Cortex User Group Newsletter (May 1987)

Issue Number 10

Contents

1. Index
2. Editorial note
3. Letters
6. Variable listing program
8. More on Disk Inspect
12. Basic programs in EPROM
13. Extra system ROM
14. Better data separator
15. Short Tips
16. Advert's
Welcome to the first Cortex newsletter produced by Tim Gray and Ted Serwa. We apologise for its late arrival but this is due mainly to the late handover from Kevin Holloway following newsletter 9, which really should have been published before January. Since announcing our intention to take over the group during February subscriptions have been trickling in but there are still a lot to come. We have enough information for two newsletters at the moment and so will be producing Them almost at the same time. The next one will be produced as soon as enough material is at hand and from then on hopefully every two months.

The meeting of Cortex users held jointly with TI994A users was a great success in spite of the late notice. Tim Gray and Ted Serwa both had extensive Cortex systems on display showing hardware for external video interface, A.M.X. mouse interface, extended RAM + ROM, a high definition graphics card and lots of CRU based input-output systems. Software on display included Tims Drawtech drawing package plus Teds P.C.B design C.A.D. package and a multi-pass assembler editor. Also at the show with their Cortex machines were John Makenzie demonstrating his Wortex and Spelltex package and Alan Badcock with his Disk Menu Generator system. Among the other visitors to the stand were Prem Holdaway from London, Dave Hunter from Kent and O.W.Hulme from Hensford. In fact it was such a good meeting we hope to hold another one later in the year.

We are very short of Program submissions at the moment so please have a look to see if you have anything that may be useful to another member and send it in for printing in a future newsletter. If at all possible please try to send in Type-written articles or printed programme listings and remember to include your name and a short description of the programme.

We will start this issue off by trying to answer some of your letters sent in recently. Can you please note that if you require a personal reply a stamped addressed envelope must be included, the user group is none profit making and has not got the funds for extra postage. Also on the subject of postage we must ask any overseas members wishing to receive their newsletters by airmail to pay an additional £7.50 per six issues to cover the extra cost. This will make the total cost per year for airmail users £12.50.
LETTERS
------
Paul Sheridan, Dursley Glostershire

Is the series on machine code programming to be continued as I would like to learn more about it?

Yes we hope to get some more articles from Kevin Holloway as soon as he can find time to write some.

Alen Badcock, Chandlers Ford Hants.

Has anyone found a way of putting basic programmes in ROM. It would be nice to be able to load several programmes at once and switch between them. I am writing a batch file utility programme and also working on a simple multi-tasking program triggered from clock tics. Also does anyone know how to make the MDEX SAVEX command work so that compiled code can be run. Are the MPE 9901 parallel I/O cards still available.

There is a machine code program with details of how to save basic programmes in ROM and recall them using the * Command later in this issue. I don't know however how several could be loaded at once and pointers changed between them. The system pointers are held in such awkward places distributed throughout the Cortex memory map. It would however be possible for one Basic program to branch to another by using the * Command to load in the second program but any parameters would have to be passed by using some un-used memory location or possibly RAM-DISK.

We look forward to seeing your Batch file and Multi-tasking programmes when they are finished. I don't know about the MDEX SAVEX problem, mayby some other user will know and write in with the answer. We have not had much user information on MDEX yet. The MDEX system its self and anything else from MPE should be available through the group as soon as we can find time for a trip to Southampton to collect it all.

Patrick O'Leary, Eire.

I would appreciate any information on fitting Disk Drives to the Cortex but understand that there is no operating system for the new controller yet.

Any shugart compatible drives can be fitted to the Cortex both 8" and 5" systems are in use. A version of CDOS is available now for the new controller and can be obtained from the group for £45.00. So far we do not have a good PCB design for the new controller although the circuit has been built and tested using wire-wrap techniques.
LETTERS

D.Raison. Hampshire

If anyone is interested I have a small program that allows the graphics mode of the VDP to be used as a 64 column screen. If anyone is interested I will supply a listing or tape. Can you still supply Cortex hardware especially the R.G.B. Kit.

We're sure lots of people would be interested in your 64 column screen program, why don't you send it in along with anything else that may be of interest to other users. We can supply some Cortex hardware and will be issuing a list soon.

D.Fisher. Workington Cumbria.

Since I have had my Cortex I have had problems with the cassette interface even after trying several tape decks has anyone a foolproof cure.

Lots of people have had problems with Cassette loading in the past. One suggestion is to add a 1nf capacitor between pins 1 and 2 of I.C. 70 and also to connect pin 3 to +5V. I use a diode clipper on the input and it seems to work on most offending tapes see short tips in this issue.

O.C.Walden

My system is based on 8" double density disks so I have been forced to "roll my own" as far as system software is concerned although I do have MDEX. I would love to here from anyone who has 8" drives for sale with or without case or P.S.U.

We don't understand why you have a problem with system software as both MDEX and CDO$ work happily with 8" drives. Tim Gray has 8" drive capability on his Cortex.

P.D.Wrighton. Kent.

I would be interested in obtaining newsletters 1 to 4. Can you supply them?

Yes all back issues are available from the group for £1.50 each. Also available as a back issue is M.P.E.s one and only newsletter, Brainstorm one.
LETTERS


Can you supply any more information on any expansion boards especially the alternative disk controller. Also I have problems loading Cassette tapes.

We do have some alternative disk controller boards but at the moment there is a problem with the P.C.B. layout, although the circuit is correct. We will give more details in a future issue.

P.R. Cross. Kent.

I have a problem when running a program that uses a lot of Put and Get statements. After about 12 runs the error message :- "out of memory" is produced.

The early version of CDOS had this problem as it re-allocated space for the file variable for every disk access. In the new version 1.20 and later the problem has been cured. The updated version can be purchased from the group for £5.00. Please return your original master disk.

VARIABLE LIST ROUTINE

This machine code routine can be used to list all variable names used in a Basic program. All the variable names are stored in encoded form with the program on disc or tape so this routine can be used as soon as the Basic program is loaded without having to run it all the way through first. One interesting thing I have noticed is that Power Basic creates a variable name if it encounters a spelling mistake when expecting a statement. eg. If LOST were typed instead of list a variable name of LOS would be created. The only way to get rid of these "phantom " variables is to do a source save and load.

VARIABLE LISTING DEMO

CALL OF200H

SP [912] AD
E [6,3] RH
TH PH
D CX
CY S1
CO TP
SP CP
V [4,3] SV [4,2]
Y Z
XE YE
ZE S [4,4]
J N [4,3]
0 ;Variable listing programme
1 ;by Tim Gray
2 ;to use just type CALL <ENTRY>
3 ;variables will be listed to
4 ;all devices enabled
5 ;unused dimentioned variables
6 ;will be shown :-AGH[1] etc
7 ;
8
9 VNT EQU >EFBC ;VARIABLE NAME TABLE
10 NVD EQU >EFBE ;NEXT VAR DEFINITION
11 VLT EQU >EFBE ;VARIABLE LOCATION *
12 BUFF EQU >FE50 ;DATA BUFFER
13
14 F200 04C2 ENTRY: CLR R2
15 F202 0286 0003 LI R6,3 ;LOOP COUNT
16 F206 0287 FE50 LI R7,BUFF ;TEMP STORE
17 F20A 0208 0380 LI R8,0380 ;MASK FOR NUM
18 F20E C0B0 EFBC MOV @VNT,R0 ;NAME TABLE
19 F212 C2A0 EFBE MOV @NVD,R10 ;NEXT DEFINITION
20 F216 0280 S R0,R10 ;MAX NUM OF VAR
21 F218 0200 0008 LI R0,0 ;OFFSET TO 1ST
22 F21C 04C9 GETNAM: CLR R9
23 F21E C0C7 MOV R7,R3 ;GET NAME
24 F220 0223 0004 A1 R3,0004 ;not dimentioned
25 F224 C120 EFBC MOV @VNT,R4 ;NEGATE DIM'S
26 F228 A100 A R0,R4 ;IDENTIFY DIM'S
27 F22A C054 MOV *R4,R1 ;TEST FOR NUM
28 F22C 1502 TSTDIM: JGT TSTNUM ;yes numeric
29 F22E 0501 NEG R1 ;NEGATE DIM'S
30 F230 0709 SETO R9 ;IDENTIFY DIM'S
31 F232 2040 TSTNUM: COC R8,R1 ;TEST FOR NUM
32 F234 1311 JEQ NUM ;SHIFT TO ALIGN
33 F236 0241 7FF Alpha: ANDI R1,07FF ;TEST FOR VALID
34 F23A C081 MOV R1,R2 ;CLEAR TOP BIT
35 F23C 0A82 SLA R2,8 ;COPY
36 F23E 0242 1F00 ANDI R2,1F00 ;SHIFT TO ALIGN
37 F242 0262 4000 ORI R2,4000 ;ADJUST FOR
38 F246 0282 4000 CI R2,4000 ;ASCII
39 F24A 1601 JNE NOTSP ;TEST FOR VALID
40 F24C 1001 JMP NXT ;ignor it-
41 F24E DDC2 NOTSP: MOV R2,*R7+ ;if no char
42 F250 0951 NXT: SRL R1,5 ;SEND TO BUFF
43 F252 0606 DEC R6 ;ALIGN NXT CHAR
44 F254 16F0 JNE ALPHA ;DEC LOOP COUNT
45 F256 1009 JMP BUFF4 ;LOOP FOR NEXT
46 F258 4048 NUM: SIZ R8,R1 ;REMOVE CODE
47 F25A 0081 MOVB R1,R2 ;COPY
48 F25C 0922 SRL R2,2 ;SHIFT TO ALIGN
49 F25E 0262 4000 ORI R2,4000 ;ADJ FOR ASCII
50 F262 DDC2 MOVB R2,*R7+ ;SEND TO BUFFER
51 F264 0241 007F ANDI R1,007F ;MAX NUM 127
52 F268 2F41 XOR R1,13 ;OUT THE NUMBER
BUFF4: C R7, R3 ; 4 CHAR ?
54 F26C 1404 JHE NAMEND ; YES
55 F26E 0202 2000 LI R2, >2000 ; SPACE
56 F272 DDC2 MOV R2, *R7+ ; SEND SPACE TIL
57 F274 10FA JMP BUFF4 ; 4 CHAR IN BUFF
58 F276 1000 NAMEND: JMP DIM ;
59 F27B C249 DIM: MOV R9, R9 ; TEST FOR DIM
60 F27A 1328 JEQ FORMAT ;
61 F27C 0202 5000 LI R2, >5000 ; ""
62 F280 DDC2 MOV R2, *R7+ ; SEND OPEN BKT
63 F282 C160 EFBE MOV WLT, R5
64 F286 A140 A R0, R5
65 F288 C055 MOV R5, R1 ; GET VAR ADDR
66 F28A 1309 JEQ NPOINTER ; jump not used
67 F28C C0B1 GETDIM: MOV R1+, R2 ; GET DIMENSION
68 F28E 2F42 XOP R2, 13 ; OUT DIM INT
69 F290 0202 2C00 LI R2, >2C00 ; ""
70 F294 DDC2 MOV R2, *R7+ ; SEND COMMA
71 F296 C081 MOV R1+, R2 ; TEST FOR END
72 F29B 1101 JLT END ;
73 F29D 10F8 JMP GETDIM ; LOOP FOR NEXT
74 F29F 0607 ENDB: DEC R7 ; BACK UP POINT
75 F2A9 0202 5000 NPOINTER: LI R2, >5000 ; ""
76 F2A2 DDC2 MOV R2, *R7+ ; SEND CLOSE BKT
77 F2A4 1013 JMP FORMAT
78 F2A6 04C2 PRINT: CLR R2 ; CLEAR FOR NUL
79 F2AA 0002 MOV R2, *R7+ ; SEND NUL
80 F2AE 0002 DATA >0002 ; WRIT CRLF
81 F2AC 0F00 FE50 MSG @BUFF ; WRIT BUFFER
82 F2B0 0206 0003 LI R6, 3 ; RELoad R6
83 F2B4 0207 FE50 LI R7,BUFF ; RELoad R7
84 F2BB 05C0 COM1: INCT R0 ; INC NXT NAME
85 F2BA 0280 C R0, R10 ; TEST FOR END
86 F2BC 1AF8 JHE GETNAM ; LOOP FOR NEXT
87 F2BE 0287 FE50 CI R7,BUFF ; IS BUFF EMPTY
88 F2C3 1302 JEQ END1 ; YES
89 F2C4 0FA0 FE50 MSG @BUFF ; NO PRINT IT
90 F2CB 0002 END1: DATA >0002 ; WRIT CRLF
91 F2CA 0380 RTWP
92 F2CC 0287 FE64 FORMAT: CI R7, >FE64 ; BUFFER FULL?
93 F2DB 14EA JHE PRINT ; YES
94 F2D2 0202 2000 LI R2, >2000 ; SPACE
95 F2D6 DDC2 SNDSP: MOV R2, *R7+ ; SEND SPACE-
96 F2DB 0287 FE64 CI R7, >FE64 ; UNTIL HALF FUL
97 F2DC 1AFC JL SNDSP ;
98 F2DE 0206 0003 LI R6, >0003 ;
99 F2E2 10EA JMP COM1 ; NXT

VNT EFBC NVD EFBE VLT EFBE BUFF FE50
ENTRY F200 GETNAM F21C TSTDIM F22C TSTNUM F232
ALPHA F236 TSP F24E NXT F250 NUM F258
BUFF4 F26A NAMEND F276 DIM F278 GETDIM F28C
ENDB F29C NPOINTER F29E PRINT F2A6 COM1 F2B8
END1 F2CB FORMAT F2CC SNDSP F2D6 TSTDIM F22C
NAMEND F276
CDOS - DISK INSPECT (DI) UTILITY

1. If like mine, your CDOS manual makes no reference to DI, and you are struggling in the dark may I attempt to shed some light on the subject, for you.

DI

2. If you start by amending a copy of DI to match the listing below, I believe it makes it easier to use, and therefore understand.

3. The amendments do a number of things to DI:
   a. It displays the Hex and ASCII values at the same time.
   b. It accepts Hex values for the track & sector requests. Then displays the decimal values and the sector contents.
      (1) To enter a decimal number, enter the number and press RETURN.
      (2) To enter a Hex number start with a 0 end with a H and press RETURN. Note if the maximum number of digits is entered then the RETURN is automatic.
   c. You can modify the contents displayed either on the Hex or the ASCII displays. The other display is updated as you make the modifications. Which display to modify is indicated by an arrow at the bottom of the screen.
   d. The sector byte numbers down the left of the screen are in decimal. The number displayed being that of the first byte on that line. (0 to 127)

4. You should note that DI does not function for double density (DD) though it does work for double sided (DS) and 80T.

BASIC'S

5. CDOS will produce on a disk:
   a. 40T Single Sided (SS): 40 tracks (0 to 39)
   b. 40T Double Sided (DS): 80 tracks (0 to 79)
   c. 80T " " (DS): 160 tracks (0 to 159)
   d. Each track has 16 sectors (0 to 15)
   e. Each sector has 128 bytes (0 to 127)
   f. Each byte has 8 bits
      (1) A byte of information is displayed on the screen by DI as a 2 digit Hex number or a single digit ASCII character.
      (2) A Cortex Word is made up of 2 bytes. (16 bits)
   g. Track 0 is always the BOOT strap track.
   h. Track 1 is always the directory track.

6. Thus the LDIR indication of System Over Head of 32 sectors.
   (Blks in LDIR means sectors)
DISK DIRECTORY

7. You can now use the modified DI to see on the screen what follows:

8. Track 1 Sector 0: Bit map of sectors in use. See the User Mag' 6 page 4 for explanation.

9. Track 1 Sector 1 to 15 is the disk directory, on each sector there is room for the details of two files. Thus the limit of 30 files that can be saved. (DD of course doubles this)

10. Detail of any one of these sectors is:

a. Bytes 0 & 1 00 00 means: Free slot.
   A5 A5 " Auto Run file.
   5A 5A " Not Auto Run file.
   FF FF " Sequential Relative Data file.
   other " Random Relative Data file.
   and is the record length.

b. Bytes 2 - 9 Title of the file.

c. Bytes 10 & 11 00 00 means: File is Code.
   other " File is Basic.


e. Bytes 32 - 35 Contain the file disk locations for the first portion of the file.
   32 & 33 XX XY XXX Track number in Hex.
   Y Sector number in Hex.
   34 & 35 Total number of sectors in this portion of the file.

(1) The following 7 batches of 4 bytes are the same, giving a total of 8 portions that any file can be split up into to fit it on a disk.
(2) With this modified DI you can input 0XXXH direct to the track request and also 0XH to the sector request. (To reset the sector you press R or RETURN to the command request)

f. Bytes 64 - 127 Next file entry slot.

11. Now that you know what each byte means and also how to find your files on the disk, you can happily move about the disk and make modifications.

12. From the command request mode select whether you want the next sector (press I), the last sector (press D) to modify in Hex (press H) or in ASCll (press A). Then press M to modify. You now use the arrow key to select the byte to modify. Once on the byte change it as you require then:

a. RETURN passes the changes to the disk and returns you to the command scanner.

b. ESCAPE aborts the program i.e. the changes are not passed to the disk. (type RUN then reselect the sector)

13. Note you see how the changes are affecting the other display as you make them.
"MODIFIED DI BY J. MAKENZIE"

Note "↓" is produced by GRAPH + SHIFT + 1 KEYS

100 TEXT : ? @(1,17); "CDOS disk inspect utility 1.1 (C) 1985"
110 ? @(0,23); "[Ascii, Dec, Hex, Inc, Mod, Reset]"
115 ? @(16,22); "!"
120 DIM X[4],B[50]; $M="H"
130 AX=ADR[X[0]]; AB=ADR[B[0]]
140 MWD[AX]=0420H; MWD[AX+1]=06260H
145 MWD[AX+4]=00BC6H; MWD[AX+6]=02H
150 MWD[AX+B]=0380H
160 ? @(0,19); "Drive " ; ? " Track " ; ? " Sector "
165 ? @(11,19); : INPUT #1;D
167 IF D>3 THEN GOTO 165
170 ? @(8,20); : INPUT #4;T
180 IF T<0 OR T>159 THEN GOTO 170
190 ? @(9,21); : INPUT #3;S
200 IF S<0 OR S>15 THEN GOTO 190
205 GOTO 470
210 E=0
220 CALL AX,D,T,S,ADR[E],AB,0,0
230 IF E>0 THEN ? @(16,19); "READ ERROR"; $E/256 LAND 03FH: GOTO 350
240 ? @(16,19); "
250 BB=AB: CB=AB: ? "$H"
260 FOR R=0 TO 15
270 ? "$000"R*$8": "
280 FOR C=0 TO 7
290 ? "$MEMBB": "
310 BB=BB+1
320 NEXT C
325 ? "$": GOSUB 515
330 ?
340 NEXT R
350 ? @(20,20); "": ? "$10L": INPUT "Command":1,$K;
360 IF $K="I" THEN S=S+1: GOTO 430
370 IF $K="D" THEN S=S-1: GOTO 430
380 IF $K="": THEN GOTO 160
390 IF $K="A" THEN $M=$K: ? @(0,22); "!
400 IF $K="H" THEN $M=$K: ? @(0,22); "!": GOTO 350
410 IF $K="M" THEN ? @(26,22); "↓": GOTO 720
420 GOTO 160
430 IF S<0 THEN T=T-1: S=15
440 IF S>15 THEN T=T+1: S=0
450 IF T<0 THEN T=0
460 IF T>159 THEN T=159
470 ? @(8,20); "0000"T: ? @(9,21); "000"S
480 GOTO 210
490 CALL AX,D,T,S,ADR[E],AB,0,0FFH
500 IF E<0 THEN ? @(28,19); "WRITE ERROR"; $E/256 LAND 03FH
510 GOTO 350
515  FOR C=0 TO 7
520  IF MEM[CBI]<>20H THEN "$Q="".
530      ELSE "$Q=%MEM[CBI]%0
540       ? "$Q;
545      CB=CB+1
546   NEXT C
550   RETURN
560   BB=AB: R=0: C=30
570   IF MEM[BB]<>01FH THEN "$SS=%MEM[BB]%0
580      ELSE "$SS="".
590   ? @(C,R); "$SS; ? @"L";
600   K=KEY[0]: IF K=0 THEN WAIT 1: GOTO 600
610   IF K=08H THEN C=C-1: BB=BB-1
620   IF K=09H THEN C=C+1: BB=BB+1
630   IF K=0AH THEN R=R+1: BB=BB+8
640   IF K=0BH THEN R=R-1: BB=BB-8
650   IF K=0DH THEN @(26,22)" ": GOTO 490
660   IF K<01FH THEN MEM[BB]=K: GOTO 1000
670   IF C<30 THEN C=37: R=R-1
680   IF C>37 THEN C=30: R=R+1
690   IF R<0 THEN R=15: BB=AB+127
700   IF R>15 THEN R=0: BB=AB
710   GOTO 570
720   IF "$M=""A"" THEN GOTO 560
730   BB=AB: R=0: C=5
740   ? @(C,R); "$MEM[BB]; ? @"2L";
750   K=KEY[0]: IF K=0 THEN WAIT 1: GOTO 750
760   IF K=08H THEN C=C-3: BB=BB-1
770   IF K=09H THEN C=C+3: BB=BB+1
780   IF K=0AH THEN R=R+1: BB=BB+8
790   IF K=0BH THEN R=R-1: BB=BB-8
800   IF K=0DH THEN @(26,22)" ": GOTO 490
810   IF K<02FH THEN IF K<03AH THEN GOSUB 880
820   IF K<040H THEN IF K<047H THEN K=7: GOSUB 880
830   IF C<5 THEN C=26: R=R-1
840   IF C>26 THEN C=5: R=R+1
850   IF R<0 THEN R=15: BB=AB+127
860   IF R>15 THEN R=0: BB=AB
870   GOTO 740
880   K=MOD[K,16]
890   MEM[BB]=MOD(MEM[BB],16)*16+K
892   XX=(C-5)/3+30
894   IF MEM[BB]<020H THEN "$Q="".
896      ELSE "$Q=%MEM[BB]%0.
898   ? @(XX,R); "$Q;
900   RETURN
900   XX=(C-30)*3+5
910   ? @(XX,R); "$MEM[BB];
912   GOTO 570
CORTEX USER GROUP

*** STAR BASIC ***

ROUTINE FOR LOADING A BASIC PROGRAMME FROM EPROM AS A * COMMAND

The header block is the same as in the disc directory.
The programme is saved from 2090 up to 2090+load length.

2000 94C2 ; EPROM IDENTIFIER
2002 3FFF ; NEXT *PROG
2004 2016 ; START ADDR OF THIS ONE
2006 5053 ; NAME
2008 5749 ; "PSWING"
200A 4E47 ;
200C 0000 ;

PTR: 200E 0D5A ; POINTERS (EFBA-ED04)
2212 9F40 ; (EFBC-ED04)
2012 0042 ; (EFC0-EFBE)
2014 0FC4 ; LENGTH (EFC0-ED04)

START 2016 C120 MOV @ED04,R4 ; BASIC PROGRAMME LOAD ADDR
201A C0E0 MOV @2014,R3 ; LOAD LENGTH
201E 0200 LI R0,>2090 ; PROGRAMME START
2022 DD30 MOVB *R0+,*>R4+ ; LOAD THE PROG
2024 0280 CI R0,>4000 ; CHECK IF PROG IS OVER THE
2028 1A05 JL >2034 ; 8K LIMIT AND
202A 05A0 INC @>F106 ; INC THE MAPPER IN 4K BLOCKS
202E 0205 LI R5,>1000 ; IF REQ'D
2032 6005 S R5,R0 ; RESET PTR TO NEW 4K BLOCK
2034 0603 DEC R3 ; END ?
2036 16F5 JNE >2022 ; NO BACK FOR MORE
2038 0202 LI R2,>200E ; CALCULATE SLT
203C C0F2 MOV *R2+,R3
203E C060 MOV @>ED04,R1
2042 A0C1 A R1,R3
2044 C003 MOV R3,@EFBA ; STORE SLT
2048 C0F2 MOV *R2+,R3 ; CALCULATE VNT
204A A0C1 A R1,R3
204C C003 MOV R3,@EFBC ; STORE VNT
2050 C004 MOV R4,@EFBE ; CALCULATE VDT
2054 6032 S *R2+,@EFBE
2058 C004 MOV R4,@EFC0 ; STORE VDT
205C 020E LI R14,>3EF2 ; AUTO RUN RETURN
2060 020D LI R13,>F0DC ; BASIC WP
2064 02CF STST R15
2066 0208 LI R11,>0202 ; MAPPER RESET VALUES
206A 020C LI R12,>0303
206E 020A LI R10,>F104
2072 0200 LI R0,>03C0
2076 0201 LI R1,>CEBB
207A 0202 LI R2,>6BC
207E 0203 LI R3,>0380
2082 0440 B R0

PROG 2090 ; PROGRAMME SAVED HERE
Alan Badcock has sent in this design for a piggy-back board to allow an extra system ROM to be fitted to the main board. All the ROM socket lines from ROM1 are paralleled to ROM4 except for CS4 which comes from pin 7 of IC33. If wire-wrap sockets are used for IC33 and IC76 on the piggy-back board the P.C.B. will plug into the old IC33 and IC76 sockets. The new ROM is mapped in between 6000H and 7000H when ROM select is on.
Soeren Rahbek from Denmark sent in this circuit diagram and P.C.B. layout for a better data seperator for the TMS9909 floppy disc controller. He also sent in some double sided P.C.B.s for it which are available to anyone who has some software or hardware to swap for them. If you want one send your swap to the user group and we will pass it on. The FDC9126 data seperator is available from Manhattan Skyline.

Remove IC88 and make following changes on main board:

Disconnect:
IC76 pin 26 to IC69 pin 9

Connects:
IC77 pin 4 to IC88 pin 9
IC87 pin 4 to IC88 pin 6
IC76 pin 26 to IC88 pin 5

Plug DATA SEPARATOR P.C board in IC88 socket and you are in the air.

IC88: fix error any more.

----------

10 - 14
D. Fisher sent in the following:-

To change the RS232 port to 8 bit change memory location 5546 from 62F1 to 63F1.

To change the cursor character change memory location 5530 from 007F to the ascii code of the preferred character.

Use the following to replace the standard microsoft statement CHR$( ) when entering listings from magazines.

5 REM EQUIVALENT OF CHR$( )
10 DIM $A(256)
20 FOR A=0 TO 255
30 $A(A)=%A !$A(A) IS USED INSTEAD OF CHR$(A)
40 NEXT A

Tim Gray uses the following to improve cassette loading

```
<table>
<thead>
<tr>
<th>To</th>
<th></th>
<th>To</th>
</tr>
</thead>
<tbody>
<tr>
<td>Cassette</td>
<td></td>
<td>Computer</td>
</tr>
</tbody>
</table>
```

The diodes clip off any modulation distortion caused by mains hum etc.

The value of workspace pointer for Interrupt 1 is incorrect to set things right change location 0004 from ED24 to ED22 this will allow the Interrupt 1 user vector to be used.

Alan Badcock sent in the following :-

To cure a problem with the cassette motor relay not functioning correctly try changing R24 to 10K and R43 to 2K2 also a diode fitted across the relay coil may help, cathode to +V.

A bug in the re-number routine can be cured by changing memory location 2F3E from 1006 to 1007.

The 9909 disk controller timings can be changed directly by the following program. It sets it for 50MS step 35MS settle and 0MS load. The routine may be usefull to people who have trouble booting C Dos for the first time before using Config.

```
10 MEM F140H = 050H
20 MEM F141H = 015H
30 MEM F142H = 011H
40 MEM F143H = 047H
50 MEM F144H = 064H
60 MEM F145H = 046H
70 MEM F146H = 0H
80 MEM F147H = 0AH
```

10-15
CORTEX USERS CLUB SALE

RGB INTERFACE BARE BOARD £8.00 KIT £20.00
CENTRONICS INTERFACE BARE BOARD £7.00 KIT £15.00
E BUS -ALL IC`S KIT £30.00
E BUS BACK PLANE £15.00

SEMI CONDUCTORS

TMS9901 £3.00
TMS9902 £3.00
74LS612 (3 AVAILABLE) £25.00
74LS611/74LS611 (NEED PULL UP RESISTORS) £15.00

E BUS EXPANSION

E BUS (4K RAM, 8K EPROM SCKT, 16 IN/OUT LINES) £15.00
NOTE-THese CARDS ARE EX EQUIPMENT TESTED AND WORKING
E BUS (8*8K EPROM SCKT CARD BUILT NO EPROMS FITTED) £28.00
E BUS 512K DRAM(ONLY AVAILABLE COMPLETE AT PRESENT) £100.00

CORTEX EXPANSION

EXTERNAL VIDEO INTERFACE BARE BOARD £15.00 KIT £80.00
DISK CONTROLLER (WD 2797+BOARD) £40.00

CORTEX SOFTWARE

DISK OPERATING SYSTEM CDOS 1.20 AND 2.00 £45.00
CDOS 1.20 FOR 9909 SYSTEM
CDOS 2.00 FOR 2797 SYSTEM

MEMBERS SOFTWARE

WORTEX-WORD PROCESSING £15.00
INCLUDES SPELLING CHECKER
SEND TO J S MACKENZIE
4 WERSTAN CLOSE
MALVERN WR14 3NH
(INCLUDE TWO 5" DISKS)

DRAWTECH-GRAPHICS DRAWING PACKAGE £20.00
SEND TO TIM GRAY
1 LARKSPUR DRIVE
FEATHERSTONE
WOLVERHAMPTON WEST MIDLANDS WV10 7TN

MEMBERS HARDWARE

TAPE CONDITIONER KIT £10.00 BUILT £15.00
SEND TO-P MOYERS
7 PHILIP GROVE
SUTTON
ST HELENS
MERSEYSIDE WA9 3TD

10-16