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. 5
December/January 1979 Newsletter. Postal Adr. P.O. BOX 113. PLYMPTON S.A.

Well its christmas again, I hope you have all got your orders in with your better half for some extra RAM or a FLOPPY or two. In this issue we have a source listing of the nibble data transfer receiver for 8080 or Z80 systems. Also details of using the Rxdat program with CP/M including the method of combining it into a COM file. For those of you who cannot make it to our December meeting the committee and I wish you all a merry Xmas and a happy new year.

Our October meeting was well attended and Clive Pearson gave us a very instructive lecture on Binary concepts. I think that lectures of that calibre particularly with the new membership, are something that we should perhaps hold more often. If any members have any preference towards lecture subjects please let us know. Send a letter or give us a note at the meetings because verbal notification can be forgotten.

The November meeting with Neil Pollard from Applied Data Control illustrated quite well the growth of the computer industry. The range of micro equipement they handle shows how seriously the main frame companies are taking these new products. We take this opportunity to thank Neil for his efforts and wish him and his staff all the best for 1980.

December

The meeting in December will be a bring your own systems night so all you micro owners get the breadboards and ratsnests cleaned up and bring your gear along. We would also like to extend an invitation to any other groups and their members to share our evening.

January

In January Protronics of Adelaide will be giving us a display of their PET microcomputer gear, it should include the Pet floppy drives, and printer. They may also bring along a 6800 Dream machine which I understand has had a good following in Adelaide. So all those who are still without a system here is a chance to see the PET and DREAM in action.

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

Help ! I am finding it increasingly difficult to compile the newsletter with only a few faithfull contributors. We would like to see more articles from other members, they need not be long wordy pieces alot of short articles will serve just as well. To recap on the years entries from members here is a thankyou to those who contributed to our newsletter.

Tony Beresford. The CP/M System. Computer time trials. etc.

Bob Daniells. Disk and Dynamic Ram.

Bob Stunell. S.A.M.G. news

Ian Fisk. Z80 opcodes. Auto CR. Various Software Programs

Rick Matthews. Original RXDAT and TXDAT Software.

Howard Harvey. That Auxiliary Carry Bit.

Bob Hourigan. NRZ cassette interface PC board.

Last month I had a visit from Mr Bill Bolton who is a member of MEGS (Microcomputer Enthusiasts Group) of Sydney. He was over here on business and looked us up from the newsletter. Bill made the suggestion that they were thinking of getting a national newsletter published and wanted to know what our opinions were of the idea. The basic concept was to have various clubs and groups sending the local news to a central point where the articles could be put together into one newsletter. If any members have any thoughts on this matter please let us know as the committee at this stage think its a good way to get alot of different ideas into one paper.

As most of you are aware February is the annual general meeting night and I wish to advise that nominations are open for all positions on the committee. Due to increasing pressure and the busy task of writing the newsletter I will not be available for nomination as Chairman.
The December meeting is on the 14th 7.30pm

The January meeting is on the 11th 7.30pm

RECEIVE DATA BY NIBBLES COMMENTS
AND SOURCE BY E. CLARKE.

The following source code has been compiled to form a program which will work self contained in ram at 4000H. The points to change if not using CP/M are the I/O calls e.g. line 66 to 68 outputs a string of characters pointed at by DE reg. You may replace that routine with your monitor call. There are other string output calls at line 124 to 126, 133 to 135, 144 to 146, 148 to 150 and 158 to 160.

Another call is made to character in and character out, suitable calls to your monitor should be patched into this program. If you have a slow VDU response the 256 byte count "*" output to your terminal may require the TXDAT program to run rather slowly. The routine to output the "*" is at line 98 just put in 3 no-ops to bypass that call.

Now for all the CP/M users who want to use this program I will set out the operating processes of the RXDAT module. Firstly you will notice 2 org statements in the source listing one at 0100H and one at 4000H. This has been done to allow the whole program to be a COM file yet the program runs at 4000H. The trick is to load the COM file and then excecute it, then as the first part of the program is a block move it moves the RXDAT module up to 4000H then promptly jumps to 4000H and excecutes the main program.

Why run the RXDAT program at the top of memory you say, why not just use it as a standard COM file and run it at 0100H. The reason is that the data recieved by this program is best stored from location 0100H upwards so as to use the SAVE function under CP/M to store the data recieved to disk giving it a file name.

While I am still on the subject of COM files to create the COM file from the listed source I found the best way was the following:

1)Assemble the source file with zero in the DEND EQU line 42.
2)Re-edit the source to put the correct end address.
3)Assemble the source again with updated file.
4)Call DDT RXDAT.HEX putting block move at 0100H and RXDAT at 4000H.
5)Use the DDT move data routine to shift the program at 4000H down to the tail end of block move at 011AH
6)Use control C to exit CP/M then SAVE X pages giving it a secondary filename of .COM

The next issue of the newsletter will contain the TXDAT source program, however I would like to see members have a go at writing a transmitter program, particularly for other machines eg: 6800 6502 2650 and etc.

```
1:
2:
3:                  ************************************************
4:                  *                                              *
5:                  *       RXDATA is a program to control         *
6:                  *       the reception of Bytes of data         *
7:                  *       in the form of Nybbles and store       *
8:                  *       this data away in a specified          *
9:                  *       area of ram.                           *
10:                 *                                              *
11:                 ************************************************
12:
13:
14:                 ************************************************
15:                 *       This version of PIO has open ended     *
16:                 *       file length the end of file is con     *
17:                 *       trolled by bit 6 of each byte tran     *
18:                 *       sfered. As with bit 7 a low to high    *
19:                 *       level change is valid strobe.          *
20:                 *       Source by E.S.Clarke 17/11/79          *
21:                 ************************************************
22:
23:
24:
25:
26:
27:  0001 =         PAGE    EQU     01H     ;1st page data value
28:  000D =         cr      EQU     0DH     ;Carriage return
29:  000A =         lf      EQU     0AH     ;Line feed
30:  0002 =         TYPEF   EQU     2       ;Output to console funtion
31:  0001 =         CONS    EQU     1       ;Input character function
32:  003F =         SIZE    EQU     3FH     ;High order address of ram available
33:  0000 =         WARM    EQU     0000H   ;Warm boot to CP/M
34:  0005 =         BDOS    EQU     5       ;DOS entry point
35:  0009 =         PBUF    EQU     9       ;Print buffer function
36:  0083 =         INIT    EQU     083H    ;8255 Control word
37:  0043 =         CONTL   EQU     043H    ;8255 Port control address
38:  0100 =         START   EQU     0100H   ;1st location for start of data string
39:  0041 =         DATA    EQU     041H    ;8255 Port data address configured as an input
40:  011A =         SRCE    EQU     011AH   ;Block move start address
41:  4000 =         DEST    EQU     4000H   ;Block move destination
42:  419B =         DEND    EQU     419BH   ;Block move dest end address
43:                 *
44:                 *
45:  0100                   ORG     0100H   ;Com file start address
46:  0100 211A01    STRT    LXI     H,SRCE  ;Load HL with source start address
47:  0103 110040            LXI     D,DEST  ;Load DE with destination address
48:  0106 019B41            LXI     B,DEND  ;Load BC with destin end address
49:                 *
50:  0109 7E        LOOP    MOV     A,M     ;Put byte pointed by HL into A
51:  010A 23                INX     H       ;Increment HL reg
52:  010B 12                STAX    D       ;Store byte pointed by DE reg
53:  010C 7A                MOV     A,D     ;Put high byte of pointer in A
54:  010D B8                CMP     B       ;Compare if = to End high byte
55:  010E C21601            JNZ     CONT    ;If not equals forget low byte test
56:  0111 7B                MOV     A,E     ;Put low byte of pointer in A
57:  0112 B9                CMP     C       ;Compare if = to End low byte
58:  0113 CA0040            JZ      BEGIN   ;If end of block move go "Begin"
59:                 *
60:  0116 13        CONT    INX     D       ;Increment destination pointer
61: *
```

```
61:   0117 C30901          JMP     LOOP    ;Go get and put another byte
62:                *
63:                *
64:   4000                 ORG     4000H   ;Program starting address.
65:   4000 3E01    BEGIN   MVI     A,PAGE  ;Load A first page value
66:   4002 320141          STA     PSTOR   ;Store value temp store
67:   4005 110541          LXI     D,MSG1  ;Get message"Waiting for data"
68:   4008 0E09            MVI     C,PBUF  ;Load code for BDOS control
69:   400A CD0500          CALL    BDOS    ;Go BDOS output string routine
70:   400D 0680            MVI     B,80H   ;Set bit 7 to high for string control
71:   400F 3E00            MVI     A,00H   ;Clear accum. ready for clear bit 6
72:   4011 320241          STA     BIT6    ;Clears bit 6 temp store
73:                *
74:   4014 3E83    RXDATA  MVI     A,INIT  ;Get control word for port
75:   4016 D343            OUT     CONTL   ;Put word to port this configures port as input
76:   4018 210001          LXI     H,START ;Get data area start address
77:   401B 0E80            MVI     C,080H  ;Load strobe testing byte
78:                *
79:   401D CD4E40  RX1     CALL    RXNIBL  ;Go subroutine get nybble
80:   4020 5F              MOV     E,A     ;Put low order nybble to Reg E
81:   4021 CD4E40          CALL    RXNIBL  ;Go subroutine get 2nd nybble
82:   4024 07              RLC             ;Rotate nybble into high order
83:   4025 07              RLC             ;position in accumulator
84:   4026 07              RLC
85:   4027 07              RLC
86:   4028 B3              ORA     E       ;Combine low order nybble in Reg E
87:                                        ;with Reg A. Full data now in A.
88:   4029 77              MOV     M,A     ;Store data byte in A to location
89:                                        ;pointed to by H,L Reg
90:   402A 3E3F            MVI     A,SIZE  ;Get ram user ram size
91:   402C BC              CMP     H       ;Compare ram size with ram used
92:   402D CA7040          JZ      RAMFUL  ;If no more room go exit sub
93:   4030 3A0141          LDA     PSTOR   ;Get current page value
94:   4033 BC              CMP     H       ;Compare with data pointer
95:   4034 CA4040          JZ      BIT     ;If same go bit
96:   4037 3C              INR     A       ;Increment page value
97:   4038 320141          STA     PSTOR   ;Store page value
98:   403B 3E2A            MVI     A,'*'   ;Load an * into A
99:   403D CDF440          CALL    PCHAR   ;Print that *
100:  4040 3A0241  BIT     LDA     BIT6    ;Get bit 6 from temp store
101:  4043 E640            ANI     40H     ;Mask out bit 6
102:  4045 FE40            CPI     40H     ;Check if bit 6 set
103:  4047 CA8540          JZ      FINSH   ;If bit 6 set go to FINSH sub
104:  404A 23              INX     H       ;Increment pointer
105:  404B C31D40          JMP     RX1     ;Lets go get another byte
106:               *
107:               *
108:  404E DB41    RXNIBL  IN      DATA    ;Get data from input port
109:  4050 320241          STA     BIT6    ;Store for later use in program ending
110:  4053 2F              CMA             ;Complement data in A reg
111:  4054 E680            ANI     080H    ;Mask off 8th bit
112:  4056 B9              CMP     C       ;Compare contents of C reg with A reg
113:  4057 4F              MOV     C,A     ;Update C reg with previous A reg contents
114:  4058 D24E40          JNC     RXNIBL  ;If compare instruction did not invoke
115:                                       ;a carry bit go RXNIBL
116:  405B 78              MOV     A,B     ;Get string control bit in A reg
117:  405C E680            ANI     80H     ;Test it for a 1
118:  405E 17              RAL             ;Shift it to the carry
119:  405F DC6740          CC      MESGPR  ;If carry = 1 go string print
120:  4062 DB41            IN      DATA    ;Get data this time to get the nybble
121: *
```

```
121:    4064 E60F           ANI     0FH     ;Mask off high order bits
122:    4066 C9             RET             ;Return
123:                 *
124:    4067 3E2A   MESGPR  MVI     A,'*'   ;Load asterisk to put to screen
125:    4069 CDF440         CALL    PCHAR   ;Print character
126:    406C 78             MOV     A,B     ;Get MESGPR control bit
127:    406D AF             XRA     A       ;Clear accumulator
128:    406E 47             MOV     B,A     ;Put cleared bit back
129:    406F C9             RET
130:                 *
131:    4070 111F41 RAMFUL  LXI     D,MSG3  ;Load ramful message pointer
132:    4073 0E09           MVI     C,PBUF  ;String print BDOS code
133:    4075 CD0500         CALL    BDOS    ;Go print string
134:    4078 0E01           MVI     C,CONS  ;Input character BDOS code
135:    407A CD0500         CALL    BDOS    ;Go input string
136:    407D FE59           CPI     059H    ;Check character for "Y" input
137:    407F CA0040         JZ      BEGIN   ;If input = "Y" then have another go
138:    4082 C30000         JMP     WARM    ;If any other key go to CP/M
139:                 *
140:                 *
141:    4085 220341 FINSH   SHLD    CHPTR   ;Store HL contents, end of data address
142:    4088 117041         LXI     D,MSG4  ;Load End message string pointer
143:    408B 0E09           MVI     C,PBUF  ;Print string BDOS pointer
144:    408D CD0500         CALL    BDOS    ;Go print string
145:    4090 CDC840         CALL    HLHEX   ;Go HL contents hex print out
146:    4093 118641         LXI     D,MSG5  ;Get msg '(Save $' pointer
147:    4096 0E09           MVI     C,PBUF  ;String print BDOS code
148:    4098 CD0500         CALL    BDOS    ;Go print string
149:    409B 2A0341         LHLD    CHPTR   ;Get data pointer into HL
150:    409E 0E64           MVI     C,100   ;Put 100 decimal into C reg
151:    40A0 CDB740         CALL    CONDEC  ;Go Condec subroutine
152:    40A3 0E0A           MVI     C,10    ;Put 10 decimal into C reg
153:    40A5 CDB740         CALL    CONDEC  ;Go Condec sub again
154:    40A8 7C             MOV     A,H     ;Get high byte into A
155:    40A9 CDC540         CALL    CD2     ;
156:    40AC 118F41         LXI     D,MSG6  ;Get msg '  Pages)' pointer
157:    40AF 0E09           MVI     C,PBUF  ;String print BDOS code
158:    40B1 CD0500         CALL    BDOS    ;Go print string
159:    40B4 C30000         JMP     WARM    ;Go warm start to CP/M
160:                 *
161:    40B7 7C     CONDEC  MOV     A,H     ;Get high byte into A
162:    40B8 1E00           MVI     E,0     ;Clear E reg
163:                 *
164:    40BA 91     CD0     SUB     C       ;Subtract C contents from A
165:    40BB DAC240         JC      CD1     ;Go CD1 if a carry
166:    40BE 1C             INR     E       ;Increment the E reg
167:    40BF C3BA40         JMP     CD0     ;Go to CD0
168:                 *
169:    40C2 81     CD1     ADD     C       ;Add C contents to A
170:    40C3 67             MOV     H,A     ;Restore H reg
171:    40C4 7B             MOV     A,E     ;Put decimal count into
172:                 *
173:    40C5 C3E240 CD2     JMP     PNIB    ;Go print nibble
174:                 *
175:    40C8 3A0441 HLHEX   LDA     CHPTR+1 ;Load accumulator high order byte
176:    40CB CDD540         CALL    PHEX    ;Go print hex sub
177:    40CE 3A0341         LDA     CHPTR   ;Load accumulator low order byte
178:    40D1 CDD540         CALL    PHEX    ;Go print hex sub
179:    40D4 C9             RET
180:                 *
181: *
```

```
                        *
40D5 F5         PHEX    PUSH    PSW     ;Save, print hex char in A sub
40D6 0F                 RRC             ;Move high order nybble into low order pos
40D7 0F                 RRC
40D8 0F                 RRC
40D9 0F                 RRC
40DA CDE240             CALL    PNIB    ;Go print nibble subroutine
40DD F1                 POP     PSW     ;Get byte for low order nibble
40DE CDE240             CALL    PNIB    ;Go print 2nd nibble
40E1 C9                 RET
                        *
                        *
40E2 E60F       PNIB    ANI     0FH     ;Mask low 4 bits, Print nibble in reg A
40E4 FE0A               CPI     10      ;Is it larger than 9
40E6 D2EE40             JNC     P10     ;If yes go P10
40E9 C630               ADI     '0'     ;If not adjust it for number
40EB C3F040             JMP     PRN     ;Go print character
                        *
40EE C637       P10     ADI     'A' - 10        ;Adjust it for character
40F0 CDF440     PRN     CALL    PCHAR   ;Go print a character
40F3 C9                 RET
                        *
40F4 E5D5C5     PCHAR   PUSH H! PUSH D! PUSH B! ;Save regs, Print a character
40F7 0E02               MVI     C,TYPEF ;Character to console BDOS code
40F9 5F                 MOV     E,A     ;Move character into BDOS reg
40FA CD0500             CALL    BDOS    ;Go print character
40FD C1D1E1             POP B! POP D! POP H     ;Restore regs
4100 C9                 RET
                        *
                        *
4101            PSTOR   DS      1       ;Temp store page value
4102            BIT6    DS      1       ;When bit 6 is set store here
                        *
4103            CHPTR   DS      2       ;Store for HL character pointer
                        *
4105 5741495449MSG1     DB      'waiting for data please'
411C 0A0D24             DB      0AH,0DH,'$'
                        *
                        *
411F 4441544120MSG3     DB      'Data size exceeds ramsize',cr,lf,0
413B 544F205452         DB      'To try again input "Y" ',cr,lf,0
4155 4F5220414E         DB      'Or any other key to exit'
416D 0A0D24             DB      0AH,0DH,'$'
                        *
4170 0D0A444154MSG4     DB      cr,lf,'Data loaded - 0100-$'
                        *
4186 2820534156MSG5     DB      '( Save  $'
                        *
418F 2020504147MSG6     DB      '  Pages )',cr,lf,'$'
419B                    END
   : *
```

INSPIRED BY ERIC CLARKE'S NRZI CASSETTE INTERFACE MODIFICATION IN THE JUNE/JULY ISSUE OF THE CLUB MAGAZINE, I HAVE DESIGNED A P.C. BOARD AS A CLUB PROJECT.
THE BOARD IS BASICALLY THE SAME AS ERIC'S DESIGN WITH A FEW MINOR MODIFICATIONS IN PIN ALLOCATIONS AND CAPACITOR TYPES. THESE MODS HAVE BEEN DONE FOR EASE OF BOARD LAYOUT AND COMPONENT AVAILABILITY.

THE P.C. BOARD HAS PROVISION FOR TWO (2) CASSETTE INTERFACES WITH OUTPUT BUFFERING, DATA PRESENT LEDS, A +5 VOLT POWER SUPPLY, A +- 12 VOLT POWER SUPPLY, A 2 CHANNEL RS232 INTERFACE AND A KLUGE AREA FOR YOUR INTERFACE TYPE, EG 20MA.

THERE ARE ONLY FOUR JUMPERS ON THE BOARD WHICH ARE ONLY FOR THE INPUT/OUTPUT INTER-CONNECTS, THUS ENABLING YOU TO USE WHICH EVER INTERFACE YOU DESIRE WITHOUT CUTTING TRACKS.

THE BOARD MEASURES APPROX 5.5 INCHES * 5.25 INCHES.
IF YOU DO NOT REQUIRE THE POWER SUPPLIES & INTERFACES THE BOARD CAN BE TRIMMED BACK TO APPROX 5.5 * 2.75 WHICH SHOULD FIT UNDER MOST PORTABLE CASSETTES ON THE MARKET.

THE BOARDS WILL BE SUPPLIED WITH CIRCUIT DIAGRAMS AND COMPONENT OVERLAYS AT A COST OF.....$ 6.00

BOARDS SHOULD BE ORDERED THROUGH THE CLUB, AND A LIMITED NUMBER WILL BE AVALIABLE AT THE DECEMBER MEETING.

I AM RUNNING THE INTERFACE AT 1200 BAUD, AND I HAVE BEEN USING IT FOR APPROXIMATELY 2 MONTHS NOW WITHOUT A BAD LOAD IN THAT PERIOD.
I RECOMMEND THIS METHOD OF DATA STORAGE FOR ANY ONE WITH OUT THE FUNDS FOR A DISK DRIVE. THE ALL UP COST OF THE PROJECT SHOULD BE ABOUT $20.00 EXCLUDING THE COST OF THE CASSETTE RECORDER.

BOB HOURIGAN

THE MICRO SHOP

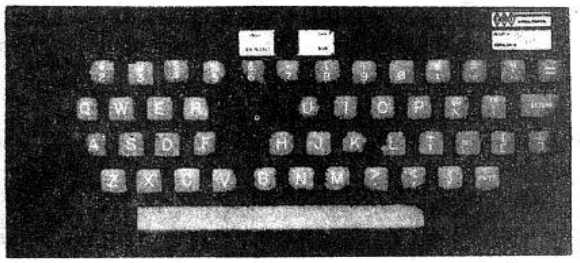
MICROCOMPUTER SERVICES,
COMPONENTS, KITS,
AND ASSEMBLIES

MAIL ORDERS:–
BOX 207, GAWLER
SOUTH AUSTRALIA 5118

ANNOUNCING! Full ASCII PROFESSIONAL Keyboard Kit!

FROM GEORGE RISK INDUSTRIES . . . "The Leader in Microcomputer Keyboards" . . comes the new Model 756 FULL ASCII Keyboard, designed especially for use with all Microcomputer systems!

NEW!



LOW COST!

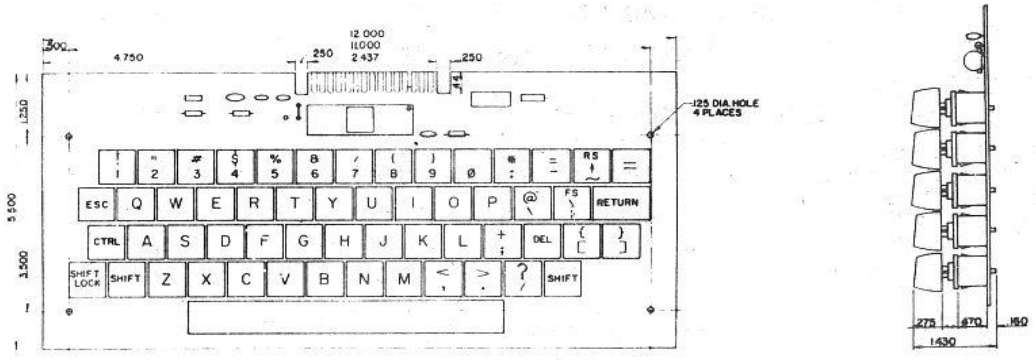
756 KEYBOARD APPLICATIONS:
- ☆ MICROCOMPUTER DATA INPUT
- ☆ SMART TERMINALS
- ☆ VIDEO DISPLAY ☆ GRAPHICS
- ☆ TERMINALS ☆ TVT'S
- ☆ MORSE/RTTY/SSTV KEYBOARDS
- ☆ CONTROL SYSTEMS

.and anywhere you need low cost, reliable ASCII!

- ☆ EASY INTERFACE WITH:
  MITS, IMSAI, SWTPC, Digital Group, Processor Technology, Cromemco, Polymorphic Systems, TDL, KIM,, VAB-2 MiniTerm, ECT, OSI, Vector Graphic, E & L, Cosmac VIP, Byte, and any other system with an 8 bit parallel input.

CHECK THESE PROFESSIONAL FEATURES!

- ☆ Full 128 Character ASCII!
- ☆ Tri-Mode MOS Encoding!
- ☆ MOS/DTL/TTL Compatable Output!
- ☆ Two-key Rollover!
- ☆ Level and Pulse Strobe!
- ☆ Shift and Alpha Lock!
- ☆ Selectable Parity!
- ☆ Positive or Negative Logic!
- ☆ All New, OEM Grade Components!
- ☆ Gold Contact, Low Bounce Keyswitches!
- ☆ Rugged G-10 Printed Circuit Board!
- ☆ Custom 2-Shot Molded Keycaps!
- ☆ Low Power Consumption!
- ☆ Optional Numeric Pad Available!
- ☆ Custom Enclosures Available!
- ☆ More!



The G.R.I. Model 756 Full ASCII Keyboard is designed to better meet the needs of Personal, Industrial, and Business microcomputer users. The 756 provides encoding for all 128 ASCII characters and control functions, imposing no limitations on software design, or hardware capability. You may expect years of reliable performance with the combination of dependable low power MOS encoding and our proven KBM series O.E.M. keyswitches. Like all our products, the 756 uses only tested, prime industrial quality components, and a rugged MIL-grade printed circuit board for easy assembly and mounting.

The 756 interface allows user selection of positive or negative logic data and strobe output, selectable parity, and offers both d.c. level and pulse strobe signals for no-modification, "hassle-free" connection to any 8 bit parallel input port, video display keyboard input, or terminal board. Both upper and lower case characters are generated by the keyboard, with latching-type Shift-Lock and provision for alpha-lock (upper-case only) mode for operator convenience.

The model 756 bridges the gap between basic keyboards like our Model 753 teletype array, and expensive, custom O.E.M. keyboards. The 756's simple electronic elegance is complemented by a line of accessories, including a numeric pad, custom cables and connectors, and deluxe heavy gauge all-steel enclosures styled to match contemporary systems. The 756 is available in either kit or assembled and tested form, and is waranteed for 90 days. Typical assembly time is less than two hours, and full documentation is included.

Microcomputer uses change, but chances are, the Model 756 is all the keyboard you'll ever need.

THE MICRO SHOP

MICROCOMPUTER SERVICES,
COMPONENTS, KITS,
AND ASSEMBLIES

S-100 SPECIALISTS

MAIL ORDERS:—
BOX 207, GAWLER
SOUTH AUSTRALIA 5118

TELEPHONE: (085) 22 3955

TELEX: MICROA AA 89094

COMPONENT & KIT CORNER

TRS-80 SPECIALS

16K MEMORY EXPANSION KITS, WITH JUMPERS & INSTRUCTIONS $110
HUH TRS-80/S100 INTERFACE KIT $125. TRS-80/S100 BUS EXPANDER (4-SLOT) $150

KEYBOARD/SCT-100 SOCKETS

Gold $4.50

S-100 SOCKETS

Gold $8 Tin $5.75

POWER SUPPLY COMPONENTS

TR-10 S-100 Transformer $45.
25A Bridge rectifier $6.

MUFFIN FANS, 240V $25
100,000 mfd. 15V $24. 55,000 mfd. 25V $13.
22,000 MFD. 25V $15. 68,000 MFD. 16V $22

CHIPS ETC.

KR-2376 Keyboard encoder $15. 2708 EPROM $15/$12. 50 TTL mixed $6.
Sub-miniature quartz clock crystals 8MHz, 18MHZ $8.

BARE BOARDS

Special Seals 8K static RAM boards (use 2102) $30. Vector 8802 $36
Vector 8800 Wire-wrap $35. Vector 8804 Wire-wrap $38. Extender board $45.
Full range of wire-wrap tools and kits coming in for Christmas.

KITS

SBC-100 Z-80 single board computer $280. SD Sales Z-80 Starter Kit $280
Ithaca/Jade Z-80 CPU $150. SD Versafloppy Floppy Disk Controller $185.
SD SALES EXPANDOPROM KIT $145. HURRY! SD HAVE ANNOUNCED 50% PRICE RISES!

Thinker Toys 32K Static 4MHz $711. Thinker Toys 16K Static 4MHz $340.
Thinker Toys Switchboard $235. Thinker Toys 8-slot Wunderbuss $82.
Xitex SCT-100 VDU, micro controlled stand-alone, 16 x 64 upper and lower case characters, dual baud rates, ASCII or baudot code. $169.

FLOPPY DISKS

MPI Mini (8mS step time) $349. Thinker Toys Discus 2 assembled & tested, with 1 Shugart 8" drive in cabinet with power supply, cables, standard software, single/double density, 50Hz version $1429. Additional Shugart drive in cabinet with 50Hz power supply $856. Bare Shugart 801R $750.

KEYBOARDS

G.R.I. Model 756 VDU layout $70. +5/-12V on-card converter $12. Plastic case $20.

ALL PRICES INCLUDE SALES TAX - APPLY FOR STUDENT EXEMPTION RATES