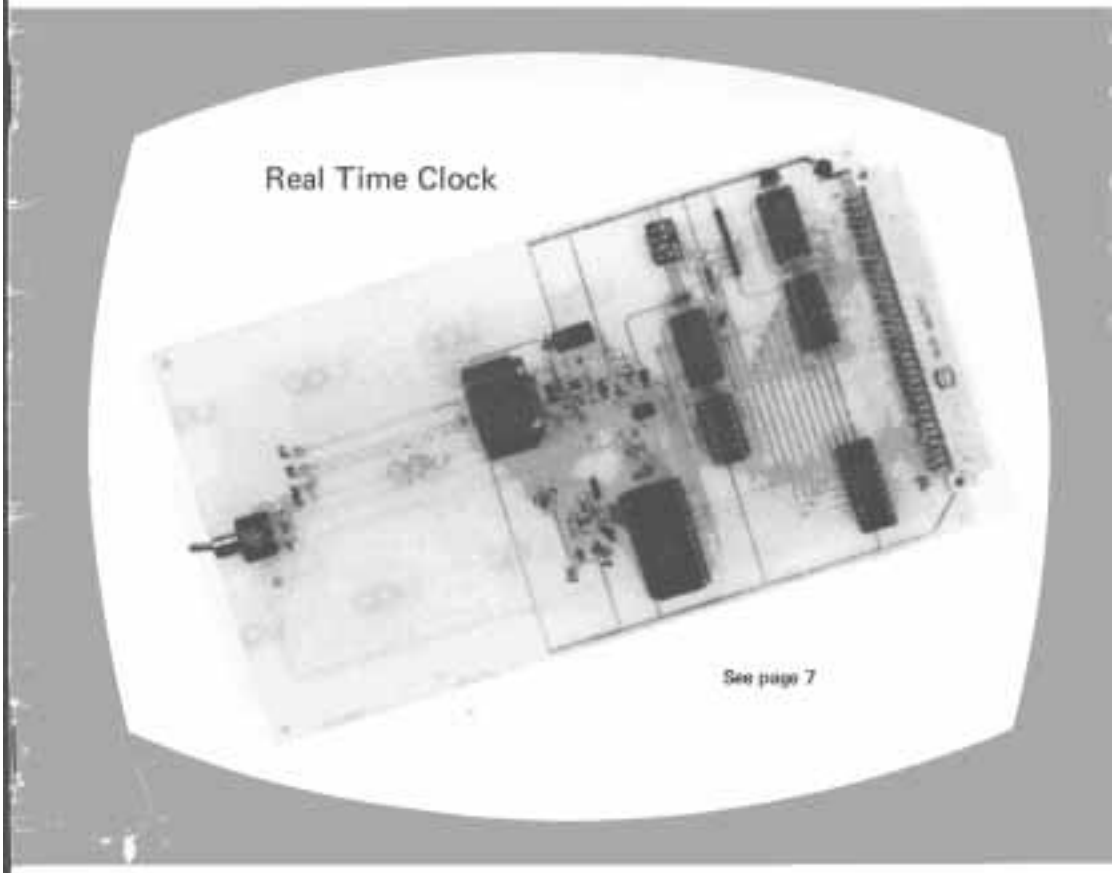
MICROTAN WORLD

Volume No. 1 | Aug.– Sept. 1983 | Issue No. 3

Programs · Reviews · Your Letters



See page 7

MICROTAN IS EXPANDING!

PUBLISHERS
MICROTANIC COMPUTER SYSTEMS LIMITED
16 UPLAND ROAD, DULWICH, LONDON SE22. 01 693 1137

MICROTANIC
COMPUTER SYSTEMS LTD.

REGISTERED OFFICE
235 FRIERN ROAD
DULWICH
LONDON SE22
TEL 01 693 7659

MICRO COMPUTERS
CONSULTANTS
SOFTWARE
SERVICES
RENTALS
HARDWARE

SHOWROOMS
16 UPLAND ROAD
DULWICH
LONDON SE22
TEL 01-693 1137

SOFTWARE FOR THE MICROTAN SYSTEM

MICROTANIC SOFTWARE PRODUCT LIST - SEP 1982

ADVENTURE 1	7K M/C	5.95
ADVENTURE 2	7K BAS	6.95
ADVENTURE 3	16K M/C	9.95
ADVENTURE PACK	3 x 7K BAS	14.95
THE DEFENDER	7K M/C	6.95
TANK RAID	16K M/C	9.95
EARTH ATTACK	7K M/C	6.95
SPACE INVADER	3K M/C	5.95
THE GOBBLER	6K M/C	6.95
SUB STRIKE	5K BAS	5.95
STAR CHESS	3K M/C	6.95
3D MAZE	7K BAS	6.95
3D OXO	7K BAS	6.95
GRAPHIC PUZZLES	7K M/C	6.95
MOLE SWAT	7K BAS	6.95
GAMES PACK 1	3 x 7K BAS	8.95
GAMES PACK 2	3 x 7K BAS	8.95
GAMES PACK 3	3 x 7K BAS	8.95
GAMES PACK 4	4 x 7K M/C	8.95
AUTHOR	7K M/C	6.95

TOOLKIT 1	2K EPROM	22.50
MICROTANTEL	2K EPROM	16.95
HI RES TOOLKIT	2K EPROM	16.95
FORTH LANGUAGE	IN EPROMS	34.95
2 PASS ASSEMBLER	EPROM	34.95
TOOLKIT 2	2K EPROM	16.95
ASIMOV WORD/PROC	EPROMS	49.95
FILE UTILITIES	BAS TAPE	9.95
M/C RELOCATOR	M/C TAPE	5.95
EPROM PROGRAMMER SELF BUILD		9.95
TEXT PROCESSOR	M/C TAPE	14.95
BAS LINKED ASSEMBLER	TAPE	6.95
MORSE CODE GENERATOR	TAPE	4.95
SINCLAIR PRINTER INTERFACE		29.95
DISPLAY PLANNER	M/C TAPE	12.95
E2 EPROM EXTENSION CARD		22.50
H2 EPROM EXTENSION CARD		24.50
25 x 64 COLOUR VDU BOARD		85.00
LOOSELEAF DATA BASE	TAPE	19.95
MUSIC COMPILER FOR BULLDOG		19.95

EPROM PROGRAMMER CARD FULLY BUILT TO PROGRAM 2716-2732-2516-2532 EPROMS 69.95

UNIVERSAL EPROM ERASER FULLY BUILT TO ERASE UP TO 8 EPROMS AT A TIME 34.95

HI RESOLUTION TOOLKIT NOW AVAILABLE FOR OWNERS OF THE BULLDOG BOARD 16.95

TOOLKIT - ADVENTURE GAME - 3 PASS ASSEMBLER AVAILABLE FOR NEW SCREEN BRD POA

PILOT LANGUAGE NOW AVAILABLE FOR THE MICROTAN - THIS EXTENSION TO BASIC WILL ENABLE YOU TO WRITE AND RUN COMPUTER ASSISTED LEARNING(CAL) AND COMPUTER ASSISTED INSTRUCTION(CAI) PROGRAMS ON YOUR MICROTAN - (TAPE OR EPROM) 19.95

LARGE SELECTION OF COMPUTER BOOKS NOW AVAILABLE FROM STOCK - WRITE FOR DETAILS

If You have any programs or Hardware ideas please Contact Mr David Northway

ALL PRICES INCLUDE V.A.T. BUT PLEASE ADD 00.60p FOR POST AND PACKING

SHOWROOM:
16 Upland Road
Dulwich, London SE22

MAIL ORDER:
235 Friern Road,
Dulwich, London SE22

TELEPHONE: 01-693 1137

Contents

WHO'S WHO AND WHAT'S WHAT

The Company

Microtanic Computer

Microtanic Computer Systems Ltd. 16 Upland Road, Dulwich, London SE22. 01 693 1137

Managing Director	David Northway
Director	Charles Cooper
Director & General Manager	Robin Northway
Design Consultant	Dr. P.N. Gaunt

Microtan World

Published by	Microtanic Computer Systems Ltd
Editor	Deryck M. Sutton
Repairs & Technical	Andy Parnell & Clive Reynolds

The Microtan System

1K Computer in Machine Code

Microtan 65 + Keypad + u/l case + Graphics

8K Computer Expansion

As above + Tanex + Mini Motherboard + Power Supply + Full ASCII Keyboard + Basic

48K Computer Expansion

As 8K + Tanram + MPS2 Power Supply + System Motherboard.

Additional Expansion

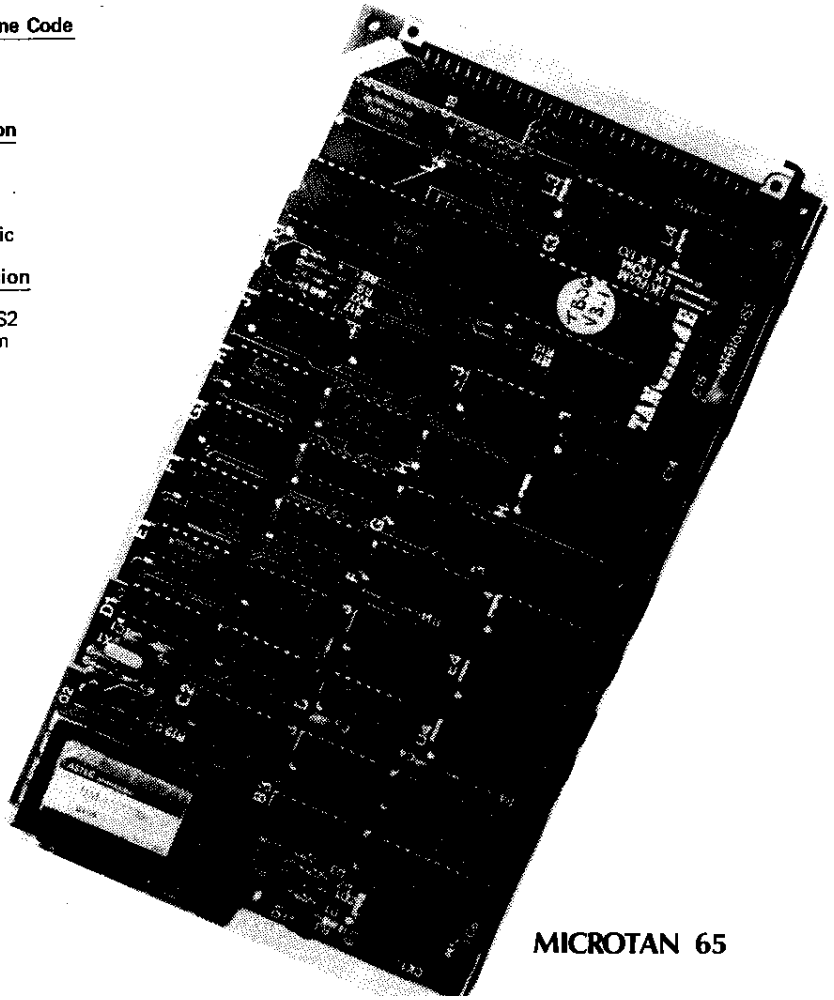
HI RES Board. VDU Colour Board. Sound Board. DOS Board. MASS Storage Eprom Board

Languages

Basic. Forth. Pilot Assembler

Also Available

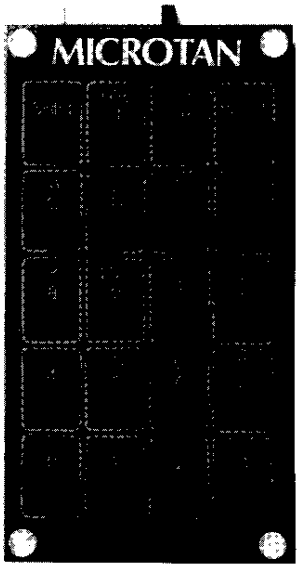
Microtutor Controller Board



MICROTAN 65

The Inside Story

by David Northway



Here we are with Issue No. 3 and that means the months are really slipping by. We are now getting more and more comments from you and we will try and use all your ideas wherever possible. Keep those letters coming in. We are continuing to expand and we have added a new overseas agent (see Shop Window) this time in Israel.

We are also getting more programs and articles and a voucher system is under way to tempt you to write for the magazine.

We are working on new software and also on providing disc based software. The real time clock is available so give us a call for details. We can also tell you that we have a new board 'Tandata' under design, but more of that another time.

From the Dr Gaunt articles you will have seen that we are expanding the system continuously and wherever possible this expansion will be kept compatible with existing hardware. I can also assure you that the original system will always be fully supported.

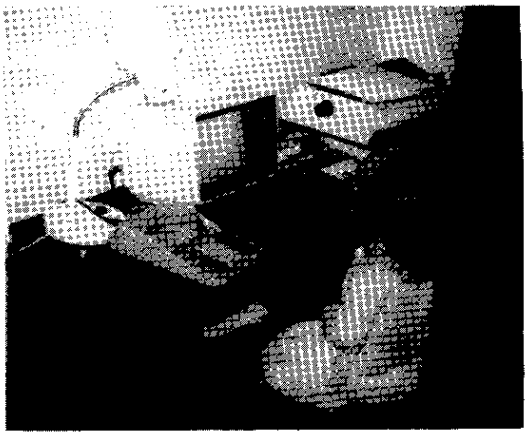
Finally, whenever possible, comments on the system etc should be sent to me at MCS Ltd and not to the Editor, this way I will be able to react to them more quickly and with less danger of them being overlooked. Comments on the magazine should be sent to the Editor.

More news in October

ED'S Page

To echo David's words, Issue No. 3 already. We are getting a good feedback from you now and we will try and reflect your wishes into the magazine.

We now have a voucher system which we hope will attract you to keep those articles coming.
The vouchers will be for £5 each if cashed or £8 if used for purchases from MCS Ltd.
The amount paid will vary according to the size and content of the article and the voucher will be issued after publication. If you wish to know in advance of publication the value of an article, please state clearly when writing in and we will advise you and wait for your confirmation.



The Editor at work!!!!

If you are submitting programs for publication please try and include a tape for checking, this will also enable us to produce a listing in a quality suitable for printing. (tapes will be returned).

To all disc users . . . we are looking for your views and comments on using the disc system and in particular hints and tips to overcome any problems you had to start with. Also any short programs which illustrate their use.

Thank you once again for all your kind words, for your suggestions and for your articles. We particularly need short programs, hints and tips etc. and although these may not qualify for payment, you will have your name in print!.

REAL TIME CLOCK/CALENDER CARD FOR THE MICROTAN

Dr P N Gaunt

A real time clock should provide two functions within a microprocessor system: it should be an accurate 'watch' for the computer to refer to when the time of day or date is required, and it should be capable of interrupting the processor at precise intervals when periodic sampling is undertaken. The latter function is correctly referred to as the 'real time clock' capability whilst the former is the 'calendar'. This card handles both of these requirements, and in addition contains 50 bytes of battery-backed RAM that can be used for any semi-permanent storage. The 146818 real time clock used on this card provides all of these facilities and occupies any 64 byte block of 1/0 space. Since the device is so versatile and also little known, the following brief description may be of interest.

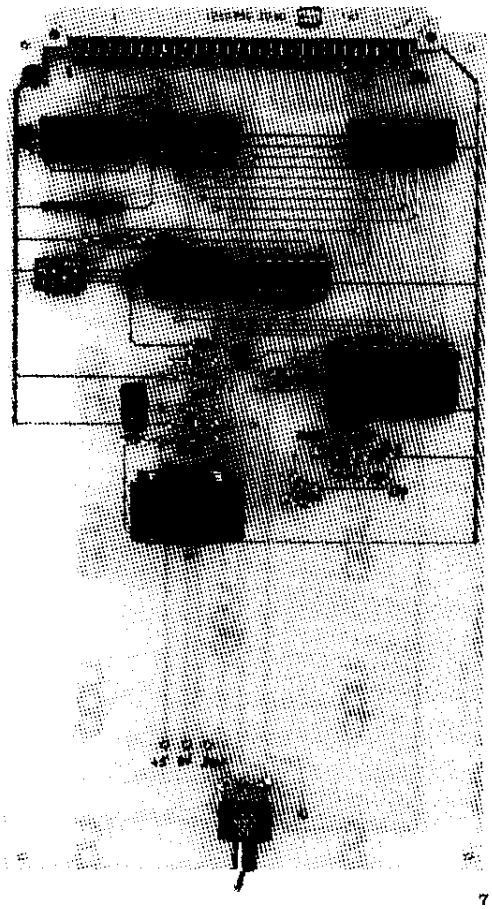
The clock counts from seconds to hours and has a hundred year calendar with day of week, date, month and year (with automatic leap-year adjustment). There are three interrupt modes, each software selectable, for a time-of-day alarm from once-a-second to once-a-day, continuous periodic interrupts from 122 microseconds to 500 milliseconds, or on the completion of the clock update-cycle. A switch is provided on the board to write protect the clock and RAM so avoiding accidental corruption - it is recommended that the board is write protected except when setting the clock or changing RAM data. The on-board NiCad rechargeable battery can maintain the clock and data of the chip for weeks with the computer turned off and recharges when the computer is on. Of the 64 bytes of I/O space required, 14 are clock registers and the remaining 50 are CMOS static RAM that may be used for any purpose. The memory map of the device is as follows:

xx00	Seconds
xx01	Seconds Alarm
xx02	Minutes
xx03	Minutes Alarm
xx04	Hours
xx05	Hours Alarm
xx06	Day of Week (1-7)
xx07	Day of Month (1-31)
xx08	Month
xx09	Year
xx0A	Register A - U1P, RS0-3
xx0B	Register B - SET,PIE,AIE,UIE,SQWE,DM,24/12
xx0C	Register C - IRQF,PF,AF,UF
xx0D	Register D - VRT
xx0E-3F	Uncommitted RAM (50 bytes)

(xx may be switch selected to any 64 byte boundary in I/O space).

Registers A & B are read/write control registers: U1P is set when time registers are being updated (and potentially invalid), RS0-3 select the periodic interrupt interval, SET disables the clock if set so that time may be set accurately, PIE, AIE & UIE enable interrupts by period, alarm or end of update, SQWE enables square wave output (should be off unless required), DM defines whether registers are binary or BCD mode, 24/12 select 24 or 12 hour clock. Registers C & D are read-only status registers: IRQ, PF, AF & UF are the interrupt flags while VRT indicates whether a power failure has occurred that may have corrupted data in the device.

These notes give only an outline of the capabilities of the versatile clock device and more comprehensive details and programming examples will be provided in the user manual.



VIEWPOINT

Dear Mr Sutton,

I see that you published my previous letter in the 'Microtan World', well I have solved the auto-repeat method since and include the copy for publishing in future issues.

I have a few more comments on the 'Microtan World' content. First where does Dr P.N. Gaunt fit into the Microtanic scene? He mentions TBUG 09, does this actually exist and if so how do I obtain it. There also appears to be slight confusion as to where the D.O.S. board should reside in the system set up, but I am still waiting for my disc controller board so it is academic at present.

This brings me onto the next point. The printing of the circuit diagrams is very useful indeed. I do hope this is just the beginning of the complete series. Setting up the system Nine needs circuit diagrams to allow the full implications to be understood. I had not realised that there was a 74LS244 bugler on the mother board for instance.

Now how about Simple Simon? I would have thought that all owners of the microtan would be beyond this, after all the standard microtan is well out of the price range of the new first time buyer. They must be going for Spectrum's ORIC's (dare I mention them) or Dragon's, VIC 20's etc. In any case all electronics and computing enthusiasts I know spend out on several magazines. These run articles for Simple Simon fans. We want information particular to microtan that cannot be obtained elsewhere.

Finally perhaps you could tell me how many microtans there are in existance, I have often wondered.

Yours sincerely
M Marten
Southampton

Dear Sirs,

I recently sent you a few items for possible inclusion in 'Microtan World'. Included was a method of altering Tanram to run in any slot, as my nother board was unable to produce the correct BE signals. After writing to you I decided to correct the mother board fault and I am enclosing the method I used which maybe of help to other readers.

For issue 3 mother boards:- PIN 15 OF the 74LS157 and PIN 5 of the 74LS138 were left floating - for correct operation these should be taken to OV.

Tanram may now be used in any slot by enabling that slot prior to entering basic. For example for slot 2 type MFFFF, XX,22 return then enter basic.

Yours faithfully,
T K Jones
Sunderland

Dear Sir,

When I came to see you on Saturday 7th May you said you would welcome any comments about the MICROTAN computer, well here goes.

ASIMOV
How about incorporating right hand justification. Useful for single/multiple page printing eg:-

Start page? Stop page?

This will enable printing single pages of documents that have had minor corrections to it.

TANSOFT TWO PASS ASSEMBLER
Increase tape names to 6 letters.
Change print routines from

Line No Source/Comment Address Object code

to

Line No Address Object code Source/comment

This will allow you to use the full width of paper and all more room for comments.
How about extending the assembler to have a compatible disassembler thus allowing you to write source code for existing programs.

PROGRAMS
A program to compress BASIC programs by deleting spaces and REM statements (Incorporate into the Toolkit?)
PASCAL!!

HARDWARE
Hi-res VDU board with say 512 * 512 resolution and 16 colours. This should also give 80 column display. With on board screen memory. Dreams !!!!!!

MICROTAN WORLD
Congratulations, Presentation 10, Content 10, keep up the good work.

Please continue BASIC by SIMPLE SIMON but it should show how to write good clear programs, eg. Do NOT use?for PRINT and leave spaces inbetween statements.

```
10 PRINT CHR$ (12) : PRINT 'HELLO'
```

is much easier to read and understand than

```
10?CHR$ (12) : ? 'HELLO'
```

although the latter takes less room, for most programs it should not matter and you can come back to the program a year or so later and still be able to read it.

IDEAS FOR ARTICLES

It would be useful to know what improvements have been made in the past to MICROTAN hardware and software ie what's the difference between

MICROTAN 65 Issue 1, Issue 2, etc.
TANEX Issue 1, Issue 2, etc
TANBUG V1, V2,3, V3
XBUG V5, V5,2 etc

I have just received the MOUSEPACKET DESIGNS EPROM SWITCHING BOARD and found it was easy to construct from the instructions given, it also works perfectly and I have found it a nice addition to my system.

I was looking through your cataloug sheets and found a mention of TOOLKIT 2 in EPROM but no explanation of what it does. I have an issue 6 System mother board and would like to know if this has facilities for memory pageing, if not is it possible to convert it and if so how?

Anyway that will do for now. I look forward to the next issue of MICROTAN WORLD and wish you and everybody connected with MICROTANIC every success for the future.

Yours faithfully
E Pittuck
Harlow, Essex

Dear Deryck,

First of all, let me congratulate you on the quality of your new magazine. I found the second issue even more interesting than the first!

One of my comments on the system is: could MCS not combine the obsolescence of the 32 x 16 screen with the snail-pace 0.75 MHz microprocessor clock frequency to produce two new boards (one could already be fulfilled by the SCB), a colour 24 x 40 + Hires VDU card (with UHF, Pal Composite Video and RGB output formats) and a multi-speed processor card (with just 1K RAM for stack & processor 'scratch-pad'). The VDU card would operate at several programmable clock speeds, programmed by wire-links on the PCB. Speeds catered for could be: 0.75, 1.0, 1.5, 2.0, 3.0 MHz. The card would not need a micro to itself, as there are plenty of cheap chips around (Prestel and Viewdata use the 24 x 40 standard) that will do the alph-numeric character generation plus discrete LS TTL (similar to the current Hi-Res card) to generate the colour graphics pixels. 8K (1 x 6164 CMOS RAM chip) would be sufficient. The processor card, with on-board very fast 1K RAM, would be 6809/6502 compatible and have wire-link programmable clock frequencies in the same steps as above. If the designer felt it worth-while, an automatic dual-speed clock could be used (hard - or soft-ware controlled).

Obviously the number one crushing problem is going to be the operating speeds that the other system cards (Tanram, HRG) will cope with. The most obvious way around this problem, using both hard- and soft-ware, is to run the processor at 1.5 MHz, twice the current clock freq., but only drive the Ø 1 and Ø 2 bussed signals on the System Mother Board at 0.75 MHz and insert a single cycle wait during any slow I/O access. All cards capable of operating at 1.5 MHz would obtain their clocks from two new 01 and 02 lines, from two of the spare unallocated lines on the Mother Brd. bus. The hardware would now function perfectly (although effective clock freq. will be lowered slightly) and the remainder will be up to the software engineer to get right. Simple really!

Congratulations on the Eprom Storage Card and I look forward to reading reviews on the RTC and Sound cards.

I would like to join David Price of Tooting (pg 13) and support his request for more software in the magazine. Short as well as involved routines would be welcome.

May I reply to E Jewell's letter on pg 14. Regarding the missing top line of his system screen memory (ie although it exists, it does not appear on his monitor/TV), the effect is caused by a slow TTL 74LS393 counter on his Microtan. Although these devices are supposed to operate at up to 30 MHz, some makes can't make it at just 6 MHz! The culprit sits in D2 and should be replaced with another 393. If E Jewell is lucky, he may get away with simply changing over the 393's in D2 and D3, although I would say this is unlikely because both chips are likely to be of the same manufacturer. I feel sure that you, Deryck, could produce a circuit for E Jewell showing him how to wire up some relays for tape-recorder control (there's one in the Tanex manual!!).

The above problem jogged my memory on the RESET 'mod' that I used to do from time to time. The sympton is that some makes (ie Signetics etc) of 6551 UART would not accept any control instructions. The cause of this is that the chip is not resetting. The RESET signal is derived from a CR combination on the Mother Board (top left corner) and does not have a fast enough rising edge to reset all chips. The solution is simply to buffer the CR-derived reset through one of the three unused Schmitt input buffers of the 74LS244 chip on the System Mother Board. Could you produce a circuit diagram of this for your readers Deryck?

You may like to add my callsign to the list of callsigns on pg 21. Mine is G8UGR and I have tried (unsuccessfully, as yet) using my Tangerine system in amateur radio. My colleague Nick G4FAT has written a Morse Tutor program on my Oric 1.

Andre Margas
Cambridge

The following article is a Preview of a New Software Package

ASKLES

This is a softward package of three programs, an Assembler, and Editor and a LinKer - crossword enthusiasts will spot the anagram. Its purpose is to assist those who would like to write programs in Machine Code.

It is virtually impossible to sit down and write a program directly in Machine Code, one must first write something a little more intelligible to human beings, which of course is Assembler Language. Since the Machine only operates in Machine Code the Assembly Language has to be translated. X Bug's T Command is of some assistance, in that it will convert the instruction muemonic, but one still has to enter the rest of the instruction in absolute terms, and there in lies the problem. The implication is that one must know at the outset where all the instructions are to be located in the machine, how else could one write BCC1468 - sorry BCC $ 1468, and don't forget the space before $! So having written it all about and deciding where all the instructions will be located, one enters it into the machine - it doesn't work! Why? for one thing an instruction has been left out! This means that one must first modify all branch instructions that jump over it, and re-enter the program from the insertion to the end. - what a bore, and what a chance for the preverbial BUG to make its entrance and produce another error that was not there in the first place.

So why ASKLES? First, one can forget all about absolute addresses; Assembler and Linker between them will do the calculations. Next, for the instruction that was left out - that's EDITORS job, either in Assembler Language or Machine Code just as you prefer.

These programs were used to generate themselves. The first, and perhaps the most useful is LINKER. The first version of which had to be produced the hard way as just described. It was the experience gained in writing LINKER that indicated the requirements of EDITOR. LINKER was then available and made the writing of EDITOR so much easier. Later they both assisted in writing ASSEMBLER. So the first two at least have been well tested.

LINKER is a program that allows one to write short sub routines of handleable size, say what one can conveniently write on a sheet of foolscap. Each of these short sub- routines should be tested independently and proved to be satisfactory. They are then each given a name. If one of them writes a steering routine, that is a program that sets up entry conditions and calls to these sub-routines, using these names, LINKER will construct a working program. It links them all together, substituting absolute addresses for the given names.

The steering routine is searched and a list of sub-routines mentioned is made, then a library is searched and copies of the required S.R's are added to the end of the steering routine, and now form part of the program. This process is repeated until no new S.R. are found. So sub-routines can call on another to any depth.

There is also a way of making a Table of data look like a subroutine so that these too can be linked into the program.

EDITOR is a program that allows one to manipulate the contents of Memory either in ASC11 characters or in Hexadecimal. Two addresses are set up, a FROM and a TO, which can be changed at any time and also the Mode - 4 for Hex and 8 for ASC11, which also can be changed at any time. One can consider the screen as working space, anything that one does to date on the screen is not immediately repeated in the store, as it is with the M command in Tanbug, so how does it work? The screen is divided into two halves, the left half is From and the right half To. Data can be caused to flow up the left hand half, from the input address which is updated as it happens, anything disappearing off the top is lost. Data may be prepared on the right hand side and scrolled up the screen. Anything disappearing off the top goes to the To address, which again is updated as it happens. Then there are commands to manipulate what is on the screen. There are two pointers, one on each side, which can be flicked from side to side to indicate which side is currently active.

Data can be made to flow from the input address up the left side, across, and up the right hand side to the To address, by holding down the repeat key. The flow will stop if a byte containing a preset valve appears at bottom left, or top right. So a search is possible.

ASSEMBLER is more mundane, its job is to interpret the Assembler muemonics and calculate jump offsets within each sub routine. The jumps may be indicated by means of labels, or directly as a relative offset. Assembler will handle a whole library of sub-routines, as required by LINKER, at one pass, including of course the steering routine that controls the calls.

Whilst these programs operate independently they are compatible, that is to say that the output from one is acceptable to the others.

L Morgan
Scunthorpe

PRICE LIST

PLEASE SUPPLY THE FOLLOWING:-

DESCRIPTION	PRICE	P/P	QTY ORDER	TOTAL PRICE
MICROTAN 65 KIT	79.35	1.50		
MICROTAN 65 ASSEMBLED	90.85	1.50		
MICROTAN BARE MINIMUM KIT	25.00	incl		
LOWER CASE OPTION	10.90	0.50		
GRAPHICS OPTION	7.50	0.50		
20 WAY KEYPAD	11.50	0.50		
MINI MOTHERBOARD	11.50	0.50		
TANEX (MIN CONFIG) KIT	49.45	1.50		
TANEX (MIN CONFIG) ASSEMBLED	60.95	1.50		
EXPANDED TANEX KIT	103.16	1.50		
EXPANDED TANEX ASSEMBLED	114.66	1.50		
10K MICROSOFT BASIC IN EPROM	56.35	0.75		
XBUG	19.95	0.30		
TANBUG V3.1 IN 2716 AND MANUAL	19.55	0.50		
TANBUG REPROGRAMMING	17.83	0.50		
TANBUG V3.1 KIT FOR ISSUE 1 MICROTANS	21.85	0.50		
MPS1 POWER SUPPLY	26.45	2.00		
±12 VOLT KIT	9.20	0.50		
T.V. LEAD	0.80	0.20		
MICROTAN MANUAL	5.00	incl		
TANEX MANUAL	5.00	incl		
BASIC MANUAL	5.00	incl		
XBUG MANUAL	5.00	incl		
SYSTEM FILE BINDER	2.50	incl		
ASCII KEYBOARD	69.95	1.50		
ASCII CASE	23.00	2.50		
1K RAM (2 x 2114)	5.98	0.30		
SERIAL I/O KIT	17.25	0.30		
6522 VIA	9.20	0.30		
PRINTED CIRCUIT BOARDS	21.85	incl		
MPS2A POWER SUPPLY	79.50	2.00		
TANRAM II 48K ASSEMBLED	93.50	1.50		
TANRAM II 64K ASSEMBLED	113.85	1.50		
MINI RACK	56.35	2.50		
SYSTEM MOTHERBOARD (4 CONNECTOR)	44.85	1.50		
SYSTEM MOTHERBOARD (12 CONNECTOR)	72.85	1.50		
TOTAL:				

DESCRIPTION	PRICE	P/P	QTY ORDER	TOTAL PRICE
EXTRA EDGE CONNECTORS	3.50	0.30		
SYSTEM RACK	67.85	2.50		
SYSTEM RACK FRONT PANEL	15.64	2.00		
MINI RACK MICRON	395.00	incl		
SYSTEM RACK MICRON	550.35	incl		
HIGH RESOLUTION GRAPHICS	90.85	1.50		
AIM/KIM BUFFER	54.64	1.50		
*SYSTEM CONTROLLER BOARD WITH SERIAL I/O, 2K RAM, 6522A				
TYPE 6502A (2MHz)	99.00	1.50		
TYPE 6802 (1MHz)	109.00	1.50		
TYPE 68B09 (2MHz)	119.00	1.50		
*SYSTEM CONTROLLER BOARD WITH SERIAL I/O, 8K RAM, 2 x 6522A				
TYPE 6502A (2MHz)	125.00	1.50		
TYPE 6802 (1MHz)	135.00	1.50		
TYPE 68B09 (2MHz)	145.00	1.50		
SERIAL I/O BOARD MIN (2 PORTS)	66.70	1.50		
SERIAL I/O BOARD MAX (8 PORTS)	135.70	1.50		
PARALLEL I/O BOARD MIN (16 LINES)	54.63	1.50		
PARALLEL I/O BOARD MAX (128 LINES)	99.48	1.50		
32K ROMBOARD (EXCL. ROM)	54.65	1.50		
DISK CONTROLLER BOARD (NO PROMS)	113.85	1.50		
TANDOS 65	40.25	0.50		
1 x 40 TRACK SINGLE-SIDED DISK DRIVE AND CABLE	251.85	6.00		
2 x 40 TRACK SINGLE-SIDED DISK DRIVE AND CABLE	458.85	7.00		
DISK CONNECTING CABLE WITH PLUGS FOR 2 DRIVES	17.25	1.00		
CENTRONICS PRINTER CABLE TWO PORT FOR TANBUG V2.3	23.81	0.50		
ONE PORT FOR TANBUG V3.0	17.25	0.50		
DIP-DIP RIBBON CONNECTOR	4.60	0.50		
ORIC 1 - 48K	169.95	5.95		
TOTAL:				

ORDER FORM

All prices include VAT.

VAT invoices available on request.

Please add packing & Postage as shown.

Please print clearly.

Name..... .....

Address..... .....

....

....

Tel.....

Please send me the goods listed.

Access No.....

Cheque or Postal Order No..... .....

Send your order to.

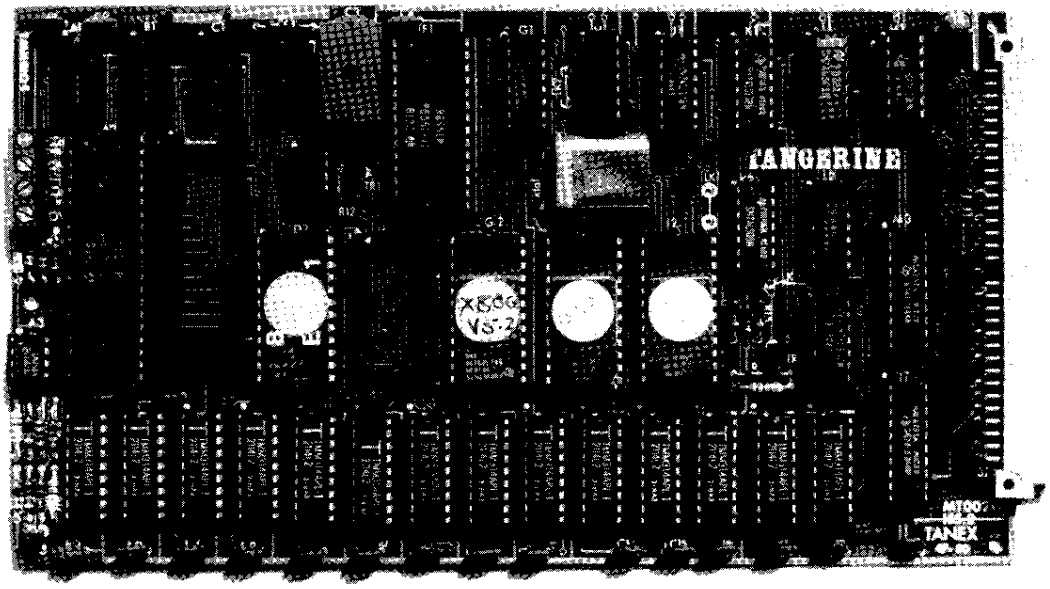
Microtanic Computer Systems Ltd.

16 Upland Road.

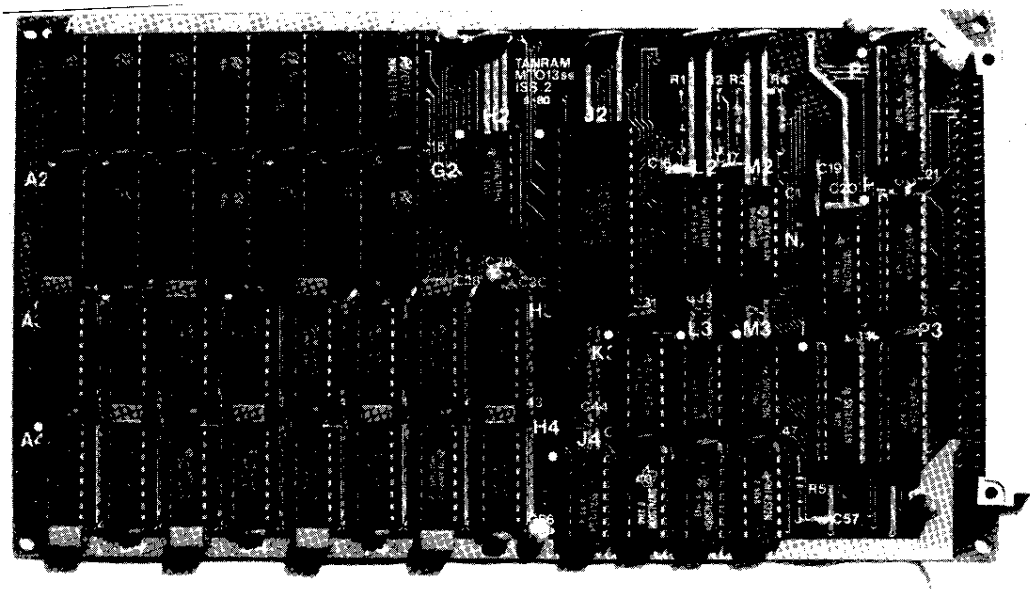
Dulwich.

London. SE 22.

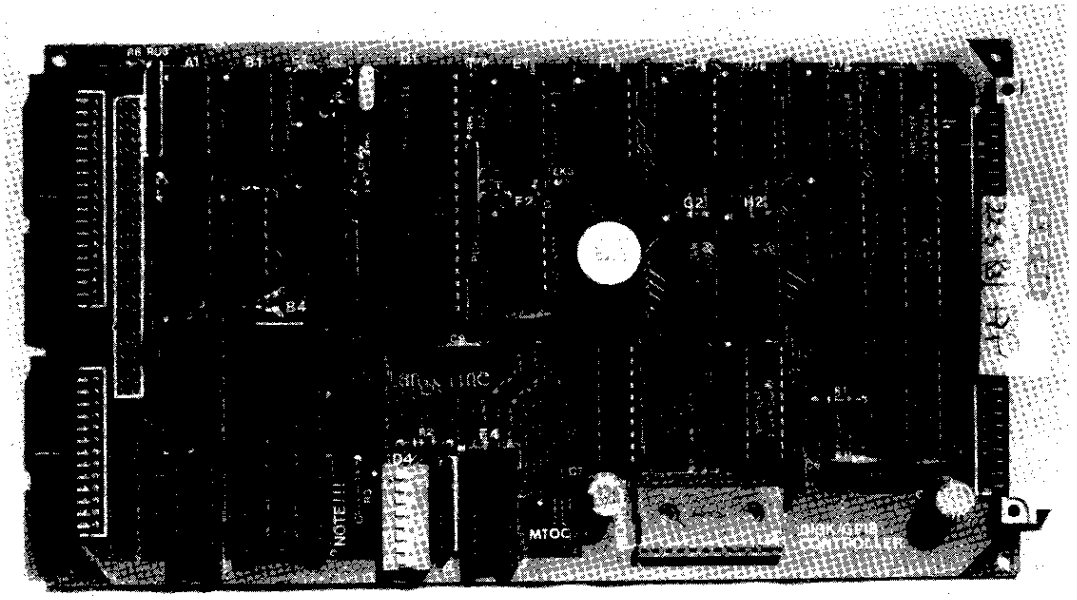
Thank you for your Order.



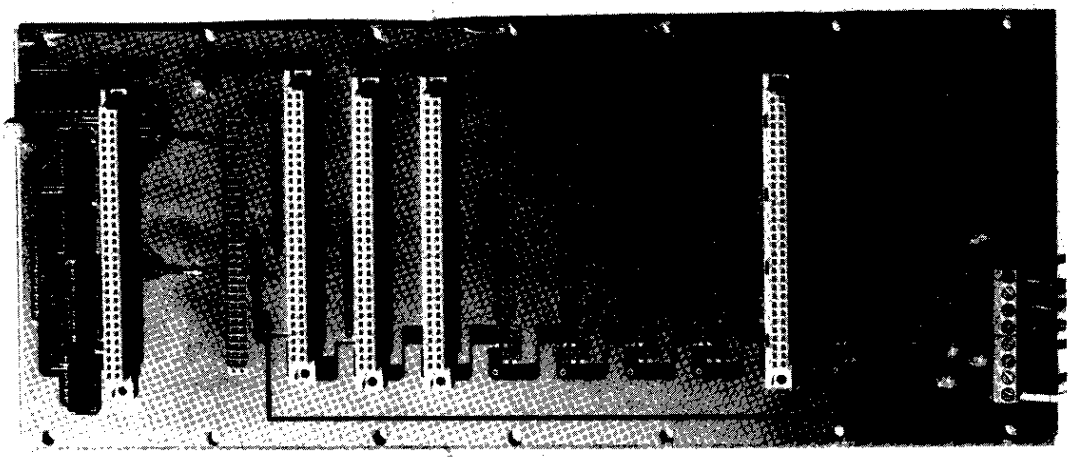
Tanex - First Expansion Board



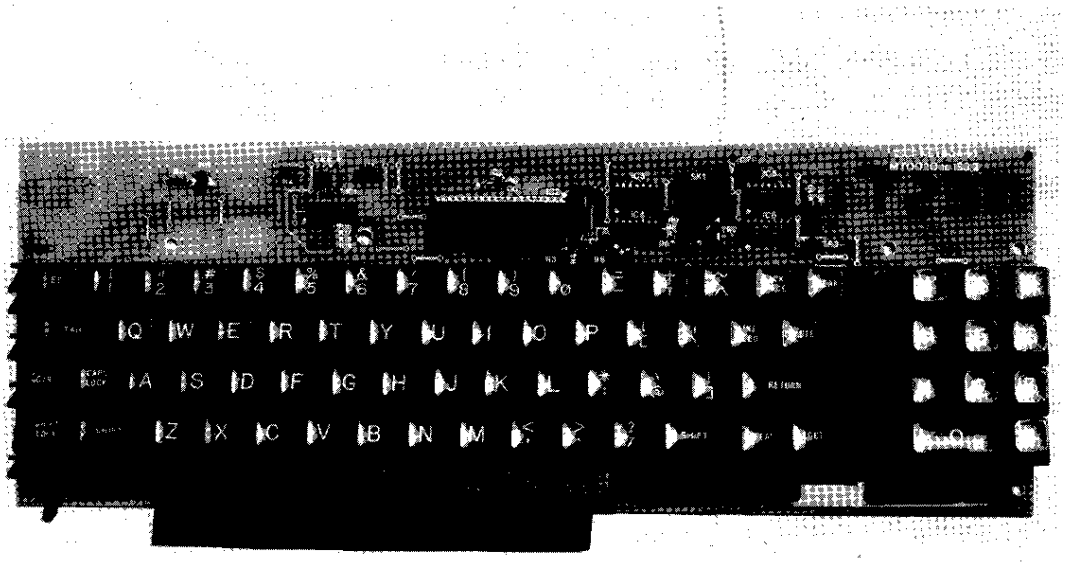
Tanram



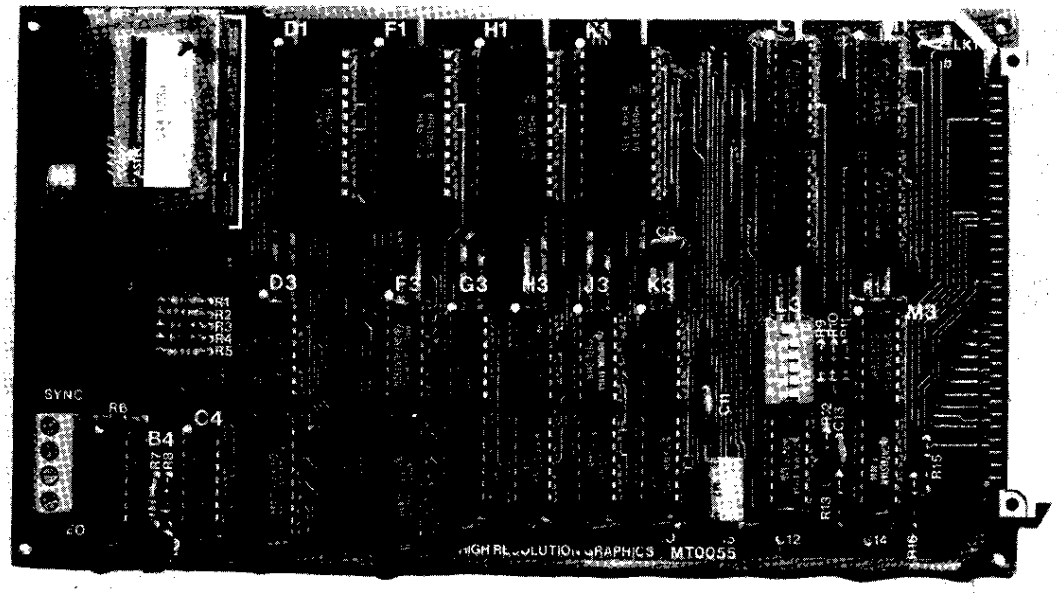
Disc Controller Board



System Mother Board



Keyboard



High Resolution Graphics Board

SHOP WINDOW

Main Microtan Dealers

Viscount Computer World.
2A Boulton Road,
Southsea,
Hants.
Tel: 0705 833633

Comsolve Ltd,
53 Charlestown Road,
Glossop,
Derbyshire,
SK13 8LB
Tel: 04574 66209 (24 Hrs) 061 428 8562

Waltham Forest Service Centre
889 Leabridge Road,
Nr. Whipps Cross,
Walthamstow. E 17.
Tel: 01 520 7747

Micron Computer Equipment Ltd.
Devonshire House,
Devonshire Square,
Loughborough,
Leicester. LE11 3DW

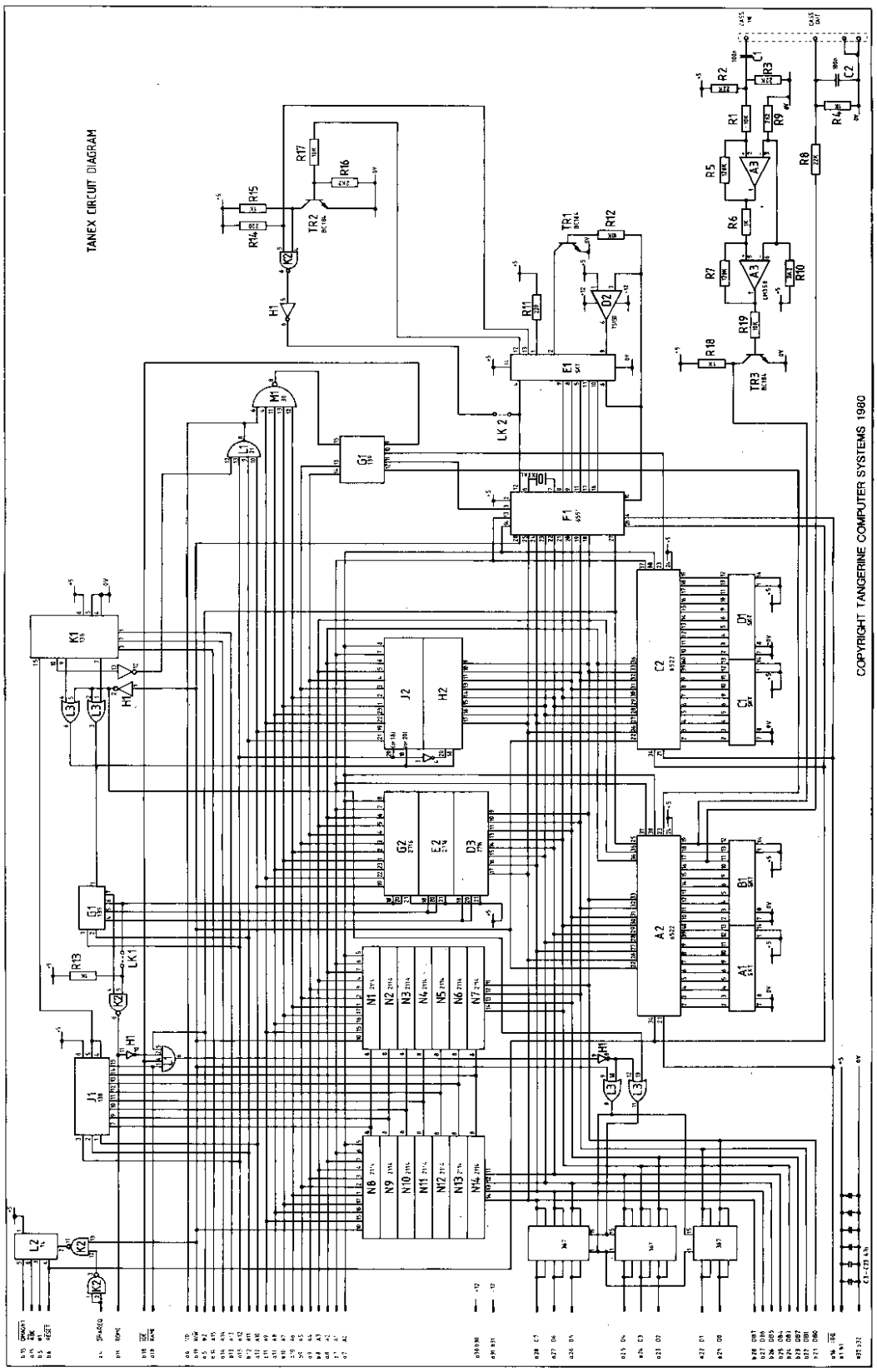
Overseas Dealers

Gloor Instruments Ltd,
Schwizerstrasse 2,
8610 Uster,
Switzerland.

J Muller,
Korblumeinweg,
Ebersheim,
West Germany.

Micromation Systems Ltd,
21 Elma Park Centre,
Edendale Road,
Edonvale 1610,
South Africa.

Mr Beni
Mayost
Rehov Hashoftim 29
Mercaz Rasco
Kiryat Motzkin
Israel.


TANEX CIRCUIT DIAGRAM
COPYRIGHT TANGERINE COMPUTER SYSTEMS 1980

PROGRAMS

M Stanton
Maidstone

BAUD RATES

Tanbug V2 contains the necessary software to drive a serial device. The serial port initialises to 110 baud whenever a reset is executed. If you require to run a device at other than 110 baud you need to be able to change the value found at address F862 in Tanbug. If you have an Eprom burner this is no great problem. Alternatively copy F850 to F87B into RAM (I find 1850 convenient) then change address 1862 to the appropriate value (see table below) using the M command. Initialise by typing G1800 and if your device is correctly connected then Tanbug will be printed on screen and on your serial device. Control V will then toggle the serial port on and off. Using this method I run an old 80 column VDU terminal at 4800 Baud.

```
91 HEX =    50 BAUD
92 HEX =    75 BAUD
93 HEX =   110 BAUD (THE PRESENT VALUE AT F862)
94 HEX =   135 BAUD
95 HEX =   150 BAUD
96 HEX =   300 BAUD
97 HEX =   600 BAUD
98 HEX =  1200 BAUD
99 HEX =  1800 BAUD
9A HEX =  2400 BAUD
9B HEX =  3600 BAUD
9C HEX =  4800 BAUD
9D HEX =  7200 BAUD
9E HEX =  9600 BAUD
9F HEX = 19200 BAUD
```

CRASHES

Since moving into a town I find my Tangerine crashes every time a heater is switched on next door, (anyone got a good mains filter!) sometimes warm start in Tanbug V2 wont get me back to basic to save the program that I had spent all night typing in. I found the best way is the forced warm start described in the Microtan companion. The only problem is the time it takes to find the three nulls which mark the end of the basic program. The longer the program the longer it takes. So I wrote the following routine. Once entered is saved on tape till it is required.

When the crash occurs press reset then load the routine. The routine prints the values that are required and enters basic. In answer to the prompt 'memory?' answer 39935 (or whatever) to protect program. Once in basic in immediate mode poke 1025 and 1026 with the dec equivalents of the HEX values from 0401 and 0402. Then poke 156 and 157 with the low and high bytes of the end off program address. Type list and there you are!

```
VALUES FOUND IN LOCATIONS $401
AND $402 ARE : 16 04
END OF PROGRAM IS : 07EC
MEMORY SIZE? 39935
TERMINAL WIDTH?

 38910 BYTES FREE

MICROTAN BASIC
(C) 1980 MICROSOFT

OK
POKE1025,22:POKE1026,4

OK
POKE156,236:POKE157,7

OK
LIST

 10 REM BASIC PROGRAM


A000 48      PHA
A001 98      TYA
A002 48      PHA
A003 8A      TXA
A004 48      PHA
A005 A90D    LDA #$000D
A007 200EF8  JSR $F80E
A00A A000    LDY #$0000
A00C 2077A0  JSR $A077
A00F AD0104  LDA $0401
A012 201AF8  JSR $F81A
A015 A920    LDA #$0020
A017 200EF8  JSR $F80E
A01A AD0204  LDA $0402

A01D 201AF8  JSR $F81A
A020 A031    LDY #$0031
A022 2077A0  JSR $A077
A025 A904    LDA #$0004
A027 8536    STA $0036
A029 A900    LDA #$0000
A02B 8535    STA $0035
A02D A000    LDY #$0000
A02F B135    LDA ($0035),Y
A031 C900    CMP #$0000
A033 F006    BEQ $A03B
A035 2070A0  JSR $A070
A038 4C2DA0  JMP $A02D
A03B C8      INY

A03C B135    LDA ($0035),Y
A03E C900    CMP #$0000
A040 F006    BEQ $A048
```

```
A042 2070A0 JSR $A070
A045 4C2DA0 JMP $A02D
A048 C8     INY
A049 B135   LDA ($0035),Y
A04B C900   CMP #$0000
A04D F006   BEQ $A055
A04F 2070A0 JSR $A070
A052 4C2DA0 JMP $A02D
A055 2070A0 JSR $A070
A058 2070A0 JSR $A070
A05B 2070A0 JSR $A070

A05E A536   LDA $0036
A060 201AF8 JSR $F81A
A063 A535   LDA $0035
A065 201AF8 JSR $F81A
A068 68     PLA
A069 AA     TAX
A06A 68     PLA
A06B A8     TAY
A06C 68     PLA
A06D 4CEDE2 JMP $E2ED
A070 E635   INC $0035
A072 D002   BNE $A076
A074 E636   INC $0036
A076 60     RTS

A077 B983A0 LDA $A083,Y
A07A F006   BEQ $A082
A07C 200EF8 JSR $F80E
A07F C8     INY
A080 D0F5   BNE $A077
A082 60     RTS

A083 0D 56 41 4C 55 45 53 20
A08B 46 4F 55 4E 44 20 49 4E
A093 20 4C 4F 43 41 54 49 4F
A09B 4E 53 20 24 34 30 31 20
A0A3 20 41 4E 44 20 24 34 30
A0AB 32 20 41 52 45 20 3B 20
A0B3 00 0D 45 4E 44 20 4F 46
A0BB 20 50 52 4F 47 52 41 4D
A0C3 20 49 53 20 3B 20 00 00
```

PATTERNS

```
10 REM HIGH RES PATTERNS
15 DIM P(8): POKE 34,0: POKE 35, 232: POKE 276,119 :?
        P= USR(P) : REM INITIALISE
20! 37, 255,0 : ! 39, 32, 0: REM SET SCREENS
25 REM DRAW SPIRAL
30 R = O : !12, 127, 127 : REM MIDDLE OF SCREEN
35 FOR I = O to 6.3 * 4 STEP 0.1
40 R = R + 0.2 : ! 7, 128 + R*2*SIN(I), 128 + 1.2*R*COS(I)
45 NEXT I
50 REM DRAW PATTERNS
55 Y = O : FOR I = 1 TO 8 : P(I) = INT (RNDCI)*255)
   NEXT : REM ARRAY FOR LINE MASK
60 for T = 1 TO 2
65 for X = 255 TO Ø   STEP - I
70! 12, X, X:16, Y*30, X : ! 6, X, Y*30
75 NEXT X
80 Y = Y + 1 : 1F Y*30 = 240 THEN 90
85 POKE (272), P(Y) : FOTO 65 : REM ALTER LINE MASK
90 Y = 0 : POKE (274), 1 : NEXT T : REM SWAP COORDINATES
   AND REDRAW
95 GOTO 55
100 END
```

The above program requires Microtanics high Res Eprom and the High Res card

The program runs continuously and could be renamed now you see it now you don't.

T.K. JONES 7.5.83

RADIO AMATEURS

Here is the latest list of call-signs:

G3ZAJ G8SUY G8EQZ G6DUI G3TDP
G4CCQ G8EGT G8UGR

Let us have your call signs and don't forget we want to know if you are using your computer for your hobby.

SIMPLE SIMON

So far we have printed two articles under this heading and so far we have had two comments (see Viewpoint) one "for" and one "against". If you have any views, drop me a line. . . Ed.

A Printer Facility for Microtanic's 2-Pass Assembler

The 2-pass assembler from Microtanic is a considerable advance over the translator in X-BUG, allowing as it does the us of labels, comments and arguments in decimal, hexadecimal and ASC11 notation. Obviously it doesn't have everything (in a 4K program it couldn't) and one of its annoying characteristics is the poor format of the listings it produces. If a comment uses all of the space allowed to it, it runs right up against the hex listing of the assembled code. Moreover, listings run on from page to page without a break and, worse still, if the paper isn't positioned exactly right, it prints on top of the perforations).

The two programs given here give a considerable improvement in this situation. Firstly, they insert spaces on every line between the comment, if any, and the hex listing. Secondly, they add 6 blank lines after every 60 lines printed (this can easily be changed for other page lengths). The more complex of the two programs also provides facilities for line- and page-feeds.

HOW THEY ARE USED

When the 2-Pass assembler is cold-started, it asks if a printer is to be used. The normal Tanbug routines can be used; all that is necessary is to type control-P or control-V to turn on the parallel or the serial printer interfaces, followed by Return. The 2-pass assembler then turns off the VDU function in Tanbug's output routine (it has its own specialised screen printing routines and doesn't need it) and uses Tanbug's routine to produce printed listings.

Alternatively, the user can supply his/her own printer routine, in which case, when the assembler asks 'PRINTER'?the entry address of the routine must be given (in hex). The routine will be called with the character to be printed in the accumulator.

In fact, since the programs given here use Tanbug's print routine, one must both give the entry address and hit ctrl-P or ctrl-V to turn on the appropriate printer output, i.e. since, for reasons of my own I have placed these routines at address A000, my response to 'Printer?' is A000-P (CR).

The 2-Pass assembler, as its name suggests, makes two passes through a program to assemble it. On the first pass it works out the length of each instruction, and therefore the address(es) it occupies. At the same time, it builds up a list of the labels it encounters, each with its address. On this pass it doesn't assemble any code, since it can't be sure of knowing all the addresses on time (e.g. an instruction such as JSR FINDIT could be completely assembled if the subroutine FINDIT had already been encountered, since its address would be known, but if FINDIT comes later, the address part of the JSR instruction could not be filled.)

On the second pass through the program all the addresses are known, and the final code is assembled. At the same time, if the print option has been activated, the assembler prints a listing.

The assembler also has a 'List Labels' function, which will produce a listing if the print option is active.

So printing is done by typing OP (CR) on the command line, which toggles the print option, followed by A2 (CR) to perform the 2nd pass of assembly, or LL (CR) to list labels.

HOW THEY WORK

These routines maintain two variables, CHARS, which counts the number of characters printed on the present line, and LINES, which counts the number of lines printed on the present page. These variables must be present to zero before the program is used, and this is done by the group of instructions at SETUP, which then jumps into the 2-pass assembler. I have put these variables in zero page locations 31 and 33 (hex). These are in Tanbug's breakpoint code store, but in fact Tanbug only uses the even numbered locations.

1. The simple one.

As mentioned above, the assembler calls the print routine with the character to be printed in the accumulator. The first thing the routine does, after pushing a copy of the character onto the stack for safe keeping, is to see if it is a carriage return (hex code OD). If not, it increases the variable CHARS, and then checks its new value. If CHARS is 33 (decimal), we are about to print the first character after the comment field, so we print two spaces to separate comments from the hex listing of the assembled code, and then print the character. If CHARS is not 33 we just print the character. (Note that since PRINT is a subroutine, the instruction JMP PRINT on the line after label DOIT is equivalent to JSR PRINT, RTS).

When a carriage return character is encountered, a different section of the program is entered at label CR. Here, the character count is reset to zero, since we are starting a new line, and the variable LINES is incremented and its value checked. This variable represents the number of lines which have already been sent to the printer, unlike CHARS, which gives the number of the character we are about to print. So if LINES says that 60 lines have been printed on this page, it is set to 6 and used to control a loop which prints 6 carriage return characters and exits with LINES equal to zero, after which the original carriage return is printed. These values can easily be changed for different page lengths or simply to modify the spacing, eg 64 lines printed and two blank.

2. The complex one.

The simple routine helps a lot, but it still has some drawbacks. To avoid having the label list following immediately on the heels of the program listing, it is necessary to feed the paper on by hand, and this puts it out of step with the count in LINES. The only way to get back into step is to move the paper on to the start of a new page and reset LINES to zero. However, I couldn't at first see any way to get around this short of modifying the 2-pass assembler itself, which I didn't particularly want to do. Finally, inspiration hit me. The print routine can examine the screen memory to see which command is active (The command will be either A2 or LL). Normally this will be the same every time the print routine is called, i.e. if we are printing a label list, the routine will be called several dozen, or even several hundred times, and every time, the LL command will be on the screen. However, if we later use the A2 command to print a listing, even though the print routine has no way of knowing if a fraction of a second has gone by since it was last called or if an hour has passed, it can tell that the command has changed from LL to A2. This is the key to the complex routine. A record is kept (in OLDONE) of the command that was in force last time a character was printed, and every time the routine is called, this is compared with the command on the screen (only the 2nd

letter of the command is checked, which is found in memory location 30A hex). If the command hasn't changed, the character is printed in the normal way, inserting spaces and extra lines as before. But if the command has changed, the user is offered a menu of options, first storing the new command in OLDONE, since we don't want to get offered the menu every time a character is printed.

The menu offers the following options.

1. 'Page & Print' (Respond 'Y')
   If the paper is not at the start of a new page, then a sufficient number of CR characters is sent to reach a new page, and then printing is resumed in the normal way.
2. 'No page, Print' (Respond 'N')
   This corresponds to the simple routine, it simply starts printing.
3. 'Page feed only' (Respond with CTRL-L)
   This sends CR characters until the start of the next page is reached, and then turns the printing off by setting variable SWITCH to zero. SWITCH is examined every time the printing section of the program is entered at label CONTIN, and if it is zero Tanbug's print routine is not called. (CTRL-L is ASC11 form-feed)
4. 'No printing' (Respond DELETE)
   This turns printing off by zeroing SWITCH. Its purpose is to reset OLDONE without doing any printing, e.g. if you have just printed a label list and for some reason you want to print it again, but on a new page, you cannot just repeat the command LL, since the command line would hold the same command as the last time the print routine was called, and it would think it was continuing the previous list label command. So you give the A2 command, and select the 'no printing' option. The assembler performs its second pass, harmlessly, and the print routine leaves '2' in OLDONE. Now give the List Labels command, and since the command has changed you get offered the menu.
5. 'Force feed' (Respond 'F')
   This is similar to the 'page feed only' option, except that page feed will not do anything if the paper is already at the top of a page. 'Force feed' always advances the paper.
6. 'Line feed' (Line feed key)
   This option sends one (CR) character to the printer, and then, unlike the other options, offers the menu again. So as many line feeds as required can be sent. This would typically be used to separate a label list from the end of a program listing.

   Any other responses are ignored and the routine continues to wait for a valid character.

TANDOS AND THE 2-PASS ASSEMBLER

As yet no version of the assembler is available for use with the floppy discs, but it is easy enough to make the two work together. Saving source files is no problem, since the assembler displays the start and end addresses, and loading a source file back into the same addresses is just a question of typing the name you gave to it. The problem is in letting the assembler know it is there. The best method I have yet discovered is the following:

After loading the source file into store, cold start the assembler, setting up the print routine if you are going to use it, and give the ES (Edit Source) command, followed by II (insert a new line at position 1), then Return. The assembler will immediately display the source end address and last line number. Now hit backspace to cancel the insert command and that's it!

The menu as listed on the Tangerines screen.

```
PAGE AND PRINT . . Y
NO PAGE, PRINT N
PAGE FEED ONLY . . CTRL-L
NO PRINTING . . . (DEL)
FORCE FEED . . . . F
LINE FEED . . . . . (LF)
```

```
0001 ;Printer facility for     1400
0002 ;2-Pass assembler.        1400
0003 ;Counts characters printed 1400
0004 ;& inserts spaces between  1400
0005 ;32nd & 33rd.              1400
0006 ;Counts CR's, resets char  1400
0007 ;count & increases line    1400
0008 ;count. Prints extra CR's  1400
0009 ;(to 66, page length) when 1400
0010 ; & preset number (60) is  1400
0011 ;reached.                  1400
0012 ;                          1400
0013 ;Counts (in $31,$33) must  1400
0014 ;be set to 0 before use.   1400
0015 ;                          1400
0016 ;                          1400
0017 ;                          1400
0018 ;                          1400
0019 LINES  EPZ $31             1400
0020 CHARS  EPZ $33             1400
0021 PRINT  EQU $FE75           1400
0022 ;                          1400
0023 ; ********************    1400
0024 ; **                **    1400
0025        ORG $A000           1400
0026 ; **                **    A000
0027 ; ********************    A000
0028 ;                          A000
0029        PHA                 A000 48
0030        CMP #$0D;<CR>       A001 C9 0D
0031        BEQ CR              A003 F0 15
0032 ISCHAR INC CHARS           A005 E6 33
0033        LDA CHARS           A007 A5 33
0034        CMP #$21;33rd char. A009 C9 21
0035        BNE DOIT            A00B D0 26
0036        LDA #$20;<SP>       A00D A9 20
0037        JSR PRINT           A00F 20 75 FE
0038        LDA #$20;<SP>       A012 A9 20
0039        JSR PRINT           A014 20 75 FE
0040        CLC                 A017 18
0041        BCC DOIT;Uncond.    A018 90 19
0042 ;                          A01A
0043 CR     LDA #0;Reset CHARS  A01A A9 00
0044        STA CHARS           A01C 85 33
0045 LF     INC LINES           A01E E6 31
0046        LDA LINES           A020 A5 31
0047        CMP #$3C;60 lines   A022 C9 3C
0048        BNE DOIT;Printed ?  A024 D0 0D
0049        LDA #$6;For 66=page A026 A9 06
0050        STA LINES           A028 85 31
0051 PRLOOP LDA #$0D;<CR>       A02A A9 0D
0052        JSR PRINT           A02C 20 75 FE
0053        DEC LINES           A02F C6 31
0054        BNE PRLOOP          A031 D0 F7
0055 DOIT   PLA ;Get char code  A033 68
0056        JMP PRINT;& return. A034 4C 75 FE
0057 ;                          A037
0058 ;                          A037
0059 ;                          A037
0060 ;                          A037
0061 SETUP  LDA #0              A037 A9 00
0062        STA CHARS           A039 85 33
0063        STA LINES           A03B 85 31
0064        JMP $C000;2-PASS    A03D 4C 00 C0
0065 ;                          A040
0066 ;                          A040
0067 ;                          A040
LINES = 0031 CHARS = 0033
PRINT = FE75 ISCHAR= A005
CR    = A01A LF    = A01E
PRLOOP= A02A DOIT  = A033
SETUP = A037
```

```
0001 ;Printer facility for       1400
0002 ;2-Pass assembler.          1400
0003 ;Counts characters Printed  1400
0004 ;& inserts sPaces between   1400
0005 ;32nd and 33rd.             1400
0006 ;Detects CR char's, resets  1400
0007 ;char count and increases   1400
0008 ;line count. Prints extra   1400
0009 ;CR's (to 66, Page length)  1400
0010 ;when a Preset number (60)  1400
0011 ;is reached.                1400
0012 ;                          1400
0013 ;Counter (in $31,$33) must  1400
0014 ;be set to 0 before use.    1400
0015 ;                          1400
0016 ;                          1400
0017 ;                          1400
0018 ;                          1400
0019 ICHAR  EPZ $1               1400
0020 LINES  EPZ $31              1400
0021 CHARS  EPZ $33              1400
0022 OLDONE EPZ $35              1400
0023 SWITCH EPZ $37              1400
0024 VDU    EQU $0200            1400
0025 INSTR  EQU $030A            1400
0026 POLLKB EQU $FDFA            1400
0027 PRINT  EQU $FE75            1400
0028 OUTPCR EQU $FE73            1400
0029 ;                           1400
0030 ; ******************        1400
0031 ; **              **        1400
0032        ORG $A000            1400
0033 ; **              **        A000
0034 ; ******************        A000
0035 ;                           A000
0036        PHA                  A000 48
0037        LDA INSTR            A001 AD 0A 03
0038        CMP OLDONE           A004 C5 35
0039        BEQ CONTIN           A006 F0 78
0040 ;                           A008
0041 ;If the instruction code    A008
0042 ;is not the same as last    A008
0043 ;time, offer menu.          A008
0044 ;                           A008
0045        STA OLDONE;It's new  A008 85 35
0046        TXA ;Save X          A00A 8A
0047        PHA                  A00B 48
0048 ;                           A00C
0049 ;First clear screen top.    A00C
0050 ;                           A00C
0051        LDA #$20;<SP>        A00C A9 20
0052        LDX #0               A00E A2 00
0053 CLS    STA VDU,X            A010 9D 00 02
0054        INX                  A013 E8
0055        BNE CLS              A014 D0 FA
0056 ;                           A016
0057 ;Now Print menu.            A016
0058 ;                           A016
0059        LDX #21              A016 A2 15
0060 MNLOOP LDA MESS1,X          A018 BD D3 A0
0061        STA VDU+36,X         A01B 9D 24 02
0062        LDA MESS2,X          A01E BD E9 A0
0063        STA VDU+68,X         A021 9D 44 02
0064        LDA MESS3,X          A024 BD FF A0
0065        STA VDU+100,X        A027 9D 64 02
0066        LDA MESS4,X          A02A BD 15 A1
0067        STA VDU+132,X        A02D 9D 84 02
0068        LDA MESS5,X          A030 BD 2B A1
0069        STA VDU+164,X        A033 9D A4 02
0070        LDA MESS6,X          A036 BD 41 A1
0071        STA VDU+196,X        A039 9D C4 02
0072        DEX                  A03C CA
0073        BPL MNLOOP           A03D 10 D9
0074 ;                           A03F
0075 ;Now get response.          A03F
0076 ;Note that POLLKB corrupts  A03F
0077 ;all the registers.         A03F
0078 ;                           A03F
0079 GETKEY TYA                  A03F 98
0080        PHA                  A040 48
0081        CLI ;Interrupts ON   A041 58
0082        JSR POLLKB           A042 20 FA FD
0083        SEI ;OFF again.      A045 78
0084        PLA                  A046 68
0085        TAY                  A047 A8
0086        LDA ICHAR            A048 A5 01
0087                             A04A
0088        CMP ##7F;Delete.     A04A C9 7F
0089        BEQ PRNOFF           A04C F0 2C
0090        CMP #'N              A04E C9 4E
0091        BEQ PRNON            A050 F0 24
0092        CMP #'Y              A052 C9 59
0093        BEQ DOPAGE           A054 F0 17
0094        CMP ##0C;CTRL-L      A056 C9 0C
0095        BEQ DOPAGE           A058 F0 13
0096        CMP ##0A;<LF>        A05A C9 0A
0097        BNE TRYF             A05C D0 06
0098        JSR LF;Tests end of  A05E 20 A9 A0
0099        CLC        ;Page.    A061 18
0100        BCC GETKEY;Uncond.   A062 90 DB
0101 TRYF   CMP #'F              A064 C9 46
0102        BNE GETKEY;Invalid.  A066 D0 D7
0103        JSR OUTPCR           A068 20 73 FE
0104        INC LINES            A06B E6 31
0105 DOPAGE JSR NEWPAG           A06D 20 C2 A0
0106        LDA ICHAR;Again.     A070 A5 01
0107        CMP #'Y              A072 C9 59
0108        BNE PRNOFF           A074 D0 04
0109 PRNON  LDA ##FF             A076 A9 FF
0110        BNE SET;Uncond.      A078 D0 02
0111 PRNOFF LDA #0               A07A A9 00
0112 SET    STA SWITCH           A07C 85 37
0113        PLA ;Recover X.      A07E 68
0114        TAX                  A07F AA
0115 ;                           A080
0116 ;Now re-enter the main      A080
0117 ;Program.                   A080
0118 ;                           A080
0119 CONTIN LDA SWITCH           A080 A5 37
0120        BNE PRNTIT           A082 D0 02
0121        PLA ;Clear stack.    A084 68
0122        RTS ;..and return.   A085 60
0123 ;                           A086
0124 PRNTIT PLA ;Recover char    A086 68
0125        PHA ;Save it again   A087 48
0126        CMP ##0D;<CR>        A088 C9 0D
0127        BEQ CR               A08A F0 18
0128 ISCHAR INC CHARS            A08C E6 33
0129        LDA CHARS            A08E A5 33
0130        CMP ##21;33rd char   A090 C9 21
0131        BNE DOIT             A092 D0 21
0132        LDA #$20;<SP>        A094 A9 20
0133        JSR PRINT            A096 20 75 FE
0134        LDA ##20;<SP>        A099 A9 20
0135        JSR PRINT            A09B 20 75 FE
0136        LDA ##23;New count   A09E A9 23
0137        STA CHARS            A0A0 85 33
0138        BNE DOIT;Uncond.     A0A2 D0 11
0139 ;                           A0A4
0140 CR     LDA #0;Reset CHARS   A0A4 A9 00
0141        STA CHARS            A0A6 85 33
0142        PLA                  A0A8 68
0143 LF     PHA                  A0A9 48
0144        INC LINES            A0AA E6 31
0145        LDA LINES            A0AC A5 31
0146        CMP ##3C;60 lines    A0AE C9 3C
0147        BNE DOIT;Printed.    A0B0 D0 03
0148        JSR NEWPAG           A0B2 20 C2 A0
0149 DOIT   PLA ;Get char code   A0B5 68
0150        JMP PRINT            A0B6 4C 75 FE
0151 ;                           A0B9
0152 ;                           A0B9
0153 SETUP  LDA #0               A0B9 A9 00
0154        STA CHARS            A0BB 85 33
0155        STA LINES            A0BD 85 31
0156        JMP #C000;To 2-PASS  A0BF 4C 00 C0
0157 ;                           A0C2
0158 ;Page feed routine.         A0C2
0159 ;(X register saved before   A0C2
0160 ;call).                     A0C2
0161 ;                           A0C2
0162 NEWPAG LDX LINES            A0C2 A6 31
0163        BEQ FINISH           A0C4 F0 0C
0164 FEED   JSR OUTPCR           A0C6 20 73 FE
0165        INX                  A0C9 E8
0166        CPX #66;Page length  A0CA E0 42
0167        BNE FEED             A0CC D0 F8
0168        LDA #0               A0CE A9 00
0169        STA LINES            A0D0 85 31
0170 FINISH RTS                  A0D2 60
0171 ;                           A0D3
0172 ;Message data follows.      A0D3
0173 ;                           A0D3
0174 MESS1  BYT 'P,'A,'G,'E      A0D3 50 41 47 45
0175        BYT ' ,'A,'N,'D      A0D7 20 41 4E 44
0176        BYT ' ,'P,'R,'I      A0DB 20 50 52 49
0177        BYT 'N,'T,',,'.      A0DF 4E 54 2E 2E
0178        BYT 'Y,32,32,32      A0E3 59 20 20 20
0179        BYT 32,32            A0E7 20 20
0180 MESS2  BYT 'N,'O,' ,'P      A0E9 4E 4F 20 50
0181        BYT 'A,'G,'E,',      A0ED 41 47 45 2C
0182        BYT ' ,'P,'R,'I      A0F1 20 50 52 49
0183        BYT 'N,'T,',,'.      A0F5 4E 54 2E 2E
0184        BYT 'N,32,32,32      A0F9 4E 20 20 20
0185        BYT 32,32            A0FD 20 20
0186 MESS3  BYT 'P,'A,'G,'E      A0FF 50 41 47 45
0187        BYT ' ,'F,'E,'E      A103 20 46 45 45
```

```
0188          BYT 'D,' ,'O,'N     A107 44 20 4F 4E
0189          BYT 'L,'V,'.,'.     A10B 4C 56 2E 2E
0190          BYT 'C,'T,'R,'L     A10F 43 54 52 4C
0191          BYT '+,'L           A113 2B 4C
0192 MES94    BYT 'N,'O,' ,'P     A115 4E 4F 20 50
0193          BYT 'R,'I,'N,'T     A119 52 49 4E 54
0194          BYT 'I,'N,'G,'.     A11D 49 4E 47 2E
0195          BYT '.,'.,'.,'.     A121 2E 2E 2E 2E
0196          BYT 'K,'D,'E,'L     A125 3C 44 45 4C
0197          BYT 'X,32           A129 3E 20
0198 MES95    BYT 'F,'O,'R,'D     A12B 46 4F 52 43
0199          BYT 'E,' ,'F,'E     A12F 45 20 46 45
0200          BYT 'E,'D,'.,'.     A133 45 44 2E 2E
0201          BYT '.,'.,'.,'.     A137 2E 2E 2E 2E
0202          BYT 'F,32,32,32     A13B 46 20 20 20
0203          BYT 32,32           A13F 20 20
0204 MES96    BYT 'L,'I,'N,'E     A141 4C 49 4E 45
0205          BYT ' ,'F,'E,'E     A145 20 46 45 45
0206          BYT 'D,'.,'.,'.     A149 44 2E 2E 2E
0207          BYT '.,'.,'.,'.     A14D 2E 2E 2E 2E
0208          BYT '<,'L,'F,'>     A151 3C 4C 46 3E
0209          BYT 32,32           A155 20 20
```

```
ICHAR = 0001 LINES = 0031
CHARS = 0033 OLDONE= 0035
SWITCH= 0037 VDU   = 0200
INSTR = 030A POLLKB= FDFA
PRINT = FE75 OUTPCR= FE73
CLS   = A010 MNLOOP= A018
GETKEY= A03F TRYF  = A064
DOPAGE= A069 PRNON = A076
PRNOFF= A07B SET   = A07C
CONTIN= A080 PRNTIT= A086
ISCHAR= A08C CR    = A0A4
LF    = A0A9 DOIT  = A0B5
SETUP = A0B9 NEWPAG= A0C2
FEED  = A0C6 FINISH= A0D2
MESS1 = A0D3 MESS2 = A0E9
MESS3 = A0FF MESS4 = A115
MESS5 = A128 MESS6 = A141
```

TELEPRINTER ROUTINE

Having reached the stage where I could not bear to carry on trying programs out without some form of printer to record program developments, I investigated the possibilities of using an ex-G.P.O. Teleprinter 7E. An article in 'Computing Today' December 1982 provided the interface circuitry and printer details, while a colleague provided the teleprinter. All I had to do was to provide a program that could be used with my Microtan.

For those of you who would like a cheap but noisy method of providing a print out, here is the program which is used. No attempt was made to shorten it once it was working. It copies the top fourteen lines of Microtan's screen and has been found useful for printing disassembled listings. In use, Command 1 (addr.) RETURN, will provide fourteen lines on the screen, ESC breaks from XBUG, leaving the fourteen lines starting from line one on the screen. G1E60 will then provide a print out, while G1E63 can be used so that a blank line is not left between each screen print. BASIC listings can be printed out from the screen via the P–USR(P) function. (POKE 34, 96: POKE 35, 30). The interface circuit is driven from the first 6522, Port B Bit 0 on Tanex i.e. socket B1 Pin 2.

As the ASC11 Code Set is not used by teleprinters, a code conversion routine takes place using a look up table and also converts lower case characters to upper case. The code format required is a start bit followed by five data bits followed by two stop bits, eight bits in all. If a code match cannot be found a space is printed allowing signs like < or > to be added afterwards. The OUTCH and PRINT parts of this program have been used in two further programs, one to print out a complete BASIC listing (without using the screen first) and uses many routines from the BASIC ROMS, and one to print out from the 2 PASS ASSEMBLER.

J J Buss
Rainham
Kent

```
0001 : TELEPRINTER ROUTINE      1E60
0002 :                          1E60
0003 :   J.BUSS  7-3-1983       1E60
0004 :                          1E60
0005 : COPIES MICROTAN'S SCREEN1E60
0006 : TO PRINTER.              1E60
0007 :                          1E60
0008 : £ = HEX.  + = IMMED.MODE1E60
0009 :                          1E60
0010 DATA   EPZ £80             1E60
0011 DATA1  EPZ £81             1E60
0012 LAST   EPZ £82             1E60
0013 SHIFT  EPZ £83             1E60
0014 COUNT  EPZ £84             1E60
0015 :                          1E60
0016 :START WITH CARRIAGE       1E60
0017 :      RETURN LINE FEED    1E60
0018 :                          1E60
0019 START  JSR CRLF            1E60 20 F7 1E
0020 :                          1E63
0021 :IDENTIFY END OF LINE 14.  1E63
0022 :PRINTER IN LETTER MODE.   1E63
0023 :IDENDIFY START OF LINE.   1E63
0024 :                          1E63
0025        LDA +£C0            1E63 A9 C0
0026        STA LAST            1E65 85 82
0027        LDA +£1             1E67 A9 01
0028        STA SHIFT           1E69 85 83
0029        LDA +£1F            1E6B A9 1F
0030        STA DATA            1E6D 85 80
0031        JSR PRINT           1E6F 20 10 1F
0032        LDX +£0             1E72 A2 00
0033        STX COUNT           1E74 86 84
0034 :                          1E76
0035 :PICK UP ASC11 CODE FOR    1E76
0036 :EACH SCREEN LOCATION.     1E76
0037 :                          1E76
0038 PAGE2  LDA £200,X          1E76 BD 00 02
0039        STA DATA1           1E79 85 81
0040        JSR OUTCH           1E7B 20 8F 1E
0041        INX                 1E7E E8
0042        BNE PAGE2           1E7F D0 F5
0043 PAGE3  LDA £300,X          1E81 BD 00 03
0044        STA DATA1           1E84 85 81
0045        JSR OUTCH           1E86 20 8F 1E
0046        INX                 1E89 E8
0047        CPX LAST            1E8A E4 82
0048        BNE PAGE3           1E8C D0 F3
0049        BRK                 1E8E 00
0050 :USE RTS FOR BASIC         1E8F
0051 :                          1E8F
0052 :SAVE X AND A REGISTERS.   1E8F
0053 :SEARCH LETTERS TABLE FOR  1E8F
0054 :ASC11 MATCH,BRANCH TO     1E8F
0055 :FIGURE TABLE IF NOT FOUND1E8F
0056 :CHECK PRINTER IN LETTER   1E8F
0057 :MODE,CHANGE IF NOT.       1E8F
0058 OUTCH  PHA                 1E8F 48
0059        TXA                 1E90 8A
0060        PHA                 1E91 48
0061        LDX +£0             1E92 A2 00
0062 SEARCH INX                 1E94 E8
0063        INX                 1E95 E8
0064        LDA LTRS-2,X        1E96 BD 3D 1F
0065        CMP +£FF            1E99 C9 FF
0066        BEQ FIGURE          1E9B F0 1E
0067        CMP DATA1           1E9D C5 81
0068        BNE SEARCH          1E9F D0 F3
0069        INX                 1EA1 E8
0070        LDA LTRS-2,X        1EA2 BD 3D 1F
0071        LDY SHIFT           1EA5 A4 83
0072        CPY +£1             1EA7 C0 01
0073        BEQ PRT             1EA9 F0 07
0074        LDY +£1F            1EAB A0 1F
0075        STY DATA            1EAD 84 80
0076        JSR PRINT           1EAF 20 10 1F
0077 :                          1EB2
0078 :STORE PRINT CODE.         1EB2
0079 :RECORD PRINTER IN LETTER  1EB2
0080 :                MODE.     1EB2
0081 :                          1EB2
0082 PRT    STA DATA            1EB2 85 80
0083        LDY +£1             1EB4 A0 01
0084        STY SHIFT           1EB6 84 83
0085        JMP END             1EB8 4C E8 1E
0086 :                          1EBB
0087 :SEARCH FIGURE TABLE FOR   1EBB
0088 :ASC11 MATCH.              1EBB
0089 :CHECK PRINTER IN FIGURE   1EBB
0090 :MODE,CHANGE IF NOT.       1EBB
```

```
0091 :                                 1EBB
0092 FIGURE LDX +£0                    1EBB A2 00
0093 LOOP   INX                        1EBD E8
0094        INX                        1EBE E8
0095        LDA FIGS-2,X               1EBF BD B3 1F
0096        CMP +£FF                   1EC2 C9 FF
0097        BEQ SPACE                  1EC4 F0 1E
0098        CMP DATA1                  1EC6 C5 81
0099        BNE LOOP                   1EC8 D0 F3
0100        INX                        1ECA E8
0101        LDA FIGS-2,X               1ECB BD B3 1F
0102        LDY SHIFT                  1ECE A4 83
0103        CPY +£0                    1ED0 C0 00
0104        BEQ BR1                    1ED2 F0 07
0105        LDY +£1B                   1ED4 A0 1B
0106        STY DATA                   1ED6 84 80
0107        JSR PRINT                  1ED8 20 10 1F
0108 BR1    STA DATA                   1EDB 85 80
0109        LDY +£0                    1EDD A0 00
0110        STY SHIFT                  1EDF 84 83
0111        JMP END                    1EE1 4C E8 1E
0112 SPACE  LDY +£04                   1EE4 A0 04
0113        STY DATA                   1EE6 84 80
0114 :                                 1EE8
0115 :PRINT CHARACTER VIA PRINT        1EE8
0116 :SUBROUTINE.                      1EE8
0117 :CHECK FOR END OF LINE.           1EE8
0118 :                                 1EE8
0119 END    JSR PRINT                  1EE8 20 10 1F
0120        INC COUNT                  1EEB E6 84
0121        LDY +£20                   1EED A0 20
0122        CPY COUNT                  1EEF C4 84
0123        BEQ CRLF2                  1EF1 F0 17
0124 :                                 1EF3
0125 :RESTORE X AND A REGS.            1EF3
0126 :                                 1EF3
0127 END2   PLA                        1EF3 68
0128        TAX                        1EF4 AA
0129        PLA                        1EF5 68
0130        RTS                        1EF6 60
0131 :                                 1EF7
0132 :CARRIAGE RETURN LINE FEED        1EF7
0133 :                                 1EF7
0134 CRLF   LDY +£08                   1EF7 A0 08
0135        STY DATA                   1EF9 84 80
0136        JSR PRINT                  1EFB 20 10 1F
0137        LDY +£02                   1EFE A0 02
0138        STY DATA                   1F00 84 80
0139        JSR PRINT                  1F02 20 10 1F
0140        LDY +£0                    1F05 A0 00
0141        STY COUNT                  1F07 84 84
0142        RTS                        1F09 60
0143 CRLF2  JSR CRLF                   1F0A 20 F7 1E
0144        JMP END2                   1F0D 4C F3 1E
0145 :                                 1F10
0146 :SET UP VIA .                     1F10
0147 :INVERT DATA BITS.                1F10
0148 :TRANSMIT 1 BIT AT A TIME.        1F10
0149 :                                 1F10
0150 PRINT  PHA                        1F10 48
0151        LDA +£1                    1F11 A9 01
0152        STA £BFC0                  1F13 8D C0 BF
0153        LDA +£FF                   1F16 A9 FF
0154        STA £BFC2                  1F18 8D C2 BF
0155        LDA DATA                   1F1B A5 80
0156        EOR +£1F                   1F1D 49 1F
0157        LDX +£7                    1F1F A2 07
0158 TXBIT  JSR DELAY                  1F21 20 2E 1F
0159        CLC                        1F24 18
0160        ROR @                      1F25 6A
0161        ROL £BFC0                  1F26 2E C0 BF
0162        DEX                        1F29 CA
0163        BNE TXBIT                  1F2A D0 F5
0164        PLA                        1F2C 68
0165        RTS                        1F2D 60
0166 :                                 1F2E
0167 :DELAY ROUTINE I.E. WAIT          1F2E
0168 :FOR PRINTER.                     1F2E
0169 :                                 1F2E
0170 DELAY  PHA                        1F2E 48
0171        TXA                        1F2F 8A
0172        PHA                        1F30 48
0173        LDY +£11                   1F31 A0 11
0174 DEL1   LDX +£BD                   1F33 A2 BD
0175 DEL2   DEX                        1F35 CA
0176        BNE DEL2                   1F36 D0 FD
0177        DEY                        1F38 88
0178        BNE DEL1                   1F39 D0 F8
0179        PLA                        1F3B 68
0180        TAX                        1F3C AA
0181        PLA                        1F3D 68
0182        RTS                        1F3E 60
0183 :                                 1F3F
0184 :LETTERS TABLE.                   1F3F
0185 :                                 1F3F
0186 LTRS   BYT £41,£03,£42            1F3F 41 03 42 19 43 0E 44 09
0187        BYT £19,£43,£0E            1F47 45 01 46 0D 47 1A 48 14
0188        BYT £44,£09,£45            1F4F 49 06 4A 0B 4B 0F 4C 12
0189        BYT £01,£46,£0D            1F57 4D 1C 4E 0C 4F 18 50 16
0190        BYT £47,£1A,£48            1F5F 51 17 52 0A 53 05 54 10
0191        BYT £14,£49,£06            1F67 55 07 56 1E 57 13 58 1D
0192        BYT £4A,£0B,£4B            1F6F 59 15 5A 11 20 04 5B 0F
0193        BYT £0F,£4C,£12            1F77 5C 1D 5D 12 5E 04 5F 04
0194        BYT £4D,£1C,£4E            1F7F 61 03 62 19 63 0E 64 09
0195        BYT £0C,£4F,£18            1F87 65 01 66 0D 67 1A 68 14
0196        BYT £50,£16,£51            1F8F 69 06 6A 0B 6B 0F 6C 12
0197        BYT £17,£52,£0A            1F97 6D 1C 6E 0C 6F 18 70 16
0198        BYT £53,£05,£54            1F9F 71 17 72 0A 73 05 74 10
0199        BYT £10,£55,£07            1FA7 75 07 76 1E 77 13 78 1D
0200        BYT £56,£1E,£57            1FAF 79 15 7A 11 FF FF 31 17

0201        BYT £13,£58,£1D            1FB7 32 13 33 01 34 0A 35 10
0202        BYT £59,£15,£5A            1FBF 36 15 37 07 38 06 39 18
0203        BYT £11,£20,£04            1FC7 30 16 3F 19 2D 03 3A 0E
0204        BYT £5B,£0F,£5C            1FCF 25 0D 40 1A 23 11 28 0F
0205        BYT £1D,£5D,£12            1FD7 29 12 2E 1C 2C 0C 22 05
0206        BYT £5E,£04,£5F            1FDF 3D 1E 2F 1D 24 14 27 05
0207        BYT £04,£61,£03            1FE7 2B 11 3B 0E 0D 08 2A 1A
0208        BYT £62,£19,£63            1FEF 0A 02 5E 04 5F 04 60 05
0209        BYT £0E,£64,£09            1FF7 3C 04 3E 04 26 1A FF FF
0210        BYT £65,£01,£66            1FF7 65 01 66
0211        BYT £0D,£67,£1A            1F8A 0D 67 1A
0212        BYT £68,£14,£69            1F8D 68 14 69
0213        BYT £06,£6A,£0B            1F90 06 6A 0B
0214        BYT £6B,£0F,£6C            1F93 6B 0F 6C
0215        BYT £12,£6D,£1C            1F96 12 6D 1C
0216        BYT £6E,£0C,£6F            1F99 6E 0C 6F
0217        BYT £18,£70,£16            1F9C 18 70 16
0218        BYT £71,£17,£72            1F9F 71 17 72
0219        BYT £0A,£73,£05            1FA2 0A 73 05
0220        BYT £74,£10,£75            1FA5 74 10 75
0221        BYT £07,£76,£1E            1FA8 07 76 1E
0222        BYT £77,£13,£78            1FAB 77 13 78
0223        BYT £1D,£79,£15            1FAE 1D 79 15
0224        BYT £7A,£11                1FB1 7A 11
0225        BYT £FF,£FF                1FB3 FF FF
0226 :                                 1FB5
0227 :FIGURES TABLE.                   1FB5
0228 :                                 1FB5
0229 FIGS   BYT £31,£17,£32            1FB5 31 17 32
0230        BYT £13,£33,£01            1FB8 13 33 01
0231        BYT £34,£0A,£35            1FBB 34 0A 35
0232        BYT £10,£36,£15            1FBE 10 36 15
0233        BYT £37,£07,£38            1FC1 37 07 38
0234        BYT £06,£39,£18            1FC4 06 39 18
0235        BYT £30,£16,£3F            1FC7 30 16 3F
0236        BYT £19,£2D,£03            1FCA 19 2D 03
0237        BYT £3A,£0E,£25            1FCD 3A 0E 25
0238        BYT £0D,£40,£1A            1FD0 0D 40 1A
0239        BYT £23,£11,£28            1FD3 23 11 28
0240        BYT £0F,£29,£12            1FD6 0F 29 12
0241        BYT £2E,£1C,£2C            1FD9 2E 1C 2C
0242        BYT £0C,£22,£05            1FDC 0C 22 05
0243        BYT £3D,£1E,£2F            1FDF 3D 1E 2F
0244        BYT £1D,£24,£14            1FE2 1D 24 14
0245        BYT £27,£05,£2B            1FE5 27 05 2B
0246        BYT £11,£3B,£0E            1FE8 11 3B 0E
0247        BYT £0D,£08,£2A            1FEB 0D 08 2A
0248        BYT £1A,£0A,£02            1FEE 1A 0A 02
0249        BYT £5E,£04,£5F            1FF1 5E 04 5F
0250        BYT £04,£60,£05            1FF4 04 60 05
0251        BYT £3C,£04,£3E            1FF7 3C 04 3E
0252        BYT £04,£26,£1A            1FFA 04 26 1A
0253        BYT £FF,£FF                1FFD FF FF
```

AUTOREPEATTTTTT!

Autorepeat for the Tangerine ASCII Keyboard by A.C Margas, Cambridge

Intro

The IC used in a quad two-input Schmitt NAND package. It is most important that a Schmitt input gate is used in this application, and for this reason there is no alternative LS TTL device that can be used. I suppose I must have chosen it because I had one spare. CMOS could conceivably be used, but the CR values would have to be adjusted around N1. TTL is preferable, just to keep in line with the rest of the electronics around the circuit. The circuit has been in use for almost 2 years without problems.

Construction

1. Cut track leading to pin 15 of SK1. This disconnects the STROBE output from the socket. Two thin wires should be connected to either side of the cut track (the strongest mechanical points are at the socket pin and IC5 pin 11). The wire from the socket is the STROBE OUTPUT connection, while the other is the STROBE INPUT.

2. Construct the circuit shown (the layout shown is easy for mounting components on 0.1" veroboard), check for correct assembly and connect up. Switch on.

3. Press down a key on the keyboard. The character should appear once, and then after a period of about 1-1½ seconds, it should repeat at a rate determined by the setting of VR1. If all is well, set VR1 to obtain the desired repeat rate. Congratulate yourself on a job well done.

4. If you have a faulty circuit, first check your wiring. You should be then able to fault find the **circuit** using the information given under 'Circuit Operation' and some equipment (meter, LED with resistor, 'scope'etc).

5. You may like to re-wire the REPEAT key in series with the RESET key, thereby preventing accidental resetting of the processor as both keys will have to be depressed at the same time to issue a RESET.

Circuit Operation

1. Gate N1 forms an oscillator with a range of approx 80 to 1 Hz.

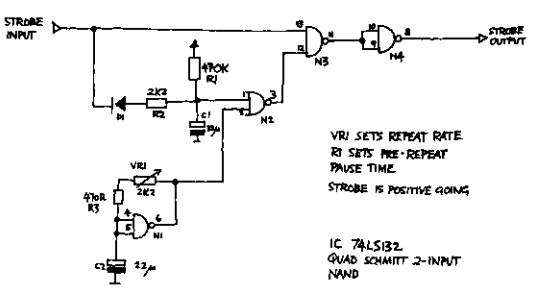
2. With no key depressed, STROBE INPUT is LO, hence pin 1 of N2 is normally HI. Pin 11 of N3 is normally HI. Pin 8 of N4 is always the inverse of pin 11 N3.

3. Everytime a key is depressed, STROBE INPUT goes HI. When a positive going pulse appears on STROBE INPUT, it is transmitted and inverted by both gates N3 and N4. As it is inverted twice, it appears in it correct polarity at STROBE OUTPUT.

4. STROBE INPUT stays HI throughout the duration of a key being depressed. As D1 is reverse biased, C1 will start charging through R1. When the voltage on C1 reaches the Schmitt level of N2 (about 1.8v) pin 3 of N2 will start toggling in sympathy of oscillator N1. This toggling signal will gate the HI STROBE INPUT signal via N3 and the keyboard will effectively be autorepeating. When the key is released, STROBE INPUT goes LO again, C1 is discharged via D1 and the circuit is reset to its condition of (2).

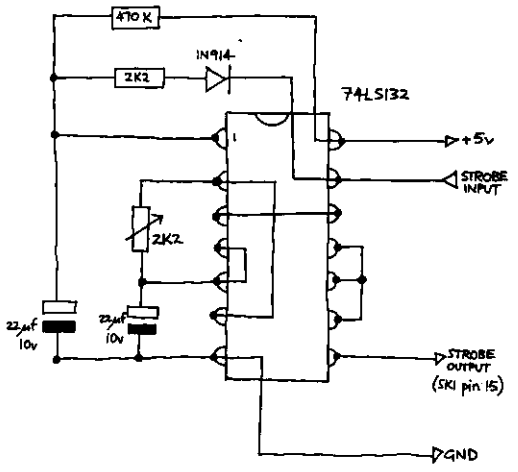
CCT. DIAGRAM

TANGEREINE ASC II KB AUTO REPEAT



LAYOUT DIAGRAM

TANGERINE ASCII KB AUTOREPEAT



Microtan Keyboard Auto-repeat : 2 M Marten B.Eng

The Microtan keyboard has a repeat key but in many ways this is inconvenient, especially when editing in Basic. Here is one way to give auto-repeat on MKII Keyboards.

The keyboard strobe output circuit is shown in fig. 1. A separate repeat oscillator formed by two gates in the 74LS00 is started when the repeat key is pressed. Transistor TR3 acts as a buffer to the mechanical switch contacts. The repeat oscillator clocks a flip flop, the output of which modulates the 'key pressed' output from the keyboard encoder clip AY-3-4592.

The strobe line now has multiple rising edges causing interrupts and interrogation of the keyboard part. All the essential elements are there to implement auto-repeat. If we pull the base of TR3 low to simulate the repeat key being operated after an initial delay of 0.5 - 15 we are home and dry!

For the delay circuit see fig. 2. All the components are easy to obtain and are probably already in the junk box. How does it work?

First we buffer the existing 'key pressed' signal using a spare gate in IC5 (74LS08). This gate then drives a transistor switch via a simple RC delay consisting of the 33 uf capacitor and part of the present potentiometer. The diode and 1K resistor give a faster discharge path so that the delay works again shortly after releasing a key. The actual component values are not at all critical, but the transistor should have a gain of 200 to ensure it saturates. High gain options such as a BC184L, BC109C etc are ideal.

Since the buffer gate used has 2 inputs, why not use it to advantage? When using the ASC11 keyboard the Microtan output part on the keyboard socket is not being used. Here we have used one output bit to enable the auto repeat allowing software control. A normal startup or reset will leave the auto-repeat enabled. It can be disabled by a write operation to $ BFF2

i.e. LDA # 00

STA $/ BFF2 in machine code

or using the Tanbug M command: in BFF2 (CR) 0 (CR) or in Basic Poke 49138,0

Loading a 1 into the same location will switch it back to auto-repeat mode.

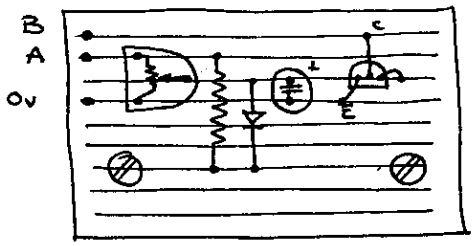
Method

The spare gate on ICs is pins 4, 5 and 6. None of these pins are connected to anything so life has been made easy, no tracks to cut. Obtain some fine hook up wire. The ideal type for this kind of job is the wire wrap wire with 'Kymar' insulation.

Wire a link from Pin 13 on SK1, the ribbon connector socket to pin 4 on IC5

Connect Pin 5 to Pin 13 on IC5
Connect Pin 6 on IC5 to the add on board point A
Connect the TR3 end of R10 to the add on point B
Connect the 0 volt line on the add on board to Pin 7 IC5

Construction of the add on unit is very much one of personal choice. Here is a suggested lay out using a strip board. There is plenty of room for it behind the numeric key pad on the reverse side of the board.



Be sure you know the pin connections of the transistor you use, there are so many variations it is easy to get it wrong. That shown is for a BC184L and the view is from the component side

Check the wiring carefully and when all is correct switch on. Reset the system and then press any key. You should get at least one character on the screen and then after a delay many more. Adjust the potentiometer to about 1/3 rotation away from point A or as required to give the desired delay. If too long a delay is set the repeat oscillator switch on rate is slow and its frequency may be high or it may fail to switch on at all.

The manual repeat key still works normally. It is possible that some versions of Tanbug do not leave the keyboard line switched on so try the software switch as described. This is necessary when using Mousepacket designs VDU card with TANBUG V3B installed.

For such a simple modification to the keyboard it gives a worthwhile benefit unless you NEVER have to edit your programmes!

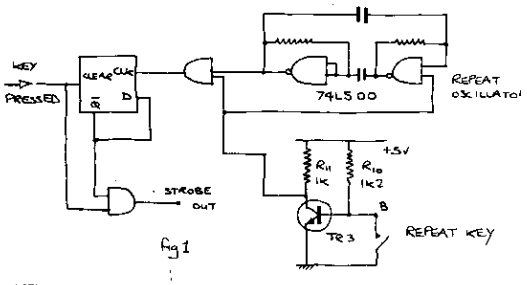


Fig 1

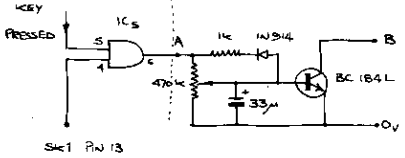


Fig 2

Microtan Keyboard Auto-repeat 2

TANSOFT 2 PASS ASSEMBLER

A Two Pass Assembler is now available for Tangerine Users. No more messy patches, just amend the Source Code and re-assemble to whatever location you desire. This Assembler is available in a 2732 Eprom and simply plugs into J2 on the Tanex Board. If you are a Basic User, there is also an Extension Eprom Board available which allows Software or Manual Switching between banks of Eprom space located between C000 and DFFF(hex). So 8k of Basic goes in one bank and the Assembler goes into the second bank leaving 8k Free for either our Forth Language or our Asimov Word Processor or 8k of your own Firmware.

What is a 2 Pass Assembler?

For those unfamiliar with 2 Pass Assemblers, they greatly simplify the task of writing Machine Code Programs. The main differences between a 2 Pass Assembler and the Line by Line Assembler(as in XBUG) are:

a) A copy of the Source Code is always retained, that is, a copy of what YOU actually type in (eg. LDA).
b) Any Instruction or Data Location can be given a Label (a name).
c) These Labels may be referenced by Instructions, so, for example, rather than JSR to a specific location you may JSR FRED, where Fred is perhaps the beginning of a Subroutine.

The great advantage of this system is that Instructions can be inserted or deleted from the Source Code, Lables swopped around, Data Areas altered in length etc, without worrying about the effect on addresses.
This system implies that you can assemble object code anywhere in memory.

Facilities:

The facilities offered by this particular Assembler are:
* Full source code editing including validation of instructions. *
* Cassette Handling Routines for maintenance of Source Code on Tape. *
* All standard 6502 Instructions are supported in the same format as used by the XBUG Line Assembler(except for ASL A which is replaced by ASL @ to avoid confusion with Labels). *
* Extra Instructions are supported - ORG(relative or absolute) EPZ(equate page zero), BYT(for single byte constants) and WOR(for 2 byte addresses). *
* Hexadecimal, character and decimal constants are supported. *
* Reference to Labels may include the format Label+constant(eg. FRED+2). *
* Various Print, List and Display options, including the facility to list the Labels and their Addresses. *
* Options to Dump or Assemble parts of the Source Code and to Load or Assemble several parts into one program. *
* Option to Assemble at an Eprom Location but to Store the Object in another RAM Address ready for Eprom Burning. *

*** AND ALL THIS IN 4K !!! ***

SUPPLIED IN EPROM..... 34.95

SPECIAL OFFER

H2 EXTENSION EPROM BOARD PLUS 2 PASS ASSEMBLER EPROM 49.45

MICROTANIC COMPUTER SYSTEMS 16 UPLAND ROAD DULWICH LONDON SE22 TEL: 01-693 1137

SUBSCRIPTION

Microtanic Computer Systems Ltd. 16 Upland Road, Dulwich
London. SE 22. 01 693 1137.

Please send me the next six copies of Microtan World.

Name.....

Address.....

.....

I enclose £10:00 in full payment.

Cheques should be made payable to Microtanic Computer Systems Ltd.

NEXT ISSUE

A lot of the content of the next issue will depend on you the readers.

We are particularly interested in articles for single board owners, useful routines, hints and tips you may have found, etc.etc.

Don't forget to let us have your letters telling us what you think about this issue and of course any comments on the system or any matter of general interest.

If you know anyone with a Microtan Computer who is not yet subscribing to Microtan World why not let him have the subscription form and see if you can persuade him to join us. The more subscribers we can get the better we can make the magazine.

Printed by Christian Graphic Arts Ltd.29 Freemantle Road,Gosport, Hants PO1 4RD