SOUTH AUSTRALIAN

MICROPROCESSOR GROUP

INC.

P.O. BOX 113,
PLYMPTON, S.A. 5038
TEL 278 7288

Meetings held at
THEBARTON HIGH SCHOOL
ASHLEY ST., THEBARTON

NEWSLETTER

Vol. 1. No. 4
October/November 1979 Newsletter. Postal Adr. P.O. BOX 113, PLYMPTON S.A.

This issue of our newsletter has a host of good articles in it for you including a new parallel interface standard which we hope a few of you will adopt for data transfer. Also inside is the follow up of the last newsletters article on the Z80 code listings and how they were generated.

Our August meeting was very well attended with quite a number of new systems to be seen. I would think at this stage our next B.Y.O.S. night will be around the end of the year.

The September meeting with Tony Beresford at the helm was quite an interesting evening, unfortunately our video distribution setup was a little bit hairy. I was sure a 216 battery was good for more than 20 minutes at 50 milliamps it just goes to show that nothing should be taken for granted, good old Murphy.

October

Our meeting in October will be a guest speaker who has given us some very interesting talks in the past. The subject matter is still under discussion so come along there is nothing like a suprise to round off the week.

November

In November we will be having a visit from Neil Pollard of Applied Data Control. The evening will feature the Ohio Scientific Microprocessor equipment being handled by A.D.C. We may also get to see the Compucolour gear they also market so this should be an interesting evening.

For those of you that did not attend last months meeting we had the return of our Secretary Bob Stunell from his overseas trip I understand he was very happy to get back to his 6800 keyboard. Bobs return brings me to the sincere thanks which must go to John Moffatt for his aid and assistance in running the group while Bob has been away. I must also point out that John has played a very large part in arranging the parts for the group V.D.U. he has also donated the parts for the V.D.U. for group use.

YOUR COMMITTEE FOR 1979

CHAIRMAN: Eric Clarke.....278-7288
SECR.TREAS:Bob Stunell.....352-5811

COMMITTEE MEMBERS:

Tony Beresford
Bob Daniells
Howie Harvey
Rick Matthews

As mentioned earlier we are developing (as far as we know) a new system of data transfer between micro's. The foundation of this idea relies upon an 8 bit port with no extra handshake lines required for syncronisation. This is achieved by transmitting the data in 4 bit Nibbles using the other half of the byte for signals such as end of data strobe, actual data nibble available strobe, reqest for data back strobe and parity bit.

These control nibbles as yet have not been decided on as the requirments of different systems and port types may decide the final criteria. Such differences might be as with a programmable port where the request for data back strobe could initiate the recieving port into changing to a transmit port to send back the byte or nibble recieved so full error checking of transmitted data can take place.

If you have ever given or recieved software with 1 bit of 1 byte incorrect in shall we say 10k of complex code, then had to debug that code, you will no doubt see the advantage of full duplex transfer. To allow each user to have a standard interface lead with plug we have made up an interface box which has 2 cannon 25 pin sockets crosswired together with 2 Mcmurdo sockets in parallel.

The reason for this is that computor (a) can have a cannon plug whilst computor (b) can have Mcmurdo and because the interface box is cross wired the output lines from (a) will connect with the input lines of (b). The interface box has been wired for full 8 bit I/O plus 2 bits for handshake if this is required and the chart of fig (1) will give you the pin connections. For those of you who have a Z80 system the software of fig (2) and (3) will apply (this software is only for explanation purposes and needs extra control) or if you have 8080 then the relative jumps etc will need changing.

If any member knows of a similar idea already in use then please let us know as there is enough different standards around in this field. More software on this system will be forthcoming as we develope this system including some disk based control programs. We hope to get it to the stage where calling a file together with the control program will be all that is needed to transfer data from (a) to (b).

Fig (1) Numbering of cannon plug.

pin	function
1to8	D0-D7 out
9	tx strobe out
10	rx acknlg out
14to21	D0-D7 in
22	rx strobe in
23	tx acknlg in
13	ground
25	ground

Fig (3)

```
B035  3E80        0052 RXDATA: LD A,PIC1       ;INITIALISE PORT
B037  D3FB        0053         OUT (CONTL),A
B039  210001      0054         LD HL,START
B03C  0E80        0055         LD C,80H
B03E  CD4FB0      0056 RX1:    CALL RXNIBL     ;GET LOW NIBBLE
B041  5F          0057         LD E,A
B042  CD4EB0      0058         CALL RXNIBL     ;GET HIGH NIBBLE
B045  07          0059         RLCA
B046  07          0060         RLCA
B047  07          0061         RLCA
B048  07          0062         RLCA
B049  B3          0063         OR A,E          ;COMBINE HIGH & LOW NIBBLES
B04A  77          0064         LD (HL),A
B04B  23          0065         INC HL
B04C  18F0        0066         JR RX1
                  0067 ;
B04E  DBF9        0068 RXNIBL: IN A,(DATA)
B050  2F          0069         CPL
B051  F680        0070         AND A,80H       ;TEST STROBE BIT
B053  B9          0071         CP A,C          ;HAS STROBE GONE HIGH?
B054  4F          0072         LD C,A
B055  30F7        0073         JR NC,RXNIBL
B057  067F        0074         LD B,7FH        ;LOAD HALF DELAY
B059  10FE        0075 RX11:   DJNZ RX11       ;TIGHT DELAY LOOP
B05B  DBF9        0076         IN A,(DATA)
B05D  E60F        0077         AND A,0FH       ;REMOVE UPPER BITS
B05F  C9          0078         RET
                  0079 ;
0000  (0000)      0080         END
```

SALE SALE SALE SALE

One North Star Micro-disk system Controller with Two Drives, single density, S-100 Bus board, hard sectored system. Software includes North Star BASIC and DOS and many diskettes from North Star library, CP/M on North Star from Lifeboat Associates, and PASCAL from UCSD.
(UCSD PASCAL requires 48K of Memory and Z80 CPU)

altogether at $850.00

Offers for drives by themselves considered, as they are compatible with other Floppy Diskette controllers.

Tony Beresford 79 2936

CROMEMCO CDOS Z80 ASSEMBLER version 02.15 PAGE 0001

DIAGRAM FIG(2)

```
                        0001 ;
                        0002 ;
                        0003 ;    **PIO**      WILL ALLOW YOU TO
                        0004 ;TRANSFER DATA BETWEEN TWO SYSTEMS
                        0005 ;USING PARALLEL I/O PORTS.
                        0006 ;
                        0007 ;
      (00FB)            0008 CONTL:   EQU 0FBH
      (00F9)            0009 DATA:    EQU 0F9H
      (C000)            0010 MONITR:  EQU 0C000H
      (0100)            0011 START:   EQU 100H
      (0FFF)            0012 END:     EQU 0FFFH
      (0080)            0013 PIO1:    EQU 80H
                        0014 ;
0000'                   0015          ORG 0B000H
                        0016 ;
B000  3E80              0017 TXDATA:  LD A,PIO1        ;INITIALISE PORT
B002  D3FB              0018          OUT (CONTL),A
B004  AF                0019          XOR A,A
B005  D3F9              0020          OUT (DATA),A     ;CLEAR PORT
B007  11FF0F            0021          LD DE,END        ;INSERT END ADDRESS
B00A  210001            0022          LD HL,START      ;INSERT START ADDRESS
B00D  7E                0023 TX1:     LD A,(HL)
B00E  CD24B0            0024          CALL TXNIBL      ;TRANSMIT LOW NIBBLE
B011  7E                0025          LD A,(HL)
B012  0F                0026          RRCA
B013  0F                0027          RRCA
B014  0F                0028          RRCA
B015  0F                0029          RRCA
B016  CD24B0            0030          CALL TXNIBL      ;TRANSMIT HIGH NIBBLE
B019  23                0031          INC HL
B01A  E5                0032          PUSH HL          ;BLOCK TRANSMITTED YET?
B01B  B7                0033          OR A,A
B01C  ED52              0034          SBC HL,DE
B01E  E1                0035          POP HL
B01F  D200C0            0036          JP NC,MONITR
B022  18E9              0037          JR TX1
                        0038 ;
B024  E60F              0039 TXNIBL:  AND A,0FH        ;REMOVE UPPER NIBBLE
B026  F680              0040          OR A,80H         ;INSERT STROBE BIT
B028  D3F9              0041          OUT (DATA),A
B02A  CD30B0            0042          CALL TXN1
B02D  AF                0043          XOR A,A          ;SWITCH OFF STROBE & DATA
B02E  D3F9              0044          OUT (DATA),A
B030  06FF              0045 TXN1:    LD B,0FFH        ;LOAD DELAY VALUE
B032  10FE              0046 TXN2:    DJNZ TXN2        ;TIGHT DELAY LOOP
B034  C9                0047          RET
                        0048 ;
                        0049 ;RECEIVE DATA ROUTINE.
                        0050 ;
                        0051 ;
```

THAT AUXILIARY CARRY BIT

A PROGRAMMING QUICKIE IN JUNE 79 BYTE DISCUSSED METHODS FOR CONVERTING A HEX NYBBLE TO AN ASCII CHARACTER.

```
(A)     ADI     090H            (B)     DAA
        DAA                             ADI     0F0H
        ACI     040H                    ACI     040H
        DAA
```

METHOD (A) WORKED FOR ME. BUT WHEN I TRIED METHOD (B), WHICH ONLY USED 5 BYTES, IT BEHAVED ERRATICALLY. THE INSTRUCTION PREVIOUS TO THE CODE SHOWN ABOVE WAS A SIMPLE NYBBLE MASKING INSTRUCTION:-

```
        ANI     00FH
```

WHEN I FURTHER INVESTIGATED THE MATTER, I FOUND SOME RATHER INTERESTING DISCREPENCIES IN BEHAVIOUR AND DOCUMENTATION BETWEEN THE 8080 AND THE 8085 WHEN A LOGICAL * AND * FUNCTION IS PERFORMED.

(1) THE 8080 SETS AUXILIARY CARRY IF EITHER OPERAND REGISTER HAS B3=1 HOWEVER THE 8080 ASSEMBLY LANGUAGE MANUAL CLEARLY STATED THAT THE AUXILIARY CARRY BIT WAS ALWAYS CLEARED DURING ANY LOGICAL OPERATION.

(2) THE 8085 ALWAYS SETS THE AUXILIARY CARRY BIT, REGARDLESS OF THE DATA IN EITHER OPERAND REGISTER.
THE 8085 ASSEMBLY LANGUAGE MANJAL IS CORRECT IN NOTING THAT THE AUXILIARY CARRY IS SET AT ALL TIMES.

(3) AN OBSCURE NOTE AT THE END OF THE 8085 DESCRIPTION OF THE BEHAVIOUR OF THE AUXILIARY CARRY BIT NOTES THE DIFFERENCE IN THE DEVICES. THIS DIFFERENCE IS NOT NOTED IN THE SECTION DEALING WITH DIFFERENCES BETWEEN THE 8080 AND 8085.

INCIDENTALLY, I HAVE NOTED THAT THE Z80 BEHAVES AND IS DESCRIBED LIKE THE 8085, NOT THE 8080, IN THIS EFFECT.

WHICH ALL GOES TO SHOW THAT THE 8080 AND 8085 ARE, FOR SOFTWARE EXECUTION, IDENTICAL - ALMOST.

AN INVITATION FOR YOU

IF YOU KNOW OF ANY SIMILAR DIFFERENCES IN BEHAVIOUR OR ERRORS IN DOCUMENTATION ON THE 8080 , 8085 OR Z80 THEN HOW ABOUT TELLING US ALL ABOUT IT. YOU NEVER KNOW, YOU MIGHT SAVE SOMEONE ELSE A LOT OF FRUSTRATION (EVEN IF NOT YOUR OWN) AND IT HELPS TO FILL THE PAGES OF THIS NEWS-LETTER.

FOR EXAMPLE, WHO WILL VOLUNTEER TO TELL US ALL ABOUT THE FLAG BYTE DIFFERENCES FOR CARRY/OVERFLOW CONDITIONS BETWEEN THE 8080 AND Z80?

Howard Harvey

```
                ;
                CNTROL:
7E4B FE0D           CPI     CR      ;CARRAGE RETURN????
7E4D C2457E         JNZ     COUT    ;IF NOT RETURN
7E50 3600       RESCTR:     MVI     M,0     ;RESET COUNTER TO ZERO
7E52 C3457E         JMP     COUT
                ;
                OCRLF:              ;ROUTINE TO OUTPUT CR LF AND ARROW
7E55 C5             PUSH    B       ;SAVE CHAR IN C
7E56 0E0D           MVI     C,CR
7E58 CD0CF0         CALL    OPC
7E5B 0E0A           MVI     C,LF
7E5D CD0CF0         CALL    OPC
7E60 0E3E           MVI     C,ARW
7E62 CD0CF0         CALL    OPC
7E65 C1             POP     B
7E66 0601           MVI     B,1     ;SET COUNTER TO 1
7E68 C3427E         JMP     RETP
                ;
                ;
7E6B            CNTR    DS      1       ;RESERVE MEM FOR COUNTER
```

PROCESSING Z80 CODES

I had not seen a list of Z80 codes in alphanumeric order of opcode. Such a list is invaluable for hand disassembly. My Z80 assembler came with a file of Z80 mnemonics, starting with this, after much sweat and toil we finally printed the enclosed list.(last newsletter)

The first step was to assemble the complete set of mnemonics. A 39 k PRN file was then available. Using the CP/M Editor the extra characters were eliminated, the page headings, addresses, statement numbers etc. The macro instruction in the Editor was invaluable for this task. The first obstacle was the fact that my EBasic considered a carriage return and a coma as an end of record. This difficulty was overcome by changing all comas to semicolins, once again with a macro Ed instruction.

The shown program UBTBCDS was then used to change the unblocked file to a fixed length blocked file. This was necessary so that random access techniques could be used when rewriting the codes in order. The program is fairly straight forward, but does show fairly well how EBASIC handles files. The FILE statement defines "Z80CDSBD" as a blocked file with fixed record lengths of 36 characters. Extra space is required for the quotation marks around the string and for the carriage return and linefeed at the end of each record.

The program CDSORT reads in the codes to sort then sorts them, producing an output file of indexes to use for writing an ordered file. The quick sort routine is one used for another project, and is another story!!! Basically it is just a copy of a quicksort routine written by Wirth, and translated into Basic by me! Compared with primative sort routines, like a bubble sort, quick sort is lightning fast. The program shows another of the advantages of EBASIC, the capacity to use meaningful variable names(up to 31 characters long). Being able to use indentation also helps to clarify logical flow.

WRCDSIO wrote an output file of the codes in order. It first read in the ordered list of indexes then using random access, read the codes in alphanumeric order and wrote an ordered output file.

A final program, not included, organized the ordered list into two columns on six pages. This was also quite an interesting exercise in string manipulation!

I hope the codes in order will aid some in debugging their Z80 programs, and that the basic programs will help in understanding EBASIC file handling and string manipulation.

I. Fisk

```
REM NAME IS UBTBCDS
REM PROGRAM TO GET CODES AND BLOCK CODES ETC
REM CHANGES UNBLOCKED FILE TO BLOCKED FIXED LENGTH FILE
REM SO THAT RANDOM ACCESS CAN BE USED TO REWRITE CODES IN ORDER
REM ALSO WRITES FILE OF KEY TO SORT(OP CODE)
INPUT "NAME OF INPUT FILE(Z80CDS) ";SOURSE$
DIM CD$(1000)
OUT1$="Z80CDSBD"
OUT2$="CODES"
FILE SOURSE$, OUT1$(36), OUT2$
IF END #1 THEN 100
MAXLEN=0
NUM=0
10 READ #1;CODE$
NUM=NUM + 1
LENGTH=LEN(CODE$)
IF LENGTH GT MAXLEN THEN MAXLEN=LENGTH
PRINT #2,NUM;CODE$
CD$(NUM)=LEFT$(CODE$,8)
GO TO 10
100 PRINT "MAXIMUM LENGTH OF LINE IS ";MAXLEN;"CHARACTERS"
PRINT "NUMBER OF CODES READ IS ";NUM
PRINT "NOW WRITE FILE OF KEY WORDS TO SORT BY LATER"
FOR I=1 TO NUM
 PRINT #3;CD$(I)
NEXT I
PRINT "ALL FINISHED"
STOP
END
```

```
REM NAME IS WRCDSIO.BAS  IE WRITE OUTPUT FILE OF Z80 CODES IN ORDER
DIM INDEX(700) REM ONLY 697 NEEDED
IF END #1 THEN 987
INFILE$="CODES.IAO"
INFILE2$="Z80CDSBD"
OUTFILE$="Z80CODES.IAO"
FILE INFILE$, INFILE2$(36), OUTFILE$
PRINT "FIRST READ IN ORDERED LIST OF CODES":PRINT
FOR I= 1 TO 697
  READ #1;INDEX(I)
NEXT I
987 PRINT "NOW TO GET CODES IN ORDER"
PRINT "AND WRITE THE OUTPUT FILE"
PRINT
FOR I=1 TO 697
  READ #2,INDEX(I);CODE$
  PRINT #3;CODE$
NEXT I
PRINT "ALL DONE"
STOP
END
```

```
REM NAME IS CDSORT
REM SORTS FILE OF CODES INTO ORDER ALL FIXED LENGTH(8)
REM ABOUT 697 OF THEM
DIM A$(700)
DIM INDX(700)
INFILE$="CODES"
OUTFILE$="CODES.IAO"
IF END #1 THEN 1999
FILE INFILE$, OUTFILE$ REM OPENS FILES
FOR I=1 TO 697
  READ #1;A$(I)
  INDX(I)=I
NEXT I
1999 PRINT "ALL CODES READ IN NOW TO SORT THEM"
N=697 REM NUMBER TO SORT
GO SUB 111
FOR I=1 TO 697
 PRINT #2;INDX(I)
NEXT I
PRINT "ALL FINISHED"
STOP
111 REM **********QUICK SORT SUBROUTINE******
REM FIRST CAL SIZE OF STACK REQUIRED M=LOGB2(N)
M=INT(LOG(N)/LOG(2) + .9999) REM MAX SIZE
DIM STACKL(M)
DIM STACKR(M)
S=1 : STACKL(1)=1 : STACKR(1)=N
20 L=STACKL(S) : R=STACKR(S) : S=S-1 REM TAKE TOP RQST FRM STACK
  30 REM SPLIT A(L)..A(R)
     I=L : J=R : MID=INT((L+R)/2) : X$=A$(MID)
     40 IF A$(I) LT X$ THEN I=I+1 : GO TO 40
     50 IF X$ LT A$(J) THEN J=J-1 : GO TO 50
        IF I LE J THEN W$=A$(I) : A$(I)=A$(J) : A$(J)=W$ : ¢
            TINDX=INDX(I):INDX(I)=INDX(J):INDX(J)=TINDX:¢
            I=I+1 : J=J-1
     IF J GE I THEN GO TO 40
   IF (J-L) LT (R-I) THEN GO SUB 3000 ELSE GO SUB 4000
   IF R GT L THEN GO TO 30
IF S NE 0 THEN GO TO 20
RETURN
3000 REM ROUTINE TO PUT RIGHT PARTITION ON STACK
IF I LT R THEN ¢
   S=S+1 : STACKL(S)=I : STACKR(S)=R
R=J REM CONTINUE SORTING LEFT PARTITION
RETURN
4000 REM ROUTINE TO PUT LEFT PARTITION ON STACK
IF L LT J THEN ¢
   S=S+1 : STACKL(S)=L : STACKR(S)=J
L=I REM CONTINUE SORTING RIGHT PARTITION
RETURN
END
```

THE MICRO SHOP

MICROCOMPUTER SERVICES,
COMPONENTS, KITS,
AND ASSEMBLIES

TELEPHONE: (085) 22 3955

XITEX UNITS

SCT-100 "Glass Teleprinter" single-board VDU kit, ASCII or baudot as reviewed in AREWISE and AMATEUR RADIO, $185 kit, $220 assembled.

MRS-100 Morse code converter, microprocessor controlled, interfaces to ASCII or baudot terminal, $279 kit, $329 assembled.

SKT-100 Complete ASCII or baudot terminal with keyboard in rugged moulded case, 240V power, RS-232 interface, connects to standard video monitor or modified TV set. Assembled only $390.

* *

G.R.I. KEYBOARDS

Range of good quality gold-contact keyboards. Model 753 is ASR-33 Teletype layout, 53 keys, suitable for use with Xitex, fully ASCII encoded, SPECIAL 4 only with cases $80 kit, $88 assembled. Model 756 is VDU data entry type, specially suitable for use with SCT-100 for microprocessor work, 56 keys, fully ASCII encoded, $70 kit, $78 assembled. 30 pin edge connector, suit SCT-100 or GRI keyboards $4.50 each. DC-512 Converter, mounts on 753 or 756 keyboard and provides -12V for encoder chip, makes keyboard single +5V supply, $12 each. Moulded plastic case for 753 or 756 keyboard (state which) $20 each.

* *

THINKER TOYS KITS

All Thinker Toys kits are fully S-100 Bus compatible, as required by the draft IEEE standard. Boards are double-sided on fibreglass laminate with plated-through holes, solder mask and silk-screened component legends. Only first-grade components are included.

16K Static Ram uses low-power 4K chips, operates up to 4MHz clock rate. $340 kit, $370 assembled.

32K Static RAM uses low-power 4K chips, operates up to 4MHz clock rate. $711 kit, $749 assembled.

SWITCHBOARD provides 8 ports of serial and parallel I/O for a microcomputer system, but also includes sockets for 4K of 2114-L RAM and 4K of 2708 EPROM. $235 kit, $285 assembled.

Disk Jockey 2D Controller is a dual-density floppy disk controller for 8" Shugart drives. Complete with on-board 1K buffer RAM and 1K PROM firmware, also has own serial I/O port for console. $415 kit, $470 assembled.

Wunderbuss is a shielded and terminated low-noise S-100 motherboard with on-board regulators for +5, +12 and -12V peripherals. Complete with all sockets 8-slot $100 kit, $140 assembled. 12-slot $128 kit, $180 assembled.

DISCUS 2-D is a complete twin-drive dual-density 8" floppy disk system with S-100 bus controller, cables, 240V power supply, DOS and Disk BASIC software. Stores up to 1 Mbyte of data. Assembled form only $2285.