PUBLICATION
OF THE
INTERNATIONAL
TO USER GROUP



PETER G. Q. BROOKS 96 BANBURY ROAD OXFORD OX2 6JT

OXFORD 510822

| Index to articles | Page |
|--|------|
| READ THIS FIRST! | 3 |
| Explanations, errors, apologies, excitement, and defence | |
| THE FORTH VERSION OF "ON X GOSUB" | в |
| JOHN ROE shows what you can do in CASEOF | |
| INTERFACE SPECIAL | 11 |
| Finally, after almost a year, BILL REED's articles are output | |
| DIAGRAMS | 15 |
| Pull-out A4 diagrams to go with Bill's article above | |
| LETTERS | 22 |
| If Readers write, Ol' Baldie will try to reply | |
| ROY MARCHANT scratches Speech Synthesis | 26 |
| | |
| EDDY TURNER's tale of Woe, with (almost) a Happy Ending | 27 |
| | |
| Lots to look at, food for thought, and things for sale | 29 |
| | |
| NEW HORIZONS RAMDISK ADVERTISEMENT | 31 |
| | |
| COMING SOON: TWO-DIMENSIONAL CELLULAR AUTOMATA, FIBONACCI SERIES YOU EVER WANTED TO KNOW ABOUT DISK DRIVES BUT KNEW BETTER THAN THE GODDARD'S DIY EXPANSION PROJECT, AND LOTS MORE | |

READ THIS FIRST

Apologies all round for the tardiness of the previous issue - due to a number of circumstances beyond my control which had a cumulative effect. This issue may well be hit by the same problems - I hope not.

The MECHATRONIC ads in the last issue were produced with blood, sweat, and tears. Not to mention some £30 worth of photography (which failed because of the predominance of black in the product colouring), 400 miles of damp, cold, but overheating, motorcycle travel, and umpteen telephone calls, etc., etc. Thanks are due to GORDON PITT, NEVILLE BOSWORTH (for not commenting on my professional photographic ability), and TREVOR DAVIES for his personal messenger service. In the end I had to resort to pen and ink. The simple solution is sometimes the best...

A few errors still made it through the Brooks Eagle Eye Proofreading System. On page 10 in V4.5, I erroneously labelled a short example with a variable called LOOP. It should have been LOOP1, and the later listing did have the correct name, but it might have confused those readers not used to my regular failings!

On page 16, I stated that other controllers increase access time to 3 ms but I should also have said that most drives can only handle 5 ms.

On page 31, I created a new word: Mong. Quite how it escaped detection, I cannot say. Mongs are slippery individuals, almost as bad as all the other pytophragical eras put gotether. If you see what I mean.

The tutored will recognise among among the garbage...

This issue should see at last BILL REED's articles on building your own RS232 and CENTRONICS interfaces, operating out of the joystick port and using MiniMemory to execute the operating system. My apologies to Bill for the delay in getting his article into print.

Elsewhere in this issue should be an advert for the NEW HORIZONS RAMDISK as promised. Further information appeared in TI-LINES volumes 2 and 3.

An alternative operating system, designed by MIAMI USERS, called V6.3, is available through GORDON PITT of the WEST MIDLANDS TI USERS. This is not a part of the New Horizons package deal, but is recommended if you wish to make the most of your TI. It is different from the O/S provided by New Horizons, and to date is not offered officially by them, but they do not disapprove of its use. Demonstrations have already been given at previous Bloxwich Workshops, and further demonstrations will be given similarly, probably early next year. Watch TI-LINES for details!

I recently received the latest version of FWEB (V4.0) and as usual, it is worth getting excited about. The ShowDirectory command in the FWEB Editor has been enhanced, allowing, amongst other things, a printout of the directory. What's more, if you press the usual "=" to get FWEB to determine the possible environment of program format files, THAT data is included in the printout.

The printout can be made to any valid output device - including a disk file, which is used in APPEND mode so that multiple directories can be compiled under the same output filename - and the layout of the printed directory is sufficiently pleasing to the eye that I have begun to print directories of my "static" disks (those whose contents don't change) and compile a folder of them.

As you will see from the two examples which I reproduce here, the layout is not wasteful of paper like so many others: a twin column approach has been used to make most efficient use of the paper. (I hate the way so many cataloguing programs follow the Disk Manager I/II approach and stick to the left hand side of the paper).

There is also a configuration program now to aid in setting up those variable items (like boot drive, colour selections, output filename, and so on) although Atrax Robustus notwithstanding, under some circumstances you can re-enter the FWEB Editor with the E/A default of white on blue, which I find a bit of a nuisance (but only a bit).

Here are two examples of printed directories to whet your appetites:

| DSK1 FW | EB4/0 | | | | | | ě | | | |
|------------|-------|---------|-----|------|---|------------|--------|---------|-----|----|
| Filecount | = 37 | Sect | ors | Used | = | 686 | Availa | able : | = ; | 34 |
| Filename | Size | Type | Rec | Þ | | Filename | Size | Type | Rec | þ |
| -READ-ME | 28 | Dis/Var | os | | | FWDOC/TIWR | 23 | Dis/Var | 80 | |
| AS | 33 | Program | | | | FWDOC/UTIL | . 65 | Dis/Var | 80 | |
| AT | 22 | Program | | | | FWSAVE | € | Dis/Fix | 80 | |
| C99PFI;O | 2 | Dis/Fix | 80 | | | LDFW | 9 | Dis/Fix | BO | |
| CHARA1 | 5 | Program | | | | LDSR/S | 4 | Dis/Var | 80 | |
| CHARA2 | 5 | Program | | | | LGEN/S | 8 · | Dis/Var | 80 | |
| CO | 3 | Program | | | | LL | 10 | Program | | |
| CONFIG | 40 | Program | | | | LOAD | 32 | Program | | |
| CTBRAM | 8 | Dis/Fix | 80 | | | MD | 2 | Program | | |
| DB | | Program | | | | MG | | Program | | |
| Db | | Program | | | | MH | | Program | | |
| EA | | Program | | | | ØD | 10 | Program | | |
| ED | | Program | | | | SAVIT | _ | Dis/Var | 0.8 | |
| EE | | Program | | | | SL | 10 | Program | | |
| FMSAVE | | Dis/Fix | сB | | | UL | | Program | | |
| FO | | Program | | | | UTIL1 | 33 | Program | | |
| FP | | Program | | | | XB4THLD | 2 | marpord | | |
| FWDOC/EASM | | Dis/Var | 80 | | | | | | | |
| FWDOC/LOAD | | Dis/Var | 08 | | | | | | | |
| FWDOC/REPT | 49 | Dis/Var | 80 | | | | | | | |

| Filecount | = 37 | Sect | tors | Used | = | 686 | Avail | able | = | 34 |
|------------|------|---------|------|------|---|------------|-------|---------|-----|--------|
| Filename | Size | Type | Rec | ρ | | Filename | Size | Type | Rec | p |
| -READ-ME | 28 | Dis/Var | 80 | | | FWDOC/TIWR | 23 | Dis/Var | 80 | _ |
| AS | 33 | Program | EA | | | FWDOC/UTIL | | Dis/Var | 80 | |
| AT | | Program | EA | | | FWSAVE | | Dis/Fix | | |
| C99PF1;0 | 2 | Dis/Fix | 80 | | | LDFW | | Dis/Fix | | |
| CHARA1 | 5 | Program | | | | LDSR/S | 4 | Dis/Var | | |
| CHARA2 | ຣ | Program | | | | LGEN/S | 8 | Dis/Var | 80 | |
| CO | | Program | EA | | | LL | 10 | Program | EA | |
| CONFIG | 40 | Program | | | | LOAD | 32 | Program | BX | |
| CTBRAM | | Dis/Fix | 80 | | | MD | 2 | Program | EA | |
| DB | 2 | Program | EA | | | MG | 33 | Program | EA | |
| DP | | Program | EA | | | MH | 30 | Program | EA | |
| EA | | Program | EA | | | QD | 10 | Program | EA | |
| ED | | Program | EA | | | SAVIT | 2 | Dis/Var | 80 | |
| EE | | Program | EA | | | SL | 10 | Program | EA | |
| FMSAVE | | Dis/Fix | 80 | | | UL | 4 | Program | EA | |
| FO | | Program | EA | | | UTIL1 | | Program | | |
| FP | | Program | EA | | | XB4THLD | 2 | Program | EA | |
| FWDOC/EASM | | Dis/Var | 80 | | | | | | | |
| FWDOC/LOAD | | Dis/Var | 80 | | | | | | | |
| FWDOC/REPT | 49 | Dis/Var | 80 | | | | | | | |
| | | | | | | | | | | |

I haven't shown any here, but fractured files (identified with an "*") are also marked on printouts.

Finally, (not really finally - I hope that some enterprising author will put pen to paper and write an in-depth review of FWEB 4) a ruler has been incorporated in the Editor so that you always know where you are on screen. It sits in place of the old End of File legend which used to mark the end of a document.

In the words of a famous TV personality, Di'll give it foive (and thus show my age...).

A bit of extra-TI news: TDK have come up with a rewritable laser system, called magneto-optical disk technology. It can provide 1000Mb storage on a 12" video disk, but I haven't seen prices yet. Outside our range, of course, but then there was a time that 80 track double sided, double density drives were outside our range, too.

I have another plea for someone to get in touch with me: somebody asked me to obtain DADDY'S HOT ROD, FUN PAC, and PEARL DIVER when they applied for ITUG membership. However, their request was written on a slip of paper without their name or address, and so far I have been unable to locate details (which they obviously supplied, but which have become separated from their order). If you could get in touch with me soon, I would be grateful.

I have been asked to repeat the details concerning the ROMOX cartridges offered through ITUG. The original information was presented in V3.12 TI-LINES, for those who have that issue.

Basically, the ROMOX module is a cartridge containing an EPROM (2764) which can be programmed with an EPROM PROGRAMMER. Such programmers are expensive to buy and not worth the outlay unless you intend programming large numbers of cartridges. Unlike standard cartridges, which contain preprogrammed ROMS, the ROMOX cartridge can be reprogrammed over and over again.

The advantage is that having bought a ROMOX cartridge (and having had it programmed) if you later decide that you no longer need/want the program currently residing within it, well, you can have it reprogrammed with something else, instead of having to mothball or resell at a fraction of the purchase price. This facility is of greater use to unexpanded system owners, since expanded system owners have disk systems to carry all the software they may need.

If the kids get bored with GrannySmacker II and want the latest Alien Mothers-in-law, all it will cost you is the fee charged by the ITUGer offering the service and post and packing.

Originally I suggested that interested parties might buy several ROMOX and keep their favourite programs on some while having others reprogrammed, but in the event I am going to have to restrict the number of ROMOX to each subscriber. The maximum will be FOUR unless circumstances force me to change that ceiling.

The one-off price will be £5 as indicated in V4.5 TI-LINES, but the price will include a program chosen from a list being compiled now. This makes the price charged for the RDMOX around £2, with a £3 fee for programming. Post and packing will be inclusive in this price if obtained through ITUG. Other sources may vary.

Here's one for the record books. I have been working with a DV80 file which generates I/O errors with LINPUT, but of an order not seen by me outside a module crash. How do you decode an I/O error of 2297 or 571 when the largest valid (supposedly) number is 99 ?

Another MiniMemory record: IAN JAMES' MiniMemory is still going strong with a battery apparently fitted in week 51 of 1982.

October seems to have been the month when things went wrong not just for me but for a number of others also. I have spent more time than usual, rescuing lost disk files, repairing disk drives, and trying to diagnose faults over the phone (an increasingly difficult task).

As a result of my ministrations, I am now ready to offer a disk and disk drive repair service. Details on the Notice Board. Prices are as competitive as I can make them, but do not include the cost of post, packing, and insurance.

During the last few months, it has come to my attention that certain authors, writing in certain newsletters emanating from certain parts of the world, have been wielding sharp knives in the direction of TONY and WILL McGOVERN, authors of the FWEB package.

I have previously adopted a policy whereby I refrain from commenting in such situations (despite fairly strenuous opinion expressed forcefully in my ear on occasions) but just this once I feel justified in coming down off the editorial fence.

I don't know Tony or Will from Adam, but I do know a lax overseas editor when I see one. Whoever sanctioned the publication of the villifying letters, containing unjustifiable personal attacks on the McGoverns, needs something more than their head examining.

In this case, the usual editorial get-out of "the opinions expressed in this newsletter are not necessarily those of the..." is not sufficient protection.

From time to time, I receive written and verbal criticisms of various TI Users in various parts of the world (but mostly the UK, naturally), the content of which would keep a solicitor in comfort for life. They represent extreme opinions, by and large, but they stay with me - they NEVER get an airing in TI-LINES, nor should they.

It is impossible to undo the damage inflicted on Tony and Will, and no amount of openly-expressed support for them can do anything to lessen the hurt they must have felt. In fact, so quietly did they accept the onslaught, I knew nothing until reading another newsletter which quoted Tony's (mild) response and the expression of support by the editor of that newsletter.

Even though it must be a drop in the ocean, and even though it must be little comfort to them, I am bound to express MY support for the pair from Australia, spiders notwithstanding, and to call for a written apology from the group who was stupid enough to set the ball rolling in the first place. Needless to say, ITUG has no connection with that group, although we were in the process of attempting to establish some kind of association. That association will now not take place.

What I find most puzzling is the fact that the group responsible have previously enjoyed an enviable reputation in the TI sphere, and I fail to understand how they could have condoned such an unpardonable action.

JOHN BUTCHER would like to obtain HCM 5.3 and 5.4 if anyone has them for sale, and is also looking for any data sheets on the SHUGART SA455 with information on how to set the links. Contact him on 0293 884204.

I have been playing with FWEB's Editor and have concluded that it is possible to use it as a relational database. The same facility exists in TI-Writer, and I will write about it in due course. It is simple to implement, and fairly slow (relying on the brevity of the related files to provide speedy access) so it might benefit from being run from a RAMDISK. It's nothing special, so don't hold your breath...

THE FORTH VERSION OF "ON X GOSUB"

By JOHN ROE

TI-Forth has quite a few additional features over and above the basic FIG-Forth. Some of these are, of course, necessary to provide for machine-specific functions such as File Handling or Graphics. There are other extras which in no way seem to be machine-specific, and I note in particular a set of words: "CASE", "DF", "ENDOF", and "ENDCASE", which between them make up the "CASE...OF" construct.

As far as I can see, this is unique to TI-Forth as I have not so far come across it in any of the various books on Forth which I have read.

I did come across the concept about three or four years ago when reading up on Pascal, but at the time I didn't have the hardware (and still do not have) to use Pascal, so couldn't try it out in practice.

Although I have read quite a lot about TI-Forth over the last few years, I have not so far seen the CASE...OF construct used in any programs nor have I seen it referred to in any way. Since it does essentially the same job as the ON X GOSUB routine in BASIC, readers of TI-LINES may be interested to learn how it works.

The TI-Forth manual does not give any examples of how to use the construct, nor is there any other comment apart from the brief notes in the glossary under the separate words. However, it appears that the contruct makes use of two numbers on the stack, which, to avoid confusion, I will refer to hereafter as n1 and n2.

The first, n1, is either entered on to the stack from the keyboard before using the word in which the construct appears, or it is left on top of the stack by some preceding word.

The second, n2, appears before the word OF in any word using the construct and provides for several alternate courses, of action. A small example will serve to make this clear:

: CHOICE CASE 1 OF ." ONE" ENDOF 2 OF ." TWO" ENDOF 3 OF ." THREE" ENDOF ENDOASE :

{Those familiar with C will recognise this construct. PB}

What happens in this case is that when one of the three numbers, 1, 2, or 3, is entered on the stack as n1 followed by CHOICE, the word OF places each of the values of n2 (1, 2, or 3) on the stack in turn, and this number is compared with n1 until a value for n2 is found which is

equal to m1.

The action appropriate to that choice, which appears between OF and ENDOF, is then performed.

For example, if 2 is placed on the stack and the word CHOICE entered, then "TWO" is displayed on the screen.

I have a program on disk which performs various calendar functions. It is a lengthy program and in any case since it was copied from one of my Forth instruction books it will almost certainly be subject to copyright restrictions. For these reasons I am not reproducing it here. I did however use the CASE...OF construct to add a screen to produce a menu of choices and to select one or other of these. This is of course all my own work so I don't have to worry about someone else's copyright.

The screen is reproduced below, but of necessity I have had to amend the choices given between OF and ENDOF. The words called in the working version are at the apex of a hierarchy of words themselves defined in the program, and so cannot be called into action without loading the whole program.

I have therefore amended the choices so as to display the names on screen, rather than call them into action:

```
scr# 99
O ( MENU AND CHOICES )
1 : CHOICE CASE 1 OF 14 10 GOTOXY ." JANUARYLIST" ENDOF
                2 OF 15 10 GOTOXY ." DAYSLEFT" ENDOF
3
                3 OF 17 10 GOTOXY ." YEAR" ENDOF
                4 OF 17 10 GOTOXY ." MONTH" ENDOF
5
           ENDCASE ;
6 : CHOOSE PAGE
    12 1 GOTOXY . " MENU"
    4 ROW . " 1. Day on which January 1st falls."
8
    6 ROW . " 2. Number of days left in the year."
9
   8 ROW . " 3. Calendar for whole of the year."
11 10 ROW . " 4. Calendar for one month only. " CR CR
12 KEY 48 - DUP 1- 4 UK IF CHOICE KEY IF MYSELF ENDIF ELSE PAGE
13 7 4 GOTOXY ." Don't mess about, now!" 8 6 GOTOXY ." Choose 1,
   2, 3 or 4!" 6 11 GOTOXY ." Press space bar to try again." CR CR
  DROP KEY 32 = IF MYSELF ENDIF ENDIF; CHOOSE
```

Some comment is required on several points. In lines 8 to 11 I have used the word ROW. This is predefined in my system in screen #3 to avoid the laborious entry of a number of GOTOXY lines where the displayed lines will all start in the same column. Its definition is as follows:

: ROW O SWAP GOTOXY ;

Some value other than 0 can be substituted if you want to start on a different column.

In line 12, "KEY 48 -" produces one of the numbers 1, 2, 3, or 4 on the stack. The words "DUP 1- 4 U(" perform a range checking function to ensure that only the keys 1, 2, 3, or 4 can produce a valid response. The use of "U(" as a comparison is discussed in more detail below, as is the use of "MYSELF" in lines 12 and 15. In line 13 the rather joking "error message" can be amended if preferred to "Out of range". In line 15 the "DROP" is necessary to get rid of the incorrect choice still on the stack.

The use of "U(" for comparison involves a rather subtle point in the way Forth (and Assembly language) deals with negative numbers. A single precision number has a range from -32768 to +32767 when signed. An unsigned single precision number on the other hand has a range 0 to 65535, so that the range of negative numbers -1 to -32768 appear to the computer to be exactly the same (in binary) as the unsigned range 32768 to 65535. If "U(" is used to compare two numbers this is the only comparison necessary since it automatically treats negative numbers as large positive numbers which are necessarily "greater than" the number being compared.

However, since 0 is treated as a positive number, it is necessary first of all to make the permissible range begin with a 0 by including "1-" after "DUP" and then testing whether the number is "less than" 4.

As Peter Brooks has explained in a recent issue of TI-LINES, it is possible to have a recursive routine in BASIC, but it is an awfully messy business and I think that the same end result could be more easily achieved (in BASIC) by other means.

By contrast, all that is necessary in Forth to make a routine recursive is to include the word "MYSELF" in the appropriate place in a definition as in lines 12 and 15 above. Using this recursive mechanism, when one action has been completed you can get back to the menu for another choice, or if you have made a boob and pressed a wrong key, you can get back to the menu and have another go.

This is, of course, very much of a one-off job and there are ways of making it more general, particularly in a program where different menus would be called for display at different stages and there is some overlapping of choices between menus.

By using "(BUILDS...DOES)" to create a new defining word to define string arrays, a message could be stored in each array and called for use in a particular menu.

In writing the above I make no pretences at being a Forth expert; I am very much still in the learning stage. Quite possibly my way of doing this particular job could well be improved upon, and I would welcome any constructive criticism along those lines.

(Perhaps John might be prevailed upon to discuss his ideas for a less messy recursive facility in BASIC; either way, Forth buffs PLEASE take up John's invitation. You can contact him at 7 Harbury Close, Matchborough West, REDDITCH, Worcs., B98 OEF. Please include return postage for a reply if you decide to write with queries. PB}

INTERFACE SPECIAL: SERIAL + PARALLEL

By BILL REED

PART ONE: THE DIY SERIAL INTERFACE

Peter, your comments in the past about not leaving the constructional items entirely to our overseas brethren have stimulated me to put some details together for you with respect to an RS232 facility which uses the joystick port.

It is quite possible that such a simple idea has already been published earlier, in which case just ditch it in the nearest wastebin. If not, this provides an asynchronous output facility for talking to a printer (or a modem) for about £3.00 all in.

{A similar interface has indeed already been published in the States, called JOYTALK or JOYTALKER, but I gather that Bill's interface does not require an external 9 volt power supply. PB}

Some time ago, Tandy were offering their TP10 thermal printer at less than half price in some of their shops, and while it is only a 32 column Job and the paper is getting a bit difficult to find sometimes, it is perfectly adequate for screen dumps, listings, etc., and good value at forty-odd quid. It's very reliable too, as I have given mine some stick over the last year and a bit, without even a single poor character being produced.

Not wishing to connect to the expansion port (I have other plans for that), nor part with excessive amounts of the folding stuff, I knocked up the circuit design and machine code driver given here, to take advantage of an otherwise redundant joystick port, using a handful of bits from my junk box.

As you will see from the details, both the circuit and the code are orientated to the TP10, which runs at 600 baud, and does not need a pull down to a negative rail on its DATA input. You will find however that a wide range of baud rates can be catered for (by changing the timing constants in the code), and the two spare inverter gates in the 74L00 IC can be wired in line if a particular printer/modem has TRUE versions of DATA and BUSY signals, rather than the COMPLEMENT versions demanded by the TP10.

The DATA protocol is 1 start bit, 8 data bits, no parity, and 1 stop bit. Strictly speaking, the TP10 manual says 2 stop bits should be used but this is only because of a slight delay in the trailing edge of the BUSY signal, turning off. It just means that if you present it with a new character before the BUSY line says you should, then the start bit of the new character would find the door still shut in its face, for a few microseconds.

As the BUSY signal is examined by the code for handshake purposes and can not therefore send a new character until the printer requests it, there is no need for a second stop bit in practice.

Should any application need a second stop bit, however, (or as many as you like), just bung another BL @SH instruction in between lines 37 and 38 of the code.

In order to ensure that the TI-99 doesn't get stuck in the driver routine if the printer/modem is not plugged in (or becomes faulty), there is a watchdog time-out facility in the code at T2, which exits after 50 seconds with an error message to the calling program (I/O error code 12) in such an event.

Use of the facility is straightforward. Just replace each BASIC PRINT# line in the calling program with a new line assigning the data to be output to a temporary string variable (say MES\$), and follow this with an additional CALL LINK("PC1", (MES\$)) line, to invoke the driver routine.

NOTE:

As the routine expects a string variable to be passed to it, it will be necessary to convert NUMERIC items into their ASCII string versions, using the STR\$() function.

It will also be necessary to insert the appropriate carriage return and line feed characters, etc., into the string, in place of BASIC controls such as the colons etc., that would normally appear in a PRINT statement and cause the computer to output such characters to the standard RS232 channel.

EXAMPLE:

replace: PRINT #1: "moved by"; -DIF; "bytes":

with: MESs="moved by "&STR\$(-DIF)&" bytes"&CHR\$(26)&CHR\$(10) CALL LINK("PC1", (MES\$))

note that CHR\$(26) and CHR\$(10) are the Tandy TP10 printer codes for Carriage Return (CR) and Line Feed (LF) respectively.

Some printers (including the TP10) also accept a single character (e.g. CHR\$(13) as combined CR and LF command, in place of the two separate characters.

I trust that all of this is clear enough (!), but if not, I am always willing to help anyone with further advice (for example, on how to interface other printers, or how to LIST a program in VDP RAM to a printer via MINIMEM etc.).

{Readers should WRITE with their specific enquiries to Bill care of ITUG at the Oxford address, and I will forward any letters to him. PB}

"PC1" - 600 BAUD PRINTER DRIVER ROUTINE FOR MINIMEMORY (MM) .

After loading, ensure name and START address is entered in REF/DEF table

Invoke from BASIC program with:-

CALL LINK("PC1", (MES\$))

Note: the brackets inform BASIC that the variable MES\$ does not need to be returned to the calling program, and return space is not needed.

The routine given over the page outputs a string of characters, passed from a calling program via a string variable (MES\$).

The routine outputs the string to the number 1 joystick port, character by character, on pin 7, and also monitors pin 5 as the printer BUSY line (low = BUSY), for handshake purposes.

Characters are output in standard RS232 serial data format (8 data bits, no parity) with one start bit and one stop bit, and buffered via a small interface circuit to match the printer (or modem).

The circuit shown on the previous page fits neatly into a matchbox or old 35mm film container, and interfaces to a Tandy TP10 thermal printer or similar device, and picks up what negligible power it needs from the printer.

In the case of the TP10, this requires that the printer be opened up and +5 volts picked up from any convenient point, and wired to the spare pin (1) of the printer interface socket.

The routine accepts a string length as defined in line 11 (in this case 128 characters) and the baud rate is determined by the constant set into R6 in subroutines SL and SH.

"PC1" is listed on this page and overleaf, and the program assumes it invoked by CALL LINK() with workspace pointer at)7088.

LISTING

3

STRREF EQU)604C ERR EQU)6050

1 START MOV R11, R10

CLR R3

2 CLR RO

Save BASIC return address

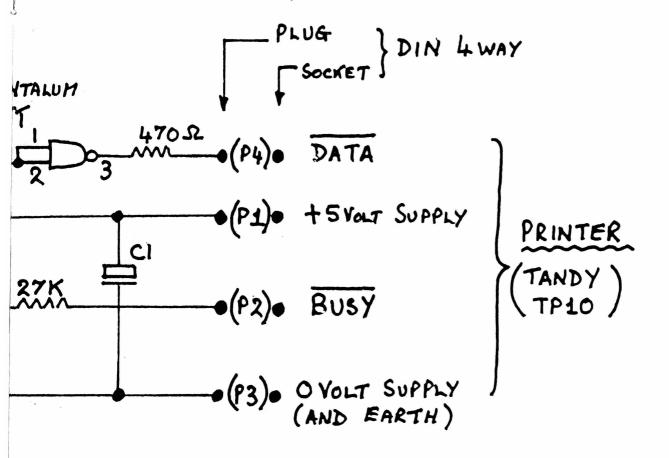
STRREF code (DATA is not an array)

Clear character store

| 4 | | CLR R4 | Joystick 1 set HIGH code |
|----|-----------|-------------------|---|
| 5 | | CLR R9 | Clear character count location |
| ε | | CLR R13 | Set buffer index to byte 0 |
| 7 | | LI R1,>0001 | Parameter No. for STRREF |
| 8 | | LI R2, B0 | Pointer to buffer area |
| 9 | | LI R5, > 0606 | Joystick 1 select LOW code |
| 10 | | LI R12, > 0024 | CRU address of joystick logic |
| 11 | | LI R8,>8000 | Length of message buffer |
| 12 | | MOVB R8, @BO | Insert into 1st byte of buffer |
| 13 | | BLWP @STRREF | Get MES\$ from calling program |
| 14 | | MOVB @BO, R9 | Get length of MES\$ from buffer |
| 15 | | SWPB R9 | Move it down to LS byte of R9 |
| 16 | T1 | CLR R3 | Clear the character store |
| 17 | | INC R13 | Point to character in buffer |
| 18 | | MOVB @BO(R13), R3 | Get character from buffer |
| 19 | | SWPB R3 | Move it down to LS byte of R3 |
| 20 | | LI R14, >0020 | 1 Set BUSY watchdog to 50 seconds |
| 21 | | CLR R15 |] (time-out count) |
| 22 | T2 | DEC RIS | |
| 23 | 12 | JNE T3 | Decrement LS byte of watchdog |
| | | | Skip unless LS count is zero, |
| 24 | | DEC R14 | otherwise decrement MS count |
| 25 | ~~ | JEQ T9 | Go to error routine if BUSY time-out, |
| 26 | TZ | TB -14 | otherwise sample BUSY input |
| 27 | | JNE T2 | Loop back if BUSY to decrement count |
| 28 | | LI R7,>0008 | Set character SHIFT count in R7 |
| 29 | | BL @SL | Output a START bit to Joystick |
| 30 | T4 | SRL R3,1 | Sample a bit from the character |
| 31 | | JOC TS | Skip forward to T5 if it is a 1, |
| 32 | | BL @SL | otherwise output a LDW, |
| 33 | | JMP TE | and skip forward to T6 |
| 34 | TS | BL @SH | Output a HIGH to joystick port |
| 35 | TE | DEC R7 | Decrement the character SHIFT count |
| 36 | | JNE T4 | Rpt from T4 until all 8 bits shifted out, |
| 37 | | BL @SH | and then output a STOP bit (HIGH) |
| 38 | | DEC R9 | Decrement the string character count |
| 39 | | JNE T1 | Repeat output sequence for each character |
| 40 | EX | MOV R10, R11 | Replace BASIC return address in R11 |
| 41 | 4. | CLR RO |] Clear BASIC "error" location, |
| 42 | | MOVB RO, @> 8370 | I ready for returning, |
| 43 | | B *R11 | and return to calling program |
| 44 | SL | LI RE,>0082 |] |
| 45 | | LDCR R5,3 |] Subroutine to output a LOW level |
| 46 | T7 | DEC RE |] to joystick 1 |
| 47 | | JNE T7 |] for 1.666 milliseconds |
| 48 | | B *R11 | |
| 49 | SH | LI R6,)0082 | |
| 50 | | LDCR R4,3 | I Subroutine to output a HIGH level |
| 51 | ВТ | DEC RE |] to joystick 1 |
| 52 | | JNE TB |] for 1.666 milliseconds |
| 53 | | B *R11 |] |
| 54 | T9 | LI RO, > 1200 | Report "CAN'T CONTINUE" message to |
| 55 | - | BLWP GERR |] calling program |
| 56 | | JMP EX | Error trap for spurious RETURN |
| 57 | во | BSS 128 | 128 byte buffer space reservation |
| -, | _ • | END | The chile space leservacion |
| | | | |

{Once again, my apologies to Bill for the delay in publishing this. PB}

ORT SERIAL DATA INTERFACE

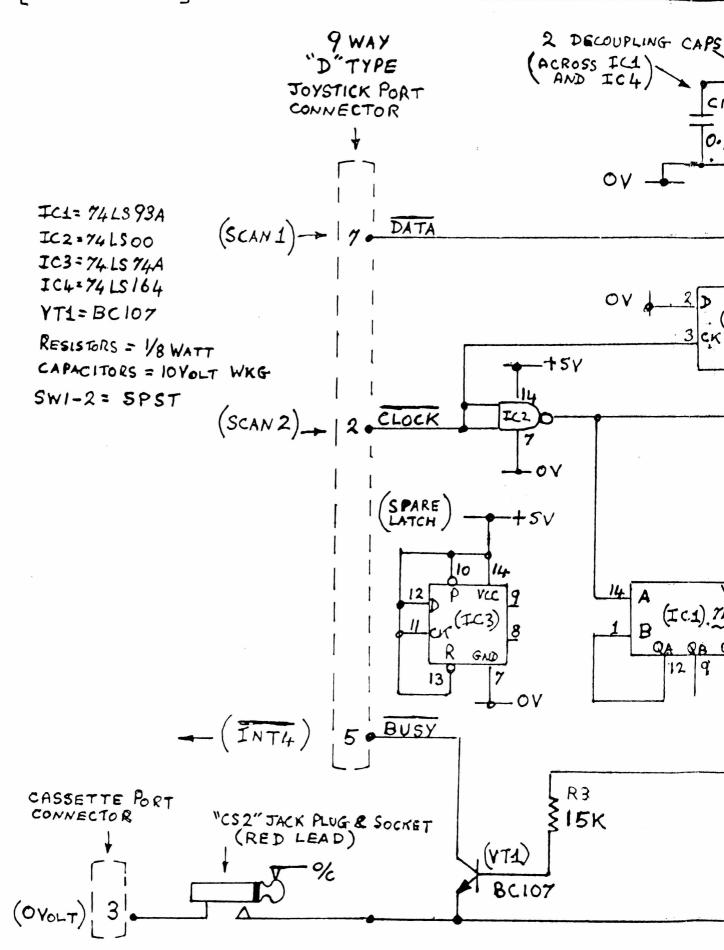


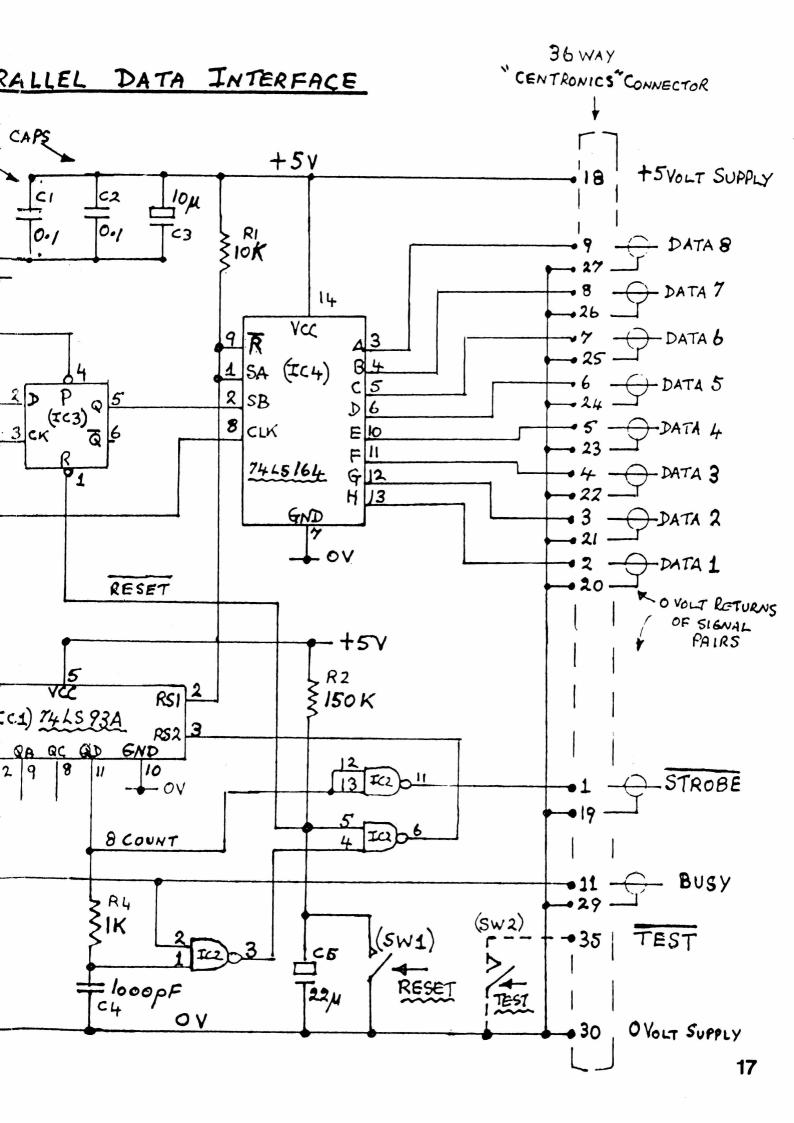
"C92" JAKK PLUG (RED lead)

olt line, is returned to the lo there is no Ovolt connection by of the Joystick port fins.

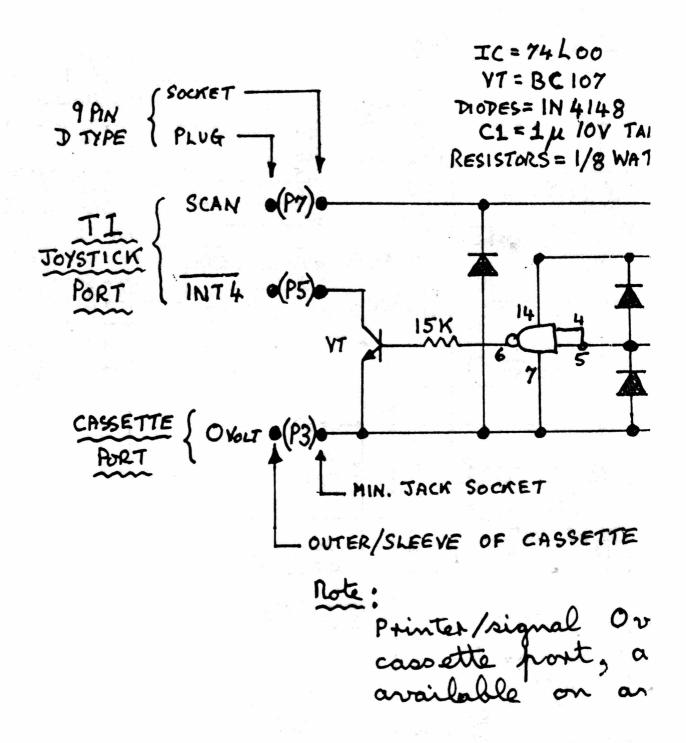
(W.REED 1986)

JOYSTICK PORT" PARALL





"JOYSTICK F



Following on from my DIY RS232 above is my Parallel Data (Centronics) Printer Driver circuit, which again I designed to be driven via the joystick port, and which I have used with my Seikosha GP500A matrix printer for some months now without any problems.

The circuit is fairly simple, uses only 4 ICs, and fits neatly in series $\$ with the printer ribbon cable, on a 4" \times 2" bit of veroboard located in a small plastic box purchased from Tandy.

The circuit picks up its +5 volt power supply (about 50 mA) from the printer, and operates as follows:

- (a) On power-up, C5 and R2 generate a RESET pulse to ensure that IC1 and IC3 take up a normal state.
- (b) Data is output in serial form (a byte at a time) using Joystick 1 and 2 SCAN lines for that purpose. SCAN 1 is pulsed low (DATA) for each bit in the byte which is a "1", to preset the FLIP/FLOP (IC3). SCAN 2 is pulsed low (CLOCK) for each data bit (ones or zeros) in the byte, and therefore shifts in each data bit into the 8 bit serial-to-parallel shift register (IC4) via the FLIP/FLOP (IC3).

The front edge of the CLOCK pulse shifts the FLIP/FLOP data into the shift register and the back edge of the CLOCK then resets the FLIP/FLOP to the "zero" state, ready for the next data bit in the byte.

- (c) The BUSY line from the printer is sampled before printing starts, and has a software watchdog timer which times out after 50 seconds (and returns to the calling program with an error message) if the printer BUSIES lock up for some reason.
- (d) The CLOCK signal is also applied to an 8 bit counter (IC1), as each bit is clocked into the shift register, and when the 8th bit has been clocked into the shift register (all 8 bits are now available in parallel at the outputs of the shift register), the 8 COUNT is used as a STROBE signal to the PRINTER interface.
- (e) The printer should signify its acceptance of the 8 bit byte by bringing its BUSY line high, which in addition to inhibiting any more output from the TI-99/4A, also allows the 8 bit counter to be reset to zero, ready for the next sequence.

R4 and C4 ensure that at least 100 nanoseconds (ns) of reset is applied to IC1, in response to the printer BUSY line going active.

The sequence of events from the TI-99/4A therefore is as follows over-leaf:

```
+FOR ALL CHARACTERS IN THE PRINT BUFFER
LOOP1)
LOOP2)
             +IF PRINTER NOT BUSY
             +---THEN DO:
                         GET CHARACTER FROM BUFFER
                        +FOR EACH BIT IN CHARACTER
                            +IF BIT IS "1"
                            +---THEN DO:
                                          PRESET FLIP/FLOP & THEN CLOCK
                            +----ELSE DO: (BIT IS "O")
                                          DO NOT PRESET F/F, JUST CLOCK
                            +ENDIF
                        +NEXT BIT
               ---ELSE DO:
                         INCREMENT WATCHDOG TIMER
                         EXIT WITH ERROR IF TIME-DUT
                         OTHERWISE TRY FROM LOOP2 AGAIN
             +ENDIF
         +NEXT CHARACTER (LOOP1)
```

In common with my earlier RS232 printer driver circuit, driven also via the joystick port, you will note that the printer 0 volt rail is tied to the TI-99/4A 0 volt rail via the cassette port connector, as TI do not provide a 0 volt reference on the joystick port itself. For convenience I make this connection via small jack socket to the RED lead of the unused "CS2" cassette leads, rather than unplugging the cassette leads each time I use the printer.

The only other items of interest are SW1 and SW2 on the diagram, which are small push-button types.

SW1 is useful for generating a reset to the circuit (IC1 and IC4) when the computer has been turned DFF and DN again for some reason (but not the PRINTER), as this (or a splurge on the power rails) can cause the 8 bit counter to spuriously increment, and the printer STROBE will then be generated in the wrong place, with rubbish printed.

SW2 is an optional facility, for forcing the printer's SELF TEST line active (LOW), if the printer has such a facility, and you want to be able to check it out quickly at the press of a button, rather than generating character streams from the computer.

be printed. NOTE: the routine as written allows for a maximum of 128 characters to be printed per line, but changing the buffer length in R8 allows for greater or fewer numbers of characters to be handled.

ERR

STRREF

EQU > 6050

EQU) 604C

WORKSPACE =)70B8ENTRY MOV R11, R10 Save BASIC return address in R10 STRREF marker (data is not an array) CLR RO 2 Clear character storage location Joystick 1 and 2 "SET HIGH" code CLR R3 3 CLR R4 4 5 CLR R9 Clear character count location Set pointer to byte 0 of buffer BO CLR R13 E 7 LI R1, >0001 Parameter number for STRREF routine LI R2, B0 Pointer to buffer area BO 8 Joystick 1 "SET LDW" code Joystick 2 "SET LOW" code Joystick 1 & 2 SCAN logic, CRU address Def buffer length (dec 128) in MSBy of R8 LI R5, > 0606 9 LI R6, > 0707 10 11 LI R12, > 0024 LI R8,)8000 12 Insert buffer LENGTH into buffer byte 0 MOVB R8, @BO 13 BLWP @STRREF MOVB @BO, R9 Get data string from calling program 14 15 Get data string LENGTH from buffer byte O SWPB R9 Move it down to LS byte of R9 16 Clear OUTPUT CHARACTER storage location CLR R3 17 T1 INC R13 18 Point to next character in buffer MOVB @BO(R13), R3 Get character from buffer, hold in R3 19 Move it down to LS byte 20 SWPB R3 SWPB R3 LI R14,70020 (LS) set BUSY watchdog 21 CLR R15 (MS) for 50 second count 22 23 T2 DEC R15 Decrement the LS watchdog count JNE T3 24 Skip unless LS count is zero DEC R14 25 Otherwise decrement MS count Go to ERROR routine if watchdog timed out JEQ T7 26 27 TI TB -14 Otherwise sample BUSY input line JNE T2 Loop back to watchdog if still busy 28 Otherwise set 8 bit char shift cnt in R7 Sample a bit from the character 29 LI R7, >0008 30 T4 SRL R3, 1 Skip to T5 if it is a "1"
(J2 clock LOW) else output a "0" serial
(J2 clock HI) data bit to the I/F circuit JOC TS 31 LDCR R6,3 32 33 LDCR R4,3 and then skip forward 34 JMP TE (set I/F latch to "1") | loutput a "1" 35 TS LDCR R5,3 36 LDCR R4,3 (clock J2 LOW) 37 LDCR RE, 3 Ito I/F circuit LDCR R4,3 38 (----) 39 TE DEC R7 Decrement the character shift count 40 JNE T4 Rpt from T4 until all 8 bits shifted out 41 DEC R9 Decrement the string character count JNE T1 Rpt "O/P Char" sequence until end of str\$ 42 MOV R10, R11 Restore BASIC return address into R11 43 44 CLR RO Clear BASIC "ERROR" loc'n ready for ret'n 45 MOVB RO, @>8370 46 B *R11 and return to calling program 47 T7 LI RO, > 1200 Report "STUCK BUSIES" (I/O code 12) 48 BLWP GERR to calling program 49 BO BSS 128 128 byte buffer space reserved

LETTERS

A number of ITUGers have written with a variety of questions and comments, so rather than publish each letter in its entirety, I have extracted the relevant points, and responses to them appear below.

JOHN SEAGER asked in V4.5 if anyone knew how to get TI-Writer to find the actual page centre when centering text with the .CE command.

This enquiry elicited a reply from STEPHEN SHAW:

"A little birdie tells me that .CE does not work. {Stephen then gave some examples which involve folding the paper to determine whether .CE had performed correctly or not. PB}

For 80 columns live, that's an EVEN number of spaces, so how do the ODD and EVEN character words fit in ?

If you want to use .CE you MUST use .LM and .RM, because .CE will centre your text between those two settings. This may not be the middle of your paper, which could be off-centre in your printer.

.LM must be higher than zero. Column 1 in the Formatter is called 1; in the Editor it is 0 of course. That's due to two programs being written by two different people...

The TI-Writer manual does not state that margins have defaults. If they appear to be there when you print out with .FI and .AD, they are not available to .CE...

If instead of setting .LM to 0 (which has an odd offset) you fail to set .LM at all, the result is very offset (I make it about ten characters).

Try it. And when using the Formatter, try to remember to set .LM and .RM and remember column 1 is actually number 1."

Stephen also commented on the disk drive article in V4.5:

"The IBM disk format (original IBM design) allows for a variety of disk formats, including differing numbers of bytes per sector. TI used just DNE implementation of this. It is not the IBM PC format, just one implementation of a very flexible design. The header track contains details — and even though our system ignores much of the information in it, TI kept to the standard and put it on the track anyway."

IAN JAMES wrote with a number of comments and queries following on from the articles on machine code programs and disk drives. He has also come across a very good little book on disk systems called USING FLDPPY DISKS by R. A. BARDEN but doesn't recommend anyone to BUY it as it is only 100 A5 pages for £10.50.

could any of the old information "show through" even after the disk has been initialised by the TI ?

The answer requires a distinction between "wiping" a disk and initialising it.

The process of wiping simply removes the "links" between sector 1 and the subsequent sectors containing directory information. A more detailed explanation of disk directory structure has already been published (check PETER KILLICK's index) and it is too large to reproduce here.

The wipe leaves all other file information intact on the disk, which is why DM1000 is able to offer a "resurrection" facility provided you know the name of the file you wish to reinstate.

The process of initialisation (or formatting - not strictly the same) involves the physical rewriting of the magnetic domains on the disk surface. This is akin to rerecording on a tape recorder. In the case of audio tape, repeated rerecording can result in previous recordings "showing through", especially if (a) the original signal was strong, and (b) the erase head performs inefficiently.

As far as disk drives are concerned, these problems should not occur. Although a disk surface is coated with material similar to that used on audio tape, the recording mechanism is different.

It is worth pointing out that, under rare circumstances and with only a few packages, it is possible for disk information to be slightly damaged, resulting in a disk which produces a NOT INITIALISED error with Disk Manager I/II. This is usually because sector zero has been damaged by being accidentally rewritten, something which DM1000 does still from time to time. This does NOT mean that the disk is truly NOT INITIALISED, and with care and skill the disk can be reclaimed. I have performed just such surgery recently - although it took over 8 hours to sort out just two sectors!

If you use disks previously used on another computer, you can still re-initialise them for use with the TI, although personally I would be wary of such material, since, like secondhand crash helmets, you can never be entirely sure that they have been treated with respect!

Ian also asked if it was possible to clean the GROM port without having to dismantle the console. For practical reasons, the answer is no. The GROM port connector has to be dismantled and the foam inside cleaned and replenished with lubricant. This would be extremely difficult to perform without dismantling (I don't say impossible, because some smartalec will no doubt manage the manual dexterity necessary), but the act of dismantling is not that complex. It merely involves removal of the screws underneath the console, removal of the switch mechanism on the front, removal of the power supply board (two screws and one connector) and two screws and one connector to get the keyboard off. The numbers of screws may vary slightly. If ITUGers would find it useful, I will publish the procedure (with line drawings rather than photographs) in a future issue. Let me know if you would like to see such an article.

Various writers have asked about the problem of saving a cassette-based

program to disk, only to find that it cannot be reloaded from disk.

The usual trick is to use CALL FILES(), a subprogram which is ONLY available if you have a disk controller connected. The default is set at 3 (meaning enough space has been set aside to allow three files to be open simultaneously). If you CALL FILES(1) in the Immediate Mode, and then NEW to force the 4A to recalculate available memory, you should then be able to load the program.

In special cases, Extended BASIC programs which are over a certain size cannot be saved to disk in program format. Instead they go out in IV254 format, one which has yet to be fully investigated. I looked at it briefly a couple of years ago, when it seemed that IV254 had special properties, but I have had no time to follow it up, and no-one else has shown any interest.

Any attempt to load from disk and save to cassette will be fraught with difficulty, and really the 32K will need to be present in order to sort things out. Like resurrecting a lost disk file, there is no easy, straightforward answer; the mechanism leading to a solution may vary with every instance.

Another question which has cropped up is how to copy, onto disk, data saved to cassette by Music Maker. This is not an easy one, but in essence you will need (a) to determine the data format as stored on cassette and (b) actually be able to load from disk back into Music Maker. It seems logical to me (being unfamiliar with Music Maker) that if you can save to cassette and load from disk, you should be able to save to disk as well, thereby solving the problem.

However, the fact that the question arises suggests that, as with the MiniMemory module, you can load/save with cassettes, but only load with disks.

The result is: you must learn about programming in machine code, learn about accessing the cassette port using machine code, and learn how to write out to disk in machine code.

In short, you have an enormous task ahead of you...

One from the past: did I ever get the Forth copying program to work (published in V1.13 and V1.14) ? The answer is no, and I never had any feedback from anyone else who was successful.

Another enquiry concerned the saving of only part of an Extended BASIC program. It IS possible to save part of a program, but that involves deleting all the lines you DON'T want to save - rather time-consuming.

Another approach is to list the program to disk (having renumbered the program so that the section you want can be isolated easily). Once the file is on disk (DV80 format) you can either use FWEB/TI-Writer to load in the part you want and then resave under another name, or write your own routine to perform a similar task. There is a public domain program knocking about somewhere (I might even have it!) which will turn a LISTed program into MERGE format, which can then be loaded and resaved as a true program. Alternatively, you could save the entire program in MERGE format and then manipulate the file to divide the wanted sections

into separate files, and then reload those as and when you need them.

There is, as you can see, no easy, straightforward answer!

If you learn anything about MERGE format (DV163) you can always let the rest of us know. Start by reading JIM PETERSON's articles (of which there are more to come in future issues).

Now, to do this for plain TI BASIC files is another matter. Unless you are willing to get into machine code programming (which would help you in performing the same feat with Extended BASIC programs) you are left with deleting all the lines you do want, and saving the rest.

But... There is no MERGE format with TI BASIC, and no possibility of merging useful blocks of TI BASIC statements as with Extended BASIC, unless, of course, you get into machine code programming.

Not much help, but then TI have made a career out of being unhelpful, haven't they?

Continuing with machine code: a couple of puzzled readers mentioned in their letters the problem of understanding how to RUN a machine code program. Their puzzlement was increased by my machine code article in which I used an imaginary BASIC to give examples of relocation. Cops!

To clear the mystery: you cannot RUN a machine code program with the word RUN - as in RUN DSK1.GRAPHICS. First you bring the machine code program onboard using CALL LOAD(). Yes, I know that CALL LOAD() is our version of POKE - it is also capable of loading a machine code program from disk (or RS232/PID, or even CS1 - although I doubt if that is really feasible since DF80 files are not allowed under standard use).

Once the program is onboard, its name is placed in an area of memory which contains what is known as the REF/DEF TABLE. If there are named subsections within the program, they too will have been placed in the same table.

The table indicates where in memory the machine code programs reside, so that they can be called up and run.

The equivalent to RUN here is CALL LINK(), where the link is made to the named program.

Thus if you loaded a graphics program containing subroutines to draw circles, ellipses, squares, lines, etc., then you might use CALL LOAD ("DSK2.GRAPHIC/O"), with subsequent CALL LINK ("CIRCLE", a, b, c) or CALL LINK ("SQUARE", a, b, c, d) and so on.

The CALL LINK() is made from either Extended BASIC or from TI BASIC when a suitable module is present (such as MiniMemory or Editor/Assembler).

It is possible to produce a machine code program which runs itself once loaded, eliminating the need for CALL LINK().

No doubt that explanation has now caused further confusion - don't forget to write in and ask for more confusion - er - explanation.

SPEECH SYNTHESIS

A brief article about Speech Synthesis by ROY MARCHANT

I obtained a copy of the Speech Synthsiser Data manual from ITUG, and it has come in very handy, informing me of some new commands about which I previously knew nothing.

Mucking about with speech

Here is a small program which may spark off a few ideas for the people who possess a speech synthesiser and the MiniMemory module, or a module which provides the CALL LOAD() command.

This program says the words TEXAS INSTRUMENTS until a key is pressed. The routine allows you to stop the phrase halfway through, so that you get the effect of it being "scratched".

- 10 REM A PROGRAM TO SCRATCH A PHRASE
- 20 CALL LOAD (-27648, 70, 0, 73, 0, 70, 0, 70, 0, 64, 0, 80, 0)
- 30 CALL KEY (0, K, S)
- 40 IF S=0 THEN 30
- 50 CALL LOAD (-27648, 112, 0)
- 60 GOTO 20

How the program works

Line 20 loads and says the words TEXAS INSTRUMENTS

Lines 30 and 40 wait for a keypress

Line 50 resets the speech synthesiser, even if it is still processing the command in line 20

Line 60 jumps back to line 20

If you have a listing of the words/phrases and their data values, then all you need to do to scratch your own choice of word/phrase is to follow the steps below:

- 1. TEXAS INSTRUMENTS data is: 06
 - 06 06 09 06 (hex)
- 2. Reverse the data value:
- 06 09 06 06
- J. Add 4X (hex) to the values:
- 46 49 46 46

- 4. Add an "end marker":
- 46 49 46 46 40
- 5. Turn the hex data into decimal: 70 73 70 70 64
- 6. Replace the appropriate values in line 20 with the new data

TRIALS AND TRIBULATIONS

EDDY TURNER (7 Ganton Court, Ridgeway, South Shields, Tyne & Wear, NE34 8AW)

NOVEMBER 1985 - It all started when I was given money as a Christmas present by my wife and my father-in-law.

I made a bee-line for a local computer shop to buy a printer (made by ROTRONICS). It was then that my problems began...

I tried to obtain a saddle lead to connect the computer to my bright new printer. The shop where I bought it said that yes, they could supply such a lead, but a month later I still had no luck. I decided to contact PARCO at Honiton. They told me that I would need an expansion box and that both this and the saddle lead could be supplied by them.

Off went my money plus the details of the printer pinouts.

After about three weeks an ATRONICS box plus saddle lead arrived (by now this would be around February 1986) but when I connected everything up - disappointment! No printout.

After about a week I decided on another phone call to Parco, and was conned into buying three cartridges to try and make the printer function (Finances, Statistics, and Home Finances). I was also told that the instructions for the Atronics (which are supplied in German) were not available in English.

It was beginning to seem like some frustrating Adventure game, but I continued to try every avenue I could think of, in an attempt to get hard copy from my printer.

Around June 1986 I decided to have the Atronics unit and the lead checked out, so I sent it back to Parco. After about two months, they were returned and I was advised that they were both OK. I tried setting everything up again, but still no joy.

Then I came across a firm in Appleby, North Yorkshire, who double-checked all the leads for me, told me that Parco's lead was not right, and supplied a rewired version. Still, however, the printer refused to work.

It was about this point that I began to converse with Gabby - er - Peter who started to give me some hope.

{The cheque is in the post! PB}

This would now be about August 1986. After a couple of months and several long phone calls, it was agreed that I should send printer, Atronics unit, and all leads and documentation to Peter, to see if he could unravel the knot. (This was AFTER I had written direct to Atronics in Germany to see if they could help, but they showed a total lack of interest). Off went an enormous parcel to Peter.

The gremlins struck yet again. The GPO lost the parcel, but after six weeks it suddenly appeared on his doorstep without explanation.

Peter got to work, and after about a month of head banging and some incomplete results (it did print, but it lost characters during the carriage returns), he decided to send the printer back to me, and the Atronics unit back to Germany via Parco, who returned it to me after about another two months.

By now this was February 1987! After fourteen months and three hundred pounds, I still did not have a working printer, and I felt really dejected, fed up, and sick. There was my working TI, raring to go but no printer to work with.

At this point, I grabbed a large notepad and began to go over every tiny detail, bit by bit.

I worked on everything I could get my hands on — Atronics box, leads, printer dipswitches, — I even got books from the local library to try and decipher the German instructions — and after a month and a call or two to Peter I was down to my TI sending via "CENT" and reporting an I/O error.

Finally something snapped. I got mad and started stripping down the lead between the Centronics interface and the Atronics unit. Armed with my little test meter and some hope, and determined to junk the lot if I did not succeed (I had just been told that Rotronics had folded — agh!), I began to teach myself as much as I could about the pin connections.

At the end of a week of furious mental effort I finally - finally!! - found the solution!

All the documentation had said that lines 10 and 11 had to be connected to 11 and 12, but this turned out to be utter hogwash. Whoopee! After almost two years I had achieved the ultimate! Printout!

Peter said that it just went to prove that if you didn't lose hope, something would be bound to happen. {Sounds like the kind of smartass thing I'm likely to say! PB}

I still have to get a translation of the German documentation, though, so the saga is not really over just yet.

The moral of the story is: keep on trying, never give up. In my case, I went through all of this on too of being made redundant (and I'm looking for another job - any offers ?!).

If anyone would like to write to me (it's pretty lonely up here in the North) you are only too welcome.

I would like to thank Peter and any of his crew who helped out, and I strongly advise anyone who has an Atronics unit to check their pinouts and avoid being fed up to the back teeth!

PS The gremlins have struck again: now that I have a working printer, I decided to order TI-Writer III from Parco. They've just advised me that it must have been lost in the post...

NOTICE BOARD

WANTED / 4 SALE / WANTED / 4 SALE / WANTED / 4 SALE / WANTED / 4 SALE /

I am now able to offer a DISK RECOVERY and DISK DRIVE REPAIR service. The disk recovery, where possible, will cost £5 per disk, regardless of how damaged it is.

The disk drive repair will carry a blanket charge of £25 excluding post, packing, and insurance. If you send a drive for repair, SEND NO MONEY. It is possible that the drive cannot be repaired, so you should not part with cash for something which cannot be done. DO PROVIDE ALL POSSIBLE INFORMATION. I cannot repair a fault unless I know in the smallest detail what it is. You will be advised before any major work is undertaken if the cost is going to be excessive, so that you may elect not to have repairs carried out. Please provide a home phone number if at all possible, to speed things up.

In some cases the repairs may take considerable time (due to poor availability of components) so you may have to be patient. I am looking at the possibility of providing DRIVE RENTAL to allow ITUGers to either keep going when their main drive lets them down, or if they want to upgrade but cannot afford the cost for a while. Prices and details on application. 0865 510822 or write to ITUG (address on title page).

WANTED: an expert on PRBASE to answer questions and perhaps write an article or two. Contact 086S 510822 or write to ITUG.

WANTED: budding authors to write personal reviews on some of their own hardware. PAYMENT MAY BE FORTHCOMING IN THE FORM OF PACKS OF 10 DISKS. Further details on application. 0865 510822 or write to ITUG.

FOR SALE: 32K card, £45 including delivery. Contact 0865 510822 in the first instance, or write to ITUG.

FOR SALE: DOUBLE-SIDED 40 TRACK FULL HEIGHT MPI drive, secondhand, suitable for insertion in the PEB or into an external case with power supply. The seller is asking £60 cash including post and packing for a quick sale. Contact ITUG in the first instance.

FOR SALE: brand new disks (double-sided, double density, 96tpi format, but can be used as single-sided, single density, 40 track with no difficulty) at only £7 per pack of ten. No library cases, but the price is inclusive of post and packing. 0865 510822 or write to ITUG.

FOR SALE: TEII with manual. £20 including post and packing. Contact 0865 510822 or write to ITUG. Complete with TEII Protocol Manual.

JOHN BUTCHER would like to obtain copies of HCM V5.3, and V5.4 if anyone is willing to part with them. He would also like any technical data on the SHUGART SA455, together with information on how to set the links. Contact him on 0293 884204.

ANTONY FIELD has decided to "end it all" and is selling up. He has a PAL 4A console which requires attention (the modulator sounds as though it is dodgy). Offers? He also has two home-made cassette leads at £3, a C15 Memorex tape with three games (Nuclear Disaster, Golf, and Sea Wolf) at 50p including postage, and a few books: PROGRAMMING BASIC WITH THE TI HOME COMPUTER by H. D. Peckham, Byte Publications, at £2; THE TEXAS PROGRAM BOOK by V. Apps, published by Phoenix (35 programs) at £3; MASTERING THE TI-99 by P. Brooks (who?), published by MicroPress, also at £3; TANTALIZING GAMES FOR YOUR TI-99/4A by H. Renko and S. Edwards, published by Addison Wesley, (31 games) at £2; and INITIATION INTO THE TEXAS INSTRUMENTS TI-99/4A ASSEMBLY LANGUAGE (requires Minimemory) which has been translated from the French (original author not given and I cannot remember who it was!), published by Shift Editions, at £4.

Contact Antony at 31 OAKFIELD AVENUE, WRENBURY, NANTWICH, CHESHIRE, CW5 8ER.

JO ANN COPELAND has suggested that those looking for thermal paper for their TI thermal printers might find a solution from TENEX in the States who advertise two 100 feet rolls at \$11.95. It does state that the paper is for the Star Micronics STX-80 but Jo feels that it may be compatible. I know nothing about the Star Micronics printers, so I cannot advise whether the paper is compatible. Anyone fancy ordering some and letting the rest of us know? I don't have TENEX' details to hand - you might like to contact Jo on 063 881 3457 and talk to her.

I couldn't fit this information conveniently in the LETTERS section: MAURICE RYMILL and others have pointed out an error in the listing of HANGMAN in V4.4. In line 3660 the loop which defines colours for sets 9 to 16 will not of course work in Extended BASIC, as there are only 14 colour sets! The loop should run from 9 to 14. In addition, the CALL COLOR() in line 3670 refers to set 1 all the time, instead of the set pointed to by the variable I. Change it to (I, 5, 15) to make things more colourful.

PETER KILLICK and others (including TREVOR TABERNER who supplied the listing) have pointed out that Extended BASIC permits several variables which are being set to the same value to have the assignment performed in a single statement, thus: A, B, C, D, E(1), F=999. In any lines where several variables are being assigned the same value, therefore, you can shorten (and speed) the program by compressing appropriately.

Keep these error reports coming in...

DOUG WARD tells me that DEREK FERN on 021 353 5730 is selling FERGUSON GREEN SCREEN MONITORS with Atari Leads for £32. Doug has had one for a little while now and he is happy with it. Apparently Derek has 8 up for sale, so grab them while you can!

FRANK CASEY is progressing with a complex telephone accounting program, and is currently looking for a set of the CANADIAN NXX CODES. I gather from Frank that these could amount to some 80,000 codes - if anyone can offer any assistance, would they please contact me? 0865 510822 or write to ITUG.

Under the current provisions of the Data Protection Act, members of the public may demand details of all data held on them by ITUG. Details of how to apply, and the cost, will be given next issue.

THE NEW HORIZON RAMDISK!

INTRODUCTION

The HORIZON RAMDISK is a peripheral card for the TI-99/4A Home Computer. The RAMDISK is a battery-backed, solid state device which is intended to emulate all functions of a TI floppy disk drive. The primary advantage of the RAMDISK is speed; data transfer takes place roughly twenty times faster.

To use the HORIZON RAMDISK you need:

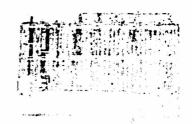
- 1) TI-99/4A console and TV or monitor
- 2) 99/4 Peripheral Expansion Box
- 3) TI Compatible Disk Controller Card
- 4) At least one floppy disk drive
- 5) One of the following Command Modules
 - A) Editor/Assembler
 - B) TI Extended BASIC
 - C) Mini Memory Command Module
- 6) 32K Memory Expansion (recommended)

COMPATIBILITY

Every attempt has been made to ensure that the HORIZON RAMDISK will be compatible with TI and third party software. The RAMDISK appears to be compatible with TI Writer, Editor/Assembler, Multiplan, Logo, Forth, BASIC, Extended BASIC and Assembly Language. Most applications concerning the above should be compatible with the RAMDISK. However, some software, designed to directly access the Disk Controller Card will not recognize the RAMDISK and hence is not compatible. Examples include the CorComp Disk Manager Program and TI Pascal. Non-standard disk access techniques could also cause compatibility problems (e.g., TI-ARTIST).







-- HORIZON RAMDISK FACTS --

- A RAMDISK card for the 99/4a PE-Box.
- 104K (SSSD, 360 sectors) capacity, expandable to 192K (DSSD, 720 sectors).
- Uses low-power CMOS static RAM chips.
- Functions EXACTLY like a TI floppy drive but at Random Access speed.
- Compatible with any software or language that uses a standard TI DSRLNK, including disk utilities, sector copy programs, and programs like DM-1000, DISK MANAGER II, and Millers Graphics EXPLORER program. Compatible languages include TI-BASIC, EXTENDED BASIC, TI-FORTH, TI-LOGO and ASSEMBLY LANGUAGE.
- The only BATTERY-BACKED Ramdisk for the 99/4a, carry it from computer to computer like a floppy diskette. Ni-cad batteries charge while the computer is running.
- Supplied with DM-1000 as the resident disk manager program. DM-1000 may be loaded in less than two seconds via a CALL DM statement from BASIC.
- other CALL statements from TI-BASIC to 1) set the Ramdisk drive #, 2) set the maximum number of sectors, 3) set the write protection, 4) turn on CRU for direct DSR access, 5) execute machine code from BASIC.
- DIP switch setting allows CRU addressing from >1000 to >1700.
- Ramdisk may be named DSK1 to DSK6.
- Comes with complete DSR source code, including a separate manual that details all DSR routines.
- Documentation explains how users may add their own assembly language CALL routines to enhance BASIC.
- Comes with development software, including a loader for any E/A opt. 3 object files to allow users to modify the DSR as desired.
- Schematic Diagram and construction guide included.
 - * 2022 note: The Mini Memory module functioned as battery backed ram well before the New Horizon Card. The button cell could be replaced with a NiCad battery.