TANGERINE USERS GROUP

TUG

NEWSLETTER

ISSUE 11

MORE ON THE EPROM PROGRAMMER

We have recieved many enquiries regarding the Eprom Programmer; apparently the initial news release of its arrival has stimulated enormous interest. The object of the exercise was to bring within reach of the users a working tool at a price which anybody could afford. Consider therefore, that you now have an opportunity to buy an eprom programmer at the same cost as a games tape!!

The idea of using an eprom programmer has, for many, been clouded by the fact that the user may programme errors into the chip, and therefore find at the end of the day that the whole idea of putting one's programs into eprom is an expensive one! This is just not so any more. Consider this. The eprom programmer project has been designed to make life very easy for the experienced or the novice. The construction of the programmer is so simple it is almost unbelievable. With only six components to solder to the board, it's ridiculous!! The notes that accompany the p.c.b. include: component overlays to show you the actual position of components on the p.c.b.; pin connection diagram for connecting to Tanex (**stop press news:** We are now able to supply the cables with plugs attached – very neat!). There is a comprehensive guide on the use of the software; circuit diagram for those with a little bit more experience and the automatic programming feature in the software allows even the beginner to achieve 'clean burns'. All round then 'a nice one Eric'.

If any of you are a little uncertain of construction, we will do it for you. Just give us a TUGPROMring at any time.....the Eprom Programming Package is in stock **NOW!** By return of post!!!

TUG LOOKS AFTER ITS MEMBERS

The Microtan System is truly an expandable system for its users. TUG has been so disappointed at the lack of interest shown in this direction, that it has now started to produce all sorts of 'goodies' for the system. Starting with the Programmer we intend to increase our production of devices to cover a wide range of user auxilliary hardware. This hardware is going to be brought to you in a simple format: no fuss, no ribbons, just plain, usable, sensible hardware at prices **you** can afford. If you want to pay for 'wrapping', go ahead. You're certainly going to get a few surprises with what we have in store for you. We are so confident that members will be buying our hardware, that we are going into production before we tell you what's available.......

Ed's Say....

It seems we started something with the 'T-SHIRT' idea, which, quite honestly we didn't expect. Now suggestions are coming in for Lapel Badges and 'Toilet Paper' (profiles of WHO?!). Anyway, we shall be looking into these possibilities; in the meantime, come on, let's have some suggestions apart from the TUG logo......**ED**

With the large numbers of Renewal registrations coming in this month, administration is being stretched to the maximum. If you encounter any delays in response to your letters etc., we apologise. Don't stop sending them in, we enjoy reading them and help where possible. Our grateful thanks for your articles, which we shall use as soon as possible......**ED**

As a matter of interest, I am constantly being asked 'Who is Eric?'. Quite simply, Eric is the *@*@*@*@* chunk of silicon that keeps me working most nights. He's got no sympathy for me, or me for him. As a prime example of Tangerine's engineering there's no better. More often than not, he gets left on and running for weeks at a time without the slightest sign of malfunction, that is, unless he's using Basic!! Since the introduction of the MPS 2, dropping out due to spikes on the mains is non-existent. So now I work him even harder.
Message from Eric "Didn't you know you could use the Interpreter with SHIFT-A on the keypad?......'I' see!

When using the cassette routines for handling files developed in the '65 1K of memory, it is well to COPY the entire program in the first page on Tanex and then dump to tape. On loading the program, simple use of the COPY command can restore the program to its original location, e.g. C0050,01FF,0450 When recovering the program from tape it would then become C0450,05FF,0050. This simple routine removes the tedious task of relocating programs after upgrading to Tanex and XBUG......**ED**

LETTERS

CALLING ALL TANBUG USERS!!

CLEARING THE SCREEN: are you fed up with machine code USR routines? Does the sight of.
'FOR X=1 TO 16:?:NEXT X' make you ill? Does your feeling of superiority over users of other nameless computers go when they retort "At least I have a decent Clear Screen comand!" If the answer to these questions is YES! YES! YES! then this is for YOU!
Insert this line at the beginning of your programs: 1 FOR X = 1 TO 16 : CL$ = CL$ + CHR$(13) : NEXT
Then, whenever you wish to clear your screen:– 1000 ? CL$; "Whatever you want to print etc."
Good Eh? Any variable could be used, but I chose CL$ because it's near enough to CLS, so it's easy to understand when LISTing.

A.L.Shepherd Orchard Cottage, Saville Hall Lane, Dodworth, Barnsley, Yorks. S75 3NG.

Dear Bob
Can you suggest a pseudo number generator which I can use for games programming, and one which I can call at random intervals. What I'm after is one which will occupy a small number of bytes.

The simplest method would be to use the 6522 VIA, by loading one of the low byte counters and storing the result at a location to be picked up by the main calling program i.e.

```
0050   ADC4BF      LDA $BFC4     ;GET RANDOM VALUE FROM COUNTER
0053   8540        STA $0040     ;STORE FOR USE BY MAIN PROG
```

Providing the value is not called by a regular cycle it makes a very good random value generator routine. You can in fact duplicate this routine again using another counter, adding the two together and then storing the result.

Dear Bob,
I have thought of a short program to include in the newsletter. I have speeded it up as much as possible:

```
 5  FOR A = 0 TO 15 : PRINT : NEXT
10  A$ = "This string can have up to 71 characters.
          The quotes don't need closing.
15  C = 768 : D = 799 : E = 767 : F = LEN(A$)-1 : I = 1
20 FOR A = C TO D : POKE A,ASC(MID$(A$,A-E,I)) : NEXT
25 A$ = RIGHT$(A$,F)+LEFT$(A$,I) : GOTO 20
```

Rather than explain what it does, type it in and run it. Hope you can find space for it.
Yours sincerely, Mark Richardson.

(Don't we get them! – **Ed**)

Dear Bob,
Thank you for an excellent Newsletter over the last 12 months. Please find enclosed my application form for the renewal of my membership for the next year.
I hope I can now contribute a small amount to help TUG. I have recently obtained a copy of Tanbug V2.3 and have made the following discoveries: Byte Zero contains various bits that control the 'new facilities'.
Numbering from LSB=1 to MSB=8, the bits I have found out about are:–
Bit 7=1 Inhibits the screen (the same as CTRL-S on the keyboard). Bit 5=1 Means that all print statements should be output to a parallel printer. Bit 6=1 I believe will enable output to a serial printer on the UART, but I don't have a printer to try it out.
To go along with these there are several subroutines which can be accessed from Basic. Call F941 to initialise the VIA for printing. A useful sequence is:–

```
10 POKE 34,65: POKE 35,249: X=USR(X):POKE 0, PEEK(0) AND 239: REM INIT VIA
20 REM USE 'GOSUB 30' WITH STRING TO BE OUTPUT IN 'Z$'
30 POKE 0, PEEK(0) OR 16: PRINT Z$: POKE 0, PEEK(0) AND 239: RETURN
```

If you don't want to see it on the screen, then make the 'OR 16' into 'OR 80', and the 'AND 239' into 'AND 179'. You can then output to the printer as and when you like without having to worry about any other 'prints' and 'inputs' used to 'drive' the program.
Other useful subroutines:–
F96E Call once = screen off; call 2nd time = screen on.
F934 Same action as above for parallel printer.
F981 Same as above for serial printer.
FA23 Clear screen routine.
FA3B Scroll screen routine.
E6D9 In the basic ROM is a useful delay routine, provided you fill out Registers X and Y with the required values before the JSR.
Yours sincerely, Ray Griffith, 20, Claremont Avenue, Hersham, Waton-on-Thames, Surrey. KT12 4NS.

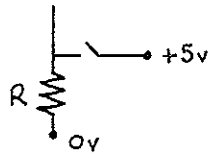
Ray Griffith, 20 Claremont Avenue, Hersham, Walton-on-Thames, Surrey. KT12 4NS. (Edited).

PROJECTS . . . . HARDWARE . . . . SOFTWARE

CONNECTING 'ATARI' JOYSTICK CONTROLS TO TANGERINE

By J Haines

An Atari joystick contains 5 on/off switches, one for each of the four principal directions, and one for the 'fire' button. The colours of the wires coming from the unit are:
Up – white, down – blue, left – green, right – brown, fire – orange; the common line to all five is black. (Note: this has been true for 3 joysticks in my experience but it would be as well to check).
There are two basic ways to use an on/off switch to control the micro.The easiest is to connect it to the keyboard or keypad and use it as an extension to that. The second, and more satisfactory, is to connect it to an I/O port. This requires the switch to set either of two states: high (plus 5v) or low (earth). This is done using the simple circuit shown below:



R should be about 1K

Using the Atari joysticks, the neatest place to put the resistors is inside the joystick case. As you can see, this does have one drawback; each switch needs **3** lines, 0v, plus 5v and out. So it is necessary to run another wire to the unit if all five functions are to be used. My latest effort, comissioned by our noble editor, was to make up a joystick to be used with the TUG Library program, SPACE FIGHTER. This uses six on/off switches and the additional one can easily be fitted into the joystick case, and the two extra wires required run into the case. The moulded on plug has to be cut off and a 14 pin DIL plug fitted to connect it to the Tanex socket. In the absence of any standard it is necessary to have either a dedicated joystick for each game, or make up a range of interface boards with a socket and plug connected in the right way, which connects inbetween the joystick and Tanex. The big advantage of using an I/O port is that movement can be diagonal as well as in the four basic directions. It is well worth taking a deep breath and finding out how to use the basic I/O capabilities. Atari joysticks cost £12 – £13 per pair and are available from some shops that stock Atari Machines.

* *

FROM..... BYTE MASTER.....

EXTENDING THE BASIC COMMAND SET
==============================

It is relatively easy to add DIRECT commands to BASIC - all you have to do is intercept data from the keyboard before it goes back to BASIC by altering the Slow Interuppt Link.The data can be checked for particular user-defined commands which could then be executed before returning to BASIC.

This method will,of course,NOT work for commands IN programs,to be executed at runtime.However,I`ve discovered a way of adding such commands,though there are some limitations,the worst of which is that the display has to be scolled in the process of executing the User-defined command(s).Hence all graphic information is lost.This is not important if ,for example,the command is Clear Screen,a feature the Microtan Basic does not have.Clear Screen (CLS) is the most useful addition I can think of,a program for which is printed here.In practice,it is probably not worth incorporating this procedure for just the one command-most of this program is concerned with leaving and reentering BASIC correctly,and is required only once,for any number of commands.So if you can think of perhaps 6 or 7 useful additions,they could all be included in one program in about 300 BYTES.

If you think of any good additions,write in to BYTE MASTER.
    The whole method described here rests upon the fact that whenever BASIC prints OK and returns to command mode,it jumps to a location in Zeropage first, which contains the instruction JMP $CA1Ø . This returns the processor to a BASIC subroutine which prints OK.If you diss-assemble the BASIC from C34D,you will see this.The values in A and IY are necessary to indicate to the subroutine at CA1Ø that OK is to be printed.(A=67,IY=C2 because OK is stored from C267-look for yourself).
    If you put the command CLS in a program,BASIC will stop running,print SYNTAX ERROR IN LINE XXXX,scroll the screen,then print OK.BUT it will have jumped to location 1A first,and if you alter this to go to your own routine you can add the command CLS.
    You must ensure that the jump was for the right reason-not for some other error,STOP or END statement or any other reason for printing OK.If the jump is for the wrong reason,return to BASIC is easily accomplished by---

```
LDA #$67
LDY #$C2
JMP $CA1Ø
```

--- as would have occurred if 1A,1B,1C had not been altered in the first place.
    If the jump was for a user-defined command,after executing it,the processor must return to BASIC and continue running the program.There is probably a location to which you can jump for this purpose,though I have not found it.If any of you find it please let me know.Instaed,the method I use,which takes the largest part of the program,is to automatically give the computer the direct command GOTO YYYY which is the line after that which contained CLS.Therefore CLS (or whatever) has to be the last or only statement in a line.

PROGRAM
=======

N.B. E9 and EA are used by Basic to point at the location which it is interpreting at any given time either from $35 if a Direct command or from $Ø4ØØ if in a program -either as the first line in a program, or as a direct command-
N.B. THIS MUST BE A ONE-LINE COMMAND

```
Memory size-8Ø11       To initialize-POKE27,75:POKE28,31

1F4B A5Ø1    LDA $ØØØ1        %      Jumps here from 1A
1F4D C9Ø3    CMP #$ØØØ3       I      Here because "Break" has been typed?
1F4F FØ57    BEQ $1FA8        pW     Yes,return to Basic
1F51 A5EA    LDA $ØØEA        %j     Look at pointer
1F53 FØ53    BEQ $1FA8        pS     =ØØ indicates jump to 1A was after a direct command-return
1F55 A5E9    LDA $ØØE9        %i
1F57 DØØ8    BNE $1F61        P
1F59 A5EA    LDA $ØØEA        %j
1F5B C9Ø4    CMP #$ØØØ4       I
1F5D FØ49    BEQ $1FA8        pI     Pointer at Ø4ØØ-beginning of program-return
1F5F C6EA    DEC $ØØEA        Fj
1F61 C6E9    DEC $ØØE9        Fi
1F63 AØØØ    LDY #$ØØØØ
1F65 B1E9    LDA ($ØØE9),Y    1i     Look at previous location to one being pointed at
1F67 E6E9    INC $ØØE9        fi
1F69 DØØ2    BNE $1F6D        P
1F6B E6EA    INC $ØØEA        fj
1F6D C98F    CMP #$ØØ8F       I      Code for STOP-reason for jump to 1A was for a STOP instruction
1F6F FØ37    BEQ $1FA8        p7
1F71 C98Ø    CMP #$ØØ8Ø       I      Code for END-reason for jump was for END instruction
1F73 FØ33    BEQ $1FA8        p3
1F75 C6EA    DEC $ØØEA        Fj
1F77 AØØØ    LDY #$ØØØØ
1F79 88      DEY                     If we get here,jump to 1A was for runtime error
1F7A B1E9    LDA ($ØØE9),Y    1i     and pointer will look at location after last letter
1F7C C93A    CMP #$ØØ3A       I:     of command which caused the error.
1F7E FØØC    BEQ $1F8C        p      Look back until beginning of this command is found-
1F8Ø C5A9    CMP $ØØA9        E)     3A will be colon seperating command from previous one
1F82 DØF5    BNE $1F79        Pu     if  it is a multiple statement line.If command is the
1F84 88      DEY                     only statement in the line,previous locations will be
1F85 B1E9    LDA ($ØØE9),Y    1i     the binary version of the linenumber-also stored in A8,A9
1F87 C8      INY              H      so compare with A8 and A9.
1F88 C5A8    CMP $ØØA8        E(
1F8A DØED    BNE $1F79        Pm
1F8C C8      INY              H      Found beginning of command
1F8D B1E9    LDA ($ØØE9),Y    1i
1F8F C943    CMP #$ØØ43       IC     First letter =C?
1F91 DØ13    BNE $1FA6        P      No,wrong reason for jump-return to Basic
```

```
1F93 C8     INY              H
1F94 B1E9   LDA ($00E9),Y    1i      Second letter =L?
1F96 C94C   CMP #$004C       IL      No,return
1F98 D00C   BNE $1FA6        P
1F9A C8     INY              H
1F9B B1E9   LDA ($00E9),Y    1i      Third letter =S?
1F9D C953   CMP #$0053       IS
1F9F D005   BNE $1FA6        P       No,return
1FA1 C8     INY              H
1FA2 E6EA   INC $00EA        fj
1FA4 B1E9   LDA ($00E9),Y    1i
1FA6 F003   BEQ $1FAB        p       Next location zero ?If not,command is longer than CLS-return to Basic
1FA8 4CF91F JMP $1FF9        Ly
1FAB A920   LDA #$0020       )       At last!Reason for jump to 1A WAS for CLS
1FAD A200   LDX #$0000       "       Do it.
1FAF 9D0002 STA $0200,X
1FB2 9D0003 STA $0300,X
1FB5 E8     INX              h
1FB6 D0F7   BNE $1FAF        Pw
1FB8 8603   STX $0003                Clear display index
1FBA A001   LDY #$0001
1FBC B1E9   LDA ($00E9),Y    1i      If this lcation is zero.....
1FBE D005   BNE $1FC5        P
1FC0 C8     INY              H
1FC1 B1E9   LDA ($00E9),Y    1i      and this one is zero,
1FC3 F034   BEQ $1FF9        p4      this CLS command was the last command in the program-return
1FC5 A204   LDX #$0004       "       Otherwise reenter program by leaving in Basic input buffer---
1FC7 BDA2C0 LDA $C0A2,X      ="@     GOTO  as stored at C0A2 onwards
1FCA 297F   AND #$007F       )
1FCC 9534   STA $0034,X      4       I/P buffer from $35 onwards
1FCE CA     DEX              J
1FCF D0F6   BNE $1FC7        Pv
1FD1 A003   LDY #$0003               Peep at linenumber of next line as stored 3 locations further on
1FD3 B1E9   LDA ($00E9),Y    1i      after location pointed to by E9 and EA
1FD5 AA     TAX              *
1FD6 C8     INY              H
1FD7 B1E9   LDA ($00E9),Y    1i      Put binary version of next linenumber in A and IX
1FD9 206BDC JSR $DC6B        F\      and jump to Binary to decimal subroutine in Basic
1FDC A001   LDY #$0001               which prints it on the screen
1FDE B10A   LDA ($000A),Y    1       Fetch it from screen
1FE0 C920   CMP #$0020       I
1FE2 F00A   BEQ $1FEE        p       Finished
1FE4 993800 STA $0038,Y      8       and place in input buffer after GOTO already stored
1FE7 A920   LDA #$0020       )
1FE9 910A   STA ($000A),Y            Rub off linenumber from screen
1FEB C8     INY              H
1FEC D0F0   BNE $1FDE        Pp      Loop until done
1FEE A900   LDA #$0000       )       Indicate end of command by placing a zero in next location
1FF0 993800 STA $0038,Y      8       in input buffer.
1FF3 A8     TAY              (       IY=0
1FF4 A234   LDX #$0034       "4      IX=34    - to tell Basic to look at Input buffer
1FF6 4C57C3 JMP $C357        LWC     Jump back to Basic to subroutine which will interpret the command
1FF9 A967   LDA #$0067       )g      Return to Basic if jump to 1A was for wrong reason-
1FFB A0C2   LDY #$00C2        B      by setting up A and IY and jumping to CA10 as would have happened
1FFD 4C10CA JMP $CA10        LJ      if 1A,1B,1C had not been altered originally.
```

THE WALL

D.JAMES

Although this programme has been written for the Microtan 1K there should be no difficulty in relocating it. Written for use with the keypad the game commences after s short pause. For two players sharing the keypad, their key controls are (Left Hand Player) 'C' UP, '8' DOWN, '4' LEFT, '0' RIGHT & (Right Hand Player) 'F' UP, 'B' DOWN, '7' LEFT, '3' RIGHT.. Scores are kept in the respective top corners of the screen, and the game starts with both players situated in the lower half of the screen. The idea is to avoid crashing into the barrier or wall whilst trying to make your opponent crash. BEWARE.. You are liable to crash into yourself...

If you find the games speeds up too quickly, change the value at location 00AA from 08 to a lower value. To move the starting positions, change the values at 0015 (low order A) and 0019 (low order B). If it ends up going to fast, then increase the value at 00A4 from 20.   START G50 (CR)

```
0050 20 BC 00  START      JSR INIT          ;Initialise PTRS
0053 20 CB 00  NXTGAM     JSR CLSETP        ;Reset PTRS
0056 A5 40     MNLOOP     LDA Point A       ;Head on collision
0058 C5 42                CMP Point B
005A D0 06                BNE NO Draw       ; No skip
005C A5 41                LDA Point A+1     ; Check high byte
005E C5 43                CMP Point B+ 1
0060 F0 32                BEQ Draw          ; Yes, jump
0062 A2 00     NODRAW     LDX *$00
0064 A1 40                LDA (PointA,X)    ;A CRASH ?
0066 D0 19                BNE ACRASH        ;YES jump
0068 A1 42                LDA (PointB,X)    ;B CRASH ?
006A D0 20                BNE BCRASH        ;YES jump
006C A9 FF                LDA *$FF          ;Put next Point
006E 81 40                STA (PointA,X)    ; Onto screen for
0070 81 42                STA (PointB,X)    ; A and B
0072 20 47 01             JSR RDMDFY        ;READ KP,Modify addr
0075 E6 4B                INC TIME          ;How long lasted!
0077 A4 4A                LDY SPEED         ; How long next move
0079 CA        DLX 1      DEX               ;Delay
007A D0 FD                BNE DLY 1
007C 88                   DEY
007D D0 FA                BNE DLY 1
007F F0 D5                BEQ MNLOOP        ;UNCOND jump
0081 A2 02     ACRASH     LDX *$02          ; SET PAR
0083 20 A1 00             JSR UPSCORE       ; Modify Score B
0086 A2 00                LDX *$00          ; Chech if B crashed
0088 A1 42                LDA (PointB,X     ; as well
008A F0 C7                BEQ NxtGam        ; If not start again!
008C A2 00     BCRASH     LDX *$00          ;SET PAR
008E 20 A1 00             JSR UPscore       ; Modify score A
0091 4C 53 00             JMP NxtGam        ;Start again
0094 A2 02     DRAW       LDX *$02          ;SET PAR to MOD B
0096 20 A1 00             JSR UPSCORE       ; Modify B
0099 A2 00                LDX *$00          ;PAR for A
009B 20 A1 00             JSR UPSCORE       ; MODIFY
009E 4C 53 00             JMP NxtGam        ;Start again
00A1 A5 4B     UPSCORE    LDA TIME          ; Faster next
00A3 C9 20                CMP *$20          ; time
00A5 30 06                BMI SAMSPD        ;NO branch
00A7 A5 4A                LDA SPEED         ;YES Modify speed
00A9 E9 08                SBC *$08          ;
00AB 85 4A                STA SPEED         ;
00AD F6 46     SAMSPD     INC SCORA,X       ; Update Score
00AF B5 46                LDA SCORA,X       ; Is it greater
00B1 C9 3A                CMP *$3A          ; than '9' ?
00B3 D0 06                BNE OK            ; NO branch
00B5 A9 30                LDA *$30          ;YES set back
00B7 95 46                STA SCORA,X       ; to '0'
00B9 F6 47                INC SCORA+1       ;ADD 1 to 10's
00BB 60        OK!        RTS               ; RETURN
00BC A9 00     INIT       LDA *$00          ; Set speed to slowest
00BE 85 4A                STA SPEED
00C0 A9 30                LDA *$30          ; SET all scores
00C2 85 46                STA SCORA         ; To '0'
00C4 85 47                STA SCORA+1       :ASCII=30
00C6 85 48                STA SCORB
00C8 85 49                STA SCORB+1
00CA 60                   RTS               ;RETURN
00CB A2 05     CLSETP     LDX *$05          ; SET delay time
00CD 86 4C                STX COUNT
00CF CA        DLY2       DEX               ; DELAY
00D0 D0 FD                BNE DLY2          ; About 10 secs
00D2 88                   DEY
00D3 D0 FA                BNE DLY2
```

```
0OD5 C6 4C                 DEC Count
00D7 D0 F6                 BNE DLY2
00D9 AD F0 BF              LDA $BFF0           ; Set Graphics
00DC A9 00                 LDA *$00            ; Blank Block
00DE AA                    TAX
00DF 9D 00 02  CLEAR       STA $0200,x         ; Clear Screen
00E2 9D 00 03              STA $0300,X
00E5 CA                    DEX
00E6 D0 F7                 BNE CIEAR
00E8 A9 FF                 LDA *$FF            ; FULL BLOCK
00EA A8                    TAY
0C9E A_ 1F.                LDX *$1F
0OED 9D 00 02  BOUND 1     STA $0200,X         ; Put in top and bottom
00F0 9D E0 03              STA $03E0,X         ;BOUNDRIES
00F3 CA                    DEX
00F4 10 F7                 BPL BOUND1
00F6 A2 E0                 LDX *$E0
00F8 98        BOUND2      TYA                 ; Put in side
00F9 9D 00 02              STA $0200,X         :BOUNDRIES
00FC 9D 1F 02              STA $021F,X
00FF 9D 00 03              STA $0300,X
0102 9D 1F 03              STA $031F,X
0105 8A                    TXA
0106 38                    SEC
0107 E9 20                 SBC *$20
0109 AA                    TAX
010A C9 E0                 CMP *$E0
010C D0 EA                 BNE BOUND2
010E A9 03                 LDA *$03            ;SET POINTS A
0110 85 41                 STA POINTA+1        ; & B
0112 85 43                 STA POINTB+1
0114 A9 65                 LDA *$65
0116 85 40                 STA POINTA
0118 A9 7A                 LDA *$7A
011A 85 42                 STA POINTB
011C A9 08                 LDA *$08            ;SET BOTH TO GO UP
011E 85 44                 STA DIRA
0120 85 45                 STA DIRB
0122 A9 00                 LDA *$00            ; Not lasted at all
0124 85 4B                 STA TIME            ; YET
0126 8D F3 BF              STA $BFF3           ; Text Mode
0129 A5 46                 LDA SCORA           ; PUT scores
012B 8D 01 02              STA $0201           ; onto screen
012E A5 47                 LDA SCORA+1
0130 8D 00 02              STA $0200
0133 A5 48                 LDA SCORB
0135 8D 1F 02              STA $021F
0138 A5 49                 LDA SCORB+1
013A 8D 1E 02              STA $021E
013D AD F0 BF              LDA $BFF0           ; Graphics
0140 CA        DLY3        DEX                 ;Delay
0141 D0 FD                 BNE DLY3
0143 88                    DEY
0144 D0 FA                 BNE DLY3
0146 60                    60                  ;RETURN
0147 A9 01     RDMDFY      LDA *$01            ; For 1st       on PAD
0149 8D F2 BF              STA $BFF2           ; OUTPUT to PAD
014C AD F3 BF              LDA $BFF3           ;PAD
014F F0 02                 BEQ NCHA            ; A PUSHED ? NO Branch
0151 85 44                 STA DIRA            ; YES change A's direction
0153 A9 08     NCHA        LDA *$08            ; for forth column
0155 8D F2 BF              STA $BFF2           ; output to PAD
0158 AD F3 BF              LDA $BFF3           ; Read PAD
015B F0 02                 BEQ NCHB            ; B PUSHED ? NO branch
015D 85 45                 STA DIRB            ; YES change B's direction
015F A5 44     NCHB        LDA DIRA            ; Direction of A
0161 A2 00                 LDX *$00            ; Parameter for A
0163 20 6E 01              JSR MODIFY          ; MOVE A
0166 A5 45                 LDA DIRB            ; Direction of B
0168 A2 02                 LDX *$02            ; Parameter for B
016A 20 6E 01              JSR MODIFY          ; MOVE B
016D 60                    RTS                 ;RETURN
016E C9 08     MODIFY      CMP *$08            ; Move UP ?
0170 F0 22                 BEQ UP              ; YES branch
```

```
0172 C9 04               CMP *$04          ; DOWN ?
0174 F0 10               BEQ DOWN          ;YES branch
0176 C9 02               CMP *$02          ; LEFT ?
0178 F0 06               BEQ LEFT          ; YES branch
017A B4 40               LDY POINTA,X      ; NO RIGHT get old position
017C C8                  INY               ; MOVE back
017D 94 40               STY POINTA,X      ;REPLACE
017F 60                  RTS               ; RETURN
0180 B4 40     LEFT      LDY POINTA,X      ; Get old position
0182 88                  DEY               ;MOVE Forward
0183 94 40               STY POINTA,X      ; REPLACE
0185 60                  RTS               ;RETURN
0186 B5 40     DOWN      LDA POINTA,X      ; Get old position
0188 18                  CLC
0189 69 20               ADC *$20          ;MOVE Down
018B 95 40               STA POINTA,X      ;REPLACE
018D B5 41               LDA POINTA+1,X    ; ADD 1 to upper
018F 69 00               ADC *$00          ; Byte if changed
0191 95 41               STA POINTA+1,X    ; to bottom half
0193 60                  RTS               ;RETURN
0194 B5 40     UP        LDA POINTA,X      ; Get old position
0196 38                  SEC
0197 E9 20               SBC *$20          ;MOVE up
0199 95 40               STA POINTA,X      ;REPLACE
019B B5 41               LDA POINTA+1,X
019D E9 00               SBC *$00
019F 95 41               STA POINTA+1,X
01A1 60                  RTS               ;RETURN
01A2 00 00               BRK END           ; Good luck!!
```

BINARY TO ASCII CONVERSION. by Ray Griffith

21-06-81

```
* BEFORE CALLING THIS SUBROUTINE THE USER MUST FILL OUT
* BYTES $40 TO $42 WITH THE REQUIRED INFORMATION
*
* ALL REGISTERS ARE DESTROYED THE USER MUST SAVE THEM IF REQUIRED
*
* NB       EQU $40 NUMBER OF BINARY BYTES -1.
* ADDL     EQU $41 LSB OF BINARY NUMBER FOR CONVERSION
* ADDH     EQU $42 MSB OF BINARY NUMBER FOR CONVERSION
* COUNT    EQU $43 LOCAL COUNT FOR NUMBER OF SHIFTS
* BIN      EQU $44 LOCAL BINARY SAVE AREA
* BCD      EQU $49 LOCAL BCD SAVE AREA
* ASC      EQU $42 START ADDRESS OF ASCII RESULT
* BINLEN   EQU $05 MAXIMUM LENGTH OF BINARY INPUT
* BCDLEN   EQU $07 BCD WORKSPACE SIZE
*
* THIS ROUTINE USES THE ADDRESSES IN PAGE ZERO FROM $40 TO $4F
*
* INPUT IS 1 TO 5 BINARY BYTES STORED MSB LEFT JUSTIFIED
*
* OUTPUT IS ASCII CHARACTERS MSD LEFT JUSTIFIED FROM $42 ONWARDS
* TERMINATED WITH AN 'FF'
* LEADING ZERO'S ARE SUPPRESSED BUT AT LEAST ONE ZERO WILL BE
* OUTPUT IF THE BINARY INPUT IS ZERO
**
 A000 A928   LDA £$0028      )(   BINLEN*8    NUMBER OF SHIFTS
 A002 8543   STA $0043        C   COUNT       SAVE IN COUNT
 A004 A20C   LDX £$000C      "    BINLEN+BCDLEN
 A006 A900   LDA £$0000      )    ACCU = 0
 A008 9543   STA $0043,X      C   BIN-1       CLEAR 'BIN' AND 'BCD'
 A00A CA     DEX             J
 A00B D0F8   BNE $A008       P(   CONTINUE TILL END
 A00D A440   LDY $0040       $@   NB          NUMBER OF BINARY BYTES INPUT
 A00F B141   LDA ($0041),Y   1A   GET BYTE
 A011 994400 STA $0044,Y      D   BIN         SAVE IN WORKSPACE
 A014 88     DEY
 A015 10F8   BPL $A00F        x   CONTINUE TILL END
 A017 C643   DEC $0043       FC   COUNT       IS COUNT STILL POSITIVE
 A019 102C   BPL $A047        ,   MORE SHIFTS REQUIRED
**
* MOVE THE BCD OUTPUT ONTO THE STACK LSD FIRST. THEN PULL FROM THE
* STACK DELETING LEADING ZERO'S AND CONVERTING TO ASCII CHARACTERS
**
 A01B A200   LDX £$0000       "   X = ZERO
 A01D B549   LDA $0049,X     5I   BCD         GET 2 OUTPUT DIGITS
 A01F A8     TAY              (   SAVE IN Y
 A020 290F   AND £$000F       )   MASK MSD OUT
 A022 48     PHA              H   PUSH LSD ON STACK
 A023 98     TYA                  RECOVER ORIGINAL BYTE
 A024 29F0   AND £$00F0      )p   MASK LSD OUT
 A026 4A     LSR A            J   SHIFT TO LS NIBBLE
 A027 4A     LSR A            J
 A028 4A     LSR A            J
 A029 4A     LSR A            J
 A02A 48     PHA              H   PUSH ONTO STACK
 A02B E8     INX              h
 A02C E007   CPX £$0007       '   BCDLEN      END OF BCD WORKSPACE
 A02E D0ED   BNE $A01D       Pm   CONTINUE TILL END
 A030 A00D   LDY £$000D           (BCDLEN*2)-1
 A032 A200   LDX £$0000       "   X = ZERO
 A034 68     PLA              h   UNSTACK 1ST DIGIT
 A035 D005   BNE $A03C        P   JUMP IF NOT ZERO
 A037 88     DEY
 A038 10FA   BPL $A034        z   LEADING ZERO DELETED GET NEXT DIGIT IF
                                  NOT THE END
 A03A 48     PHA              H   IF THE END REACHED ALLOW LAST ZERO
 A03B 68     PLA              h   GET DIGIT FROM STACK
 A03C 0930   ORA £$0030       0   OR WITH ASCII ZERO
 A03E 9542   STA $0042,X      B   ASC         SAVE IN OUTPUT BUFFER
 A040 E8     INX              h   X = NEXT FREE SPACE IN OUTPUT BUFFER
 A041 88     DEY                  ALL UNSTACKED ?
 A042 10F7   BPL $A03B        w   CONTINUE TILL END
 A044 9442   STY $0042,X      B   STORE '$FF' AS LAST BYTE
 A046 60     RTS              '   RETURN
**
* EACH BCD DIGIT IN THE BUFFER IF >= 5 MUST HAVE 3 ADDED TO IT
**
 A047 A207   LDX £$0007       "   BCDLEN
 A049 B548   LDA $0048,X     5H   BCD-1       GET BYTE
 A04B A8     TAY              (   SAVE IN Y
 A04C 290F   AND £$000F       )   MASK OUT MS NIBBLE
 A04E C905   CMP £$0005       I   COMPARE WITH 5
 A050 3003   BMI $A055        0   JUMP IF <5
 A052 18     CLC                  CLEAR CARRY
 A053 6903   ADC £$0003       i   ADD 3
 A055 9548   STA $0048,X      H   BCD-1        SAVE RESULT
 A057 98     TYA                  RECOVER ORIGINAL BYTE
 A058 29F0   AND £$00F0      )p   MASK OUT LS NIBBLE
 A05A C950   CMP £$0050      IP   COMPARE WITH 5
 A05C 3003   BMI $A061        0   JUMP IF <5
 A05E 18     CLC                  CLEAR CARRY
 A05F 6930   ADC £$0030      i0   ADD 3
 A061 1548   ORA $0048,X      H   OR IN LS NIBBLE FROM SAVE AREA
 A063 9548   STA $0048,X      H   SAVE COMPLETE RESULT
 A065 CA     DEX              J
 A066 D0E1   BNE $A049       Pa   REPEAT TILL END OF BUFFER
**
* HAVING DONE THE ABOVE THE NEXT THING TO DO IS TO SHIFT ALL THE
* BYTES BCD AND BIN ONE PLACE TO THE LEFT
**
 A068 A200   LDX £$0000       "   X = ZERO
 A06A A00C   LDY £$000C           BCDLEN+BINLEN
 A06C 18     CLC                  CLEAR CARRY
 A06D 3644   ROL $0044,X     6D   ROTATE BYTE
 A06F E8     INX              h   NEXT BYTE
 A070 88     DEY
 A071 D0FA   BNE $A06D       Pz   CINTINUE TILL END
 A073 F0A2   BEQ $A017       p"   GO SEE IF ALL SHIFTS DONE
```

★★★★★★★★★★★★★★★ LIBRARY ★★★★★★★★★★★★★★★

Due to the Library now taking more room in the Newsletter each month, we shall in future bring you the latest additions only. Anyone requiring an up-to-date listing of the Library should send an S.A.E. for a speedy reply.... **ED**

We pay top brass for top class programmers.

For new additions, see insert in middle of newsletter.

Microprocessor Cassettes (MP15's)

High quality data integrity tape. Recommended for those important recordings.
Packs of **ten** only. £6.00 p & p included.

§§§§§§§§§§§§§§§§§§§§§§§§§§§§§§§§§§§§§

FOR SALE

VDU, KEYBOARD, OFFERS. XBUG £10, SPACE INVADERS ROM £6. WANTED – TANRAM.
RING FRANK (EVES) 01-567 1092

FOR SALE: ELEKTOR SYSTEM – SC/MP,2K NIBL BASIC, 4K RAM; CASSETTE INTERFACE (needs calibrating). This system contains the 'ELEKTERMINAL', an excellent stand-alone terminal in its own right – six baud rates, black-on or white-on video etc, etc. Any reasonable offer considered. If nothing else, bags of components and an excellent power supply. **BOX 111, TUG H.Q.**

PLEASE NOTE . . .

TUG H.Q. is on the phone; we are only too pleased to help you with advice or problems .... give us a Tanring sometime. If Eric answers, don't hang up – just tell him your problems ....

Newsletter Issue 10: Error

In line 7 of the Tanbug V2 Review, 'pins 8, 9 and 11' should read 'pins 8, 10 and 11'.

WANTED

Has any member had any experience with the 'Apple' power supply from 'Henry's' , Edgware Road. Likewise any experience from the 'Cherry' keyboard. This information wanted for a few of our members ..... TUG ON !!

££££ ORIGINAL PROGRAMS WANTED ££££

Business – Educational – Games

We will pay up to £100.00 – Sub-routines also bought.

Second Hand Equipment Bought, Sold & Exchanged

ZACRIDE LTD.

16 IDDESLEIGH ROAD, CHARMINSTER,
BOURNEMOUTH, DORSET. BH3 7JR. 0202 294393

Sponsors of TANGERINE USERS GROUP LTD.

[[[[[[[[[[[[[[[[[[[[[[[[[[[[[[[[[[!!]]]]]]]]]]]]]]]]]]]]]]]]]]]]]]]]]]]

Already this month the Eprom Programmer is out selling all other goods offered by TUG. Quite simply, the package offers an entry into eprom blasting at a low initial cost. Add that to the fact that eproms are becoming cheaper, and a viable situation developes for the average user to programme and store in what must be considered one of the best storage forms available. Since there has been such an enormous response to TUG's 'blaster!', that we shall be supporting the product by bringing to you via the newsletters, various hints and tips to aid your programming, Clear Screen routines, printer routines and such like. As an alternative to Xbug, many other routines can be accommodated on line with Tanbug.

The simple construction makes the programmer atractive to almost all users. If your still not sure, we'll make one for you. The instruction manual includes a p.c.b. component map to indicate the position of components on the p.c.b. tracks. A pin connection diagram is included for the Tanex connection and a circuit diagram is included for the more ambitious who may want to build the unit onto a larger board.

The software programme makes the whole programmer a delight to use, complete automatic control over programming. The question is, Who would be without one!!

Just to whet the appetite.. But don't blow it yet, we've got plans for it..

This little screen clearing routine saves the accumulator and Index Y.

```
0400 48          PHA              ; Save accumulator and
0401 98          TYA              ;
0402 48          PHA              ; Index Y
0403 A9 20       LDA +$20         ; Blank space
0405 A0 00       LDY +$00         ; Get counter
0407 99 00 02    STA  $0200,Y     ; Clear upper screen
040A 99 00 03    STA  $0300,Y     ; Clear lower screen
040D 88          DEY              ; Decrease the count, index Y
040E D0 F7       BNE  $0407       ; Return if not equal
0410 68          PLA              ; Recover Index Y
0411 A8          TAY              ; and
0412 68          PLA              ; Accumulator
0413 60          RTS  (Optional)
```

A library of routines such as this simple aid, the user defined functions are better stored permanently for easy access. Next you'll be calling for our Eprom Extention Card to hold all these goodies.

Amateur Radio enthusiasts note;

G8JHE   Mike Brogan, 11 Trinity Close, Fordham, Ely, Cambs. CB7 5PB.

Additional pages are added to the newsletter simply to bring up to the minute news items to your attention as and when avalable, these pages will develope with more or less content according to the news or articles in process.

Thankfully it seems, T.C.S. are restarting their production lines for Microtan equipment. Of late we have been suffering from some delays on their goods which it appears has been caused with not only the popularity of the Microtan system and supplies, but also with the demand for the prestel adaptor. Perhaps now we may be seeing some of those long awaited modules for our systems. Without any official news releases from them as yet we are unable to comment further on what is going to be available and when.
The word is that we may be seeing an adaptor for the Microtan system that will enable Microtan to the prestel network. More news as it comes in......
There has been no news yet on the expected delivery date for the new Tanbug V manual.

P.C.W. SHOW. September 10th, 11th, 12th.

The show is being held at the Cunard Hotel, Hammersmith, London. details of entry can be found in the current magazines. Tangerine Users Group will be exhibiting in the club section along with many other groups and clubs which have evolved in this new industry. On show will be all the lastest goodies and software so bring the cheque books and leave the wives at home. A point worth mentioning here, contrary to popular belief, the first day of these shows is usually the quietest and far more comfortable to browse around.

As a matter to consider, we are increasing the turnaround period for Tug's goods. We do of course rely on the cooperation of members to assist us by including their registration numbers on all orders to us. This will then allow us to check our records of bona-fide members before dispatch. This system will be of more general importance now that we are presenting our goods to the unfortunate non-members of the Group who, are not entitled to discounts under our present scheme. Subject to stock situations our present turnaround period will be reduced to 48hrs or less on Tug's goods. Postal delays are however another matter.

Although our current software dispatch system has been working very well, the envelope packaging is being replace with padded 'Jiffy' bags, considering the large numbers involved, only the odd library case manages to get destroyed in the mail, . this portion should however be eliminated from now on.

During the last few weeks I have been cooperating with a publisher of a new mag 'MicroDecision'. This magazine is aimed at the middle business market and has as its objective, the practical application of microcomputers in the every day business world. It is not a magazine for enthusiasts as such, however, can be of enourmous benefit to continue where other mags leave off. I feel sure that many of our members could find much of its content refreshing. If your interested in a pilot copy, write, VNU Business Publications 53-55 Frith Street, London. WIA 2HG.

As some of you have pointed out, its nice to have two residences, particularly in the same town, very handy for dopping out of sight for a while. Needless to say I can't get it through their thick heads that TUG moved from its old address some nine months ago. No matter how many times we write to their User Group editors pointing out that we're not that wealthy, they still consider they know better. Still, if it keeps them happy, let 'em play...

Noticed in a mag recently, July issue PCW page 179, something that resembles Tangy's MPS 2 driving a TRS 80 disk unit, interesting, perhaps the're catching on!!

BYTE MASTER joins the TUG team this month with various articles on hardware and software. Any response can be addressed to BYTE MASTER via TUG HQ and we'll pass on relevant correspondence.

Still no news on the Tanbug manual this month, sufficient to say it's been a few weeks away for some time now.

Unofficial news this week has stimulated interest in alternative power supplies for the MPS 2. Now we have been informed that the price of the system power supply has risen to £69.00 odd plus VAT. This means an increase for those contemplating upgrading to an expanded system, understandable in this economic climate that costs should rise, however, perhaps this would be a good time to investigate alternative power supplies for the system. One advertisement seen recently, was Henry's Apple power supply. We would therefore be interested to know if any of our members have had any cause to examine or use this supply or any other that we could bring to the attention of the Group members....

2716's 2716's 2716's

OUR CURRENT PRICE OF THESE POPULAR EPROMS......... £2.95 Ex VAT, P.P.

KEYBOARD - KEYPAD SELECTOR CIRCUIT by Wireman.

When using the Microtan system it is sometimes necessary to change over from the Qwerty Keyboard to the Keypad and vice versa. This I find rather irritating and also the possibility of broken pins on the plugs etc often worry me. The solution was to design a circuit which achieves this change over using a simple toggle switch.

After consulting the Microtan manual and circuit diagram, it was found that the 7 Ascii Data connections (Pins 1-7) on the Keyboard/Keypad socket on Microtan and the Strobe line (Pin 15) needed switching. Pins 9 to 13 are only used by the Keypad. Pin 14 is the reset which goes to both and therefore does not require switching.

Essentially two devices which both generate data on 8 wires have to share the 8 lines coming from the Microtan. This problem is similar to many memory devices on the Microtan data bus. Only one device can use the bus at any one time. In the Selector circuit, this is achieved by using an Octal Buffer I.C. with Tristate capability for each data input device. If a device has Tristate logic it is possible for the gates to offer a third condition at its outputs. This third state is when the normal two logic levels are nor required and the output is put into a high impedance state.

This third state is selected by the logic level applied to the Control pins. In the case of the device used in this circuit, the control pins are 1 & 19. With logic (OV) on these pins the gates outputs act normally; i.e. the outputs present the same logic levels as that applied to the inputs. But with logic one (+5v) on these pins the gates go into the tristate condition. The two 1K resistors present logic one to these pins until the switch connects them to Ov. This is arranged so that if a wire should break on the switch, then either one data entry device would function or none at all.

At any one time only one device will be on, the other will be in tristate condition. The circuit is arranged such that either the Keyboard or Keypad may be plugged into either input socket. There is no reason why two of either may be plugged into the two input sockets.

The output from the selector board is connected via the ribbon cable to the normal socket on the Microtan. Note: When changing from one data entry device to another, it is necessary to hit RESET to allow Tanbug to detect which device is in use.

see over..........

256x256

The enormous interest being shown in our high definition graphic module has prompted us to add a few details to this months supplement. We're not giving away too much detail at the moment, suffice to say, that those who have seen the prototype in action on the system are demanding that production starts at once.

The module is suitable for both the MK1 & MK2 systems. Capable of storing user defined graphics, symbols and CURSOR. Graphical data is not corrupted during screen scrolling in Basic or M/Code. Page selectable facilities are offered for systems with or without Tanram and allows 256x256 representation without the need for large amounts of Ram. The module and concept has been described as 'User Friendly', if you are able to come along and see the system for yourself, we would be only too pleased to demonstrate if for you. Just give us a TUGring sometime.....TUG ON!!!!

Our Girl Val! (anything goes - and generally does!!)

Did you hear the one about Silicon Val, who picked up a hansom chip on the bus and took him back to her rom. After a quick byte together she ran through some of her subroutines before he had to refresh up, get his bits addressed and jump in a hexi home...

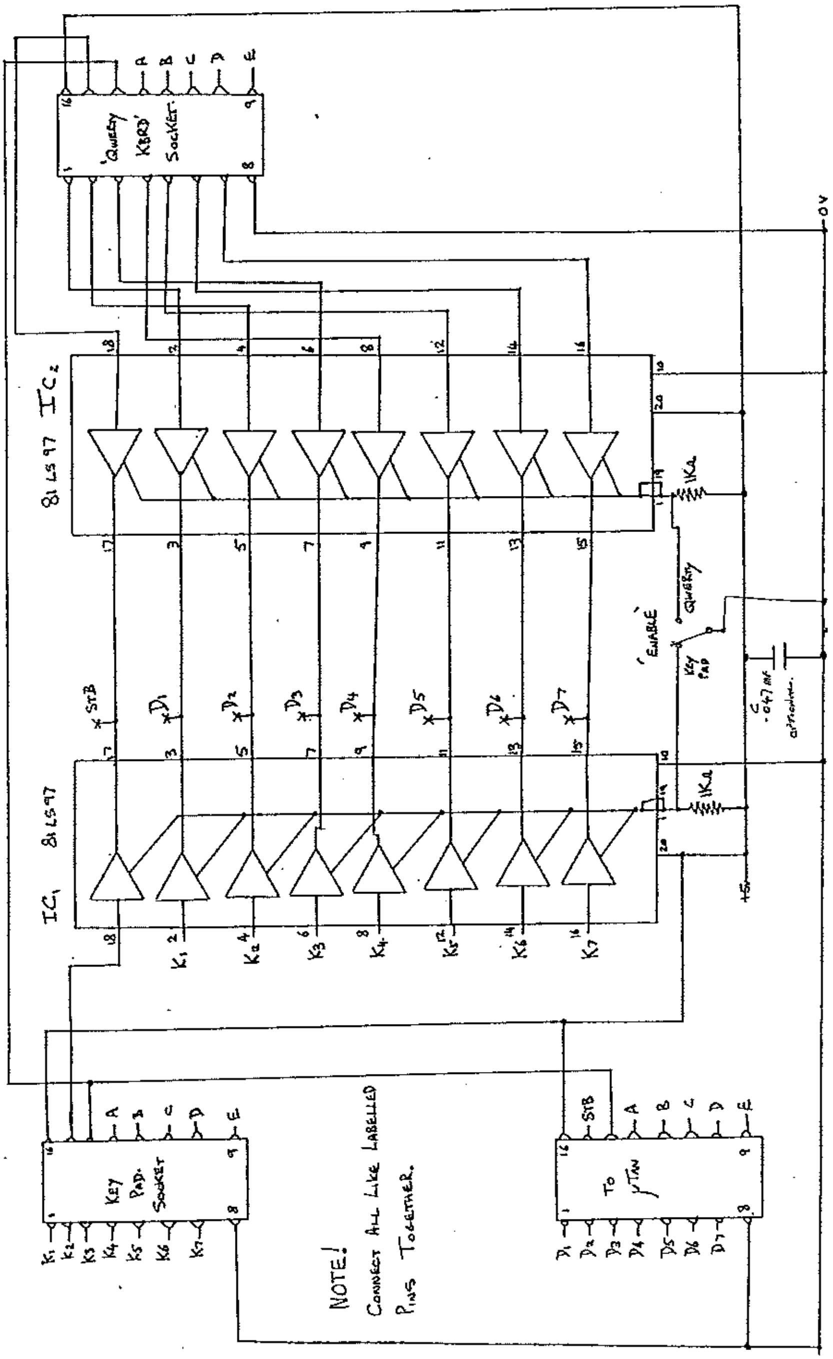
Optomism is only a dream.... I know I keep trying!!

or

The sincerest form of self expression is wishfull thinking ......

ERIC

'Newton and Freud have got it all wrong! the earth is sucking like mad!, anything else is all in the mind!!! Eric '81


'QWERTY KBRD' SOCKET.
IC2
81LS97
IC1
81LS97
1KΩ
1KΩ
'ENABLE'
QWERTY
KEY PAD
C .047 nF optional.
+5
0V
STB
D1
D2
D3
D4
D5
D6
D7
K1
K2
K3
K4
K5
K6
K7
A
B
C
D
E
KEY PAD SOCKET
TO μTAN
NOTE! CONNECT ALL LIKE LABELLED PINS TOGETHER.