Richard Brown
(213) 597-2955
Jose Ramirez
145 E. 53rd St.
Long Beach, Ca. 90805
(213) 428-1758

The next meeting of the S.C.S.U.G. and other computer users will be held at Walt Hendrickson's house located at 5008 Range Horse Lane, Rolling Hills Estates, on Oct. 8, 1987 at 8:00 p.m. The tentative agenda for the October meeting will be (1)The objectives of the expanded group (2)Sorcerer discussion.

At the last meeting in September, the group had a guest from Texaco at New Orleans, Ron Phair. We talked about the problems they have at Texaco with their Geophysical and Geologic data bases. Consequently; as it happens with our tentative agenda on occasion, we altered it, and very little was done on the expaded objectives.

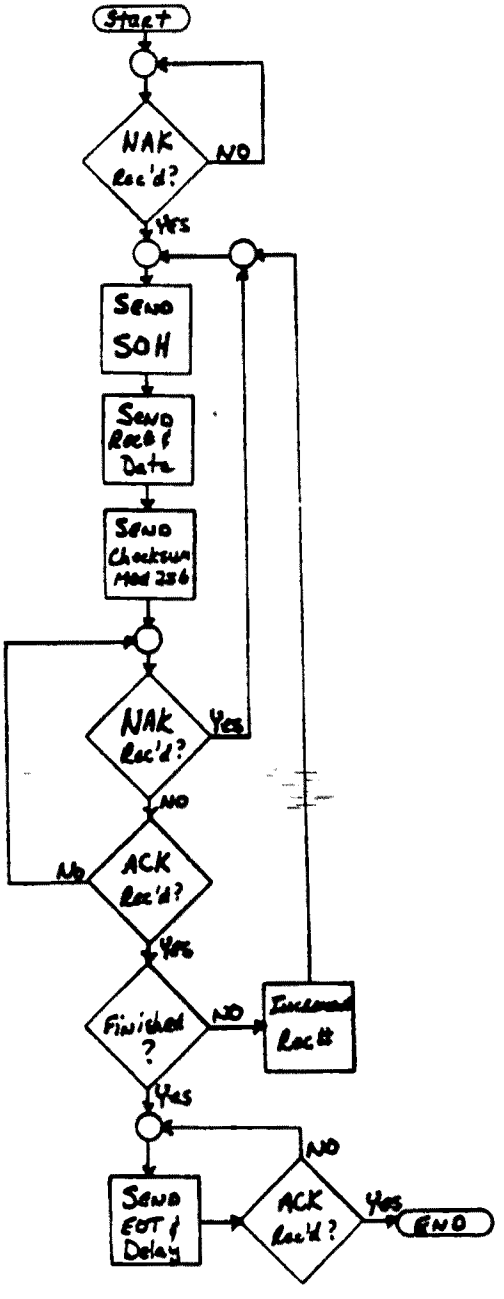
There is some good news on the bulletin board, as all of you know Eric Matlen has been dilligently working on the BB for some time. This week he requested that I be the first caller to the Southern California Sorcerer Users BB. Although, I was the first caller Eric did find a bug in the system and I crashed on it. But Eric is very confident that the system is now looming in the doorway. So for you users that are out of state or in far of places, we can communicate with each other very soon, via the BB's message board.

The addendums to the newsletter in this month and for the next two months were supplied by Dennis Perkins. I asked him for any routines, that he had, utilizing the I/O capabilities of the Sorcerer. And he sent me a few that he had, along with a flow-chart using Chistiensen's protocol. I hope that you folks will find them useful, sice that is the kind of material that we asked for at the meetings a few months ago.

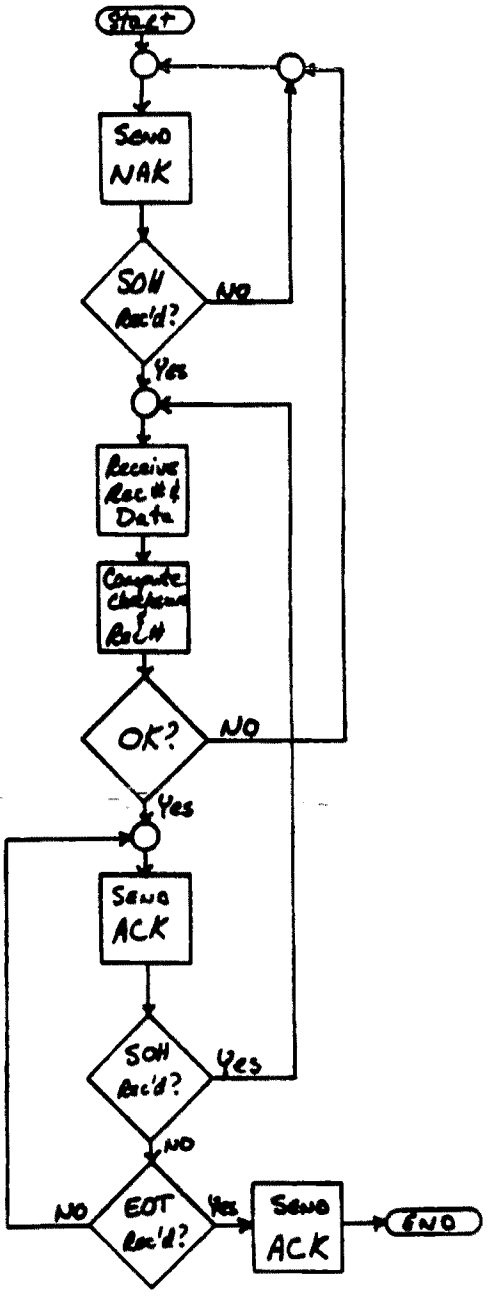
The swap meets locally have had some real good buys lately. Something we all use 5 1/4 diskettes are going for as little as $3.00 per box of ten double sided, double density diskettes. So if you can go to the swaps, it a good place to get good buys, especially the TRW or the Advanced swap meets. As most of you know admission is free and so is the parking. And there is now a free swap in Highland Park for those of you in L.A. proper.

Christensen XMODEM Protocol ----

Transmit:
Waits for NAK
Byte 1 Sends SOH;
" 2 Sends Rec# Mod 256 (natural 1-n)
" 3 1's complement of Rec#
" 4-131 128 data bytes
" 132 Checksum Mod 256. Waits
for ACK or NAK:
NAK - retransmit block
ACK - increments & transmits
next block
Repeats until all blocks
gone & ACK'd
Sends EOT every second
until ACK'd



Receive:
Sends NAK, waits
for SOH
Gets whole packet
Checks Rec#
Checks checksum of
128 byte date
Sends ACK or NAK:
NAK - failure
ACK - pass
Waits for EOT or SOH:
SOH - Repeats packet
receive
EOT - Sends ACK



```
;
;
;       INIT:   this routine clears out the buffer from 0000 to BC00;
;               tests the buffer to make sure the RAM is OK, then fills
;               with FF's...
;
;
        GLOBAL  INIT
;
INIT:   LD      HL,CHECK                ;say we're checking...
        CALL    STRPRT
;
        LD      A,00H                   ;set up with all zero's
        LD      HL,STRT                 ;set up the start
        LD      BC,ND                   ; then the end
;
CNT:    CALL    CHK                     ;check the buffer byte
        CALL    NDCK                    ; then for the last one
;
        JP      Z,INTCNT                ; if done, continue to next check
        INC     HL                      ; if not, bump HL...
        LD      A,0                     ; set up A again
        JP      CNT                     ; and start over
;
INTCNT: LD      HL,LDNG                 ;now say we're loading...
        CALL    STRPRT
;
        LD      HL,STRT                 ; set up HL with the start again
        LD      A,0FFH                  ; then set all the bits
;
CNT1:   CALL    CHK                     ; check the byte
        CALL    NDCK                    ; then check for the end
;
        JP      Z,FINIS                 ;if done, finish up
;
        INC     HL                      ;else bump HL
        LD      A,0FFH                  ; set all the bits again
        JP      CNT1                    ;  then jump back
;
FINIS:  RET                             ;leave if the buffer is filled with  FF's
;
;
;
CHK:    LD      (HL),A
        CP      (HL)                    ;compare it with the pointer contents
        RET     Z                       ; leave if ok
;
        PUSH    HL                      ;else save it
        LD      HL,ERRMSG               ; then put up the error message
        CALL    STRPRT
        POP     HL                      ; then get it back
;
        RET
;
;
NDCK:   LD      A,H                     ;get H in A
```

```
        CP      B                       ; compare to B (upper half of ND)
        RET     NZ                      ;  leave if not there
        LD      A,L                     ;then L in A
        CP      C                       ; check with C(lower half of ND)
        RET                             ;leave anyway
;
;
CLR     EQU     0CH
CR      EQU     0DH
STRPRT  EQU     0E051H
NUMPRT  EQU     0E1E8H
PRT     EQU     0E01BH
STRT    EQU     0000H
ND      EQU     08000H
;
CHECK   DB      CLR,'Setting up 32k buffer...'
        DB      CR,'Checking Buffer...',CR,0
LDNG    DB      CR,'Loading Buffer...',CR,0
ERRMSG  DB      CR,' **** Memory Error in Buffer... ****'
        DB      CR,' ****    halt and correct...    ****',0
;
;
        END
;
;
```

```
;
;
;       EXCK: checks keyboard for pause or exit keys;
;
        GLOBAL  EXCK
;
EXCK:   PUSH    HL                      ;save the registers
        PUSH    DE
        PUSH    BC
;
        CALL    KPRSS                   ;keypressed?
        JP      Z,NXCHR                 ;if not, continue
;
        CALL    KYGET                   ;if so, get it
;
        CP      CONTC                   ;^C?
        JP      NZ,KYCK
;
        LD      A,0FFH                  ;if so, set the flag
        POP     BC                      ;get back the registers
        POP     DE
        POP     HL
;
        RET                             ; then leave
;
KYCK:   CP      RS                      ;runstop or ^S?
        JP      NZ,NXCHR
;
KLOOP:  CALL    KPRSS                   ;if so, pause
        JP      Z,KLOOP
;
NXCHR:  CP      CONTP                   ;a ^P?
        JP      NZ,NXCHR1
;
        LD      A,(STOR)                ;if so, check the printer flag...
        CP      0FFH
        JP      NZ,PRNT
;
        LD      (IY+03FH),01BH          ;if set,
        LD      (IY+040H),0E0H          ; route the output to video only
;
        LD      A,0
        LD      (STOR),A                ;then reset the flag
;
        LD      HL,OFF                  ;tell 'em it's off
        LD      DE,POS
        LD      BC,3
        LDIR
;
        JP      NXCHR1
;
PRNT:   LD      (IY+03FH),093H          ;if not set...
        LD      (IY+040H),0E9H          ; send to printer, too
;
        LD      A,0FFH                  ; and set the flag
```

```
        LD      (STOR),A
;
        LD      HL,ON                   ;tell 'em it's on
        LD      DE,POS
        LD      BC,3
        LDIR
;
NXCHR1: POP     BC                      ;get the registers back
        POP     DE
        POP     HL
;
        LD      A,0                     ;set up the flag for no exit
        RET
;
;
;t-t-t-that's all, folks!!!
;
;
;               the equate table...
;
CR      EQU     0DH
LF      EQU     0AH
ULIN    EQU     05FH
RUB     EQU     07FH
BS      EQU     08H
PERIOD  EQU     02EH
QUEST   EQU     03FH
CONTC   EQU     03H
CONTP   EQU     010H
RS      EQU     013H
SPACE   EQU     020H
;
STRBUF  EQU     0BC00H
STRT    EQU     STRBUF+5
ND      EQU     STRT+5
STOR    EQU     ND+5
POS     EQU     0F2DDH
;
STRPRT  EQU     0E051H
CNVRTX  EQU     0E23DH
NUMPRT  EQU     0E1E8H
BYTPRT  EQU     0E1EDH
PRT     EQU     0E00CH
KPRSS   EQU     0E03FH
KYGET   EQU     0E018H
;
ON      DB      ' ON'
OFF     DB      'OFF'
;
        END
;
;
```

```
;
;
;       RECBUF.ASM:     a routine that uses the Christensen Xmodem protocol
;                       for receiving a file from another computer...prompts
;                       for start address for placing the file into the
;                       buffer...
;
;
        GLOBAL  RECBUF
        EXTERN  STRIN
        EXTERN  CONVRT
;
RECBUF: LD      HL,STRTST               ;we've begun
        CALL    STRPRT
;
        LD      B,4                     ;get the buffer load address
        CALL    STRIN
        CALL    CONVRT                  ; convert it
;
        LD      (STRT),DE               ;save it
;
        LD      A,1                     ;this is the first record number
        LD      (RECNM),A
;
        LD      HL,WARN                 ;give 'em a way out
        CALL    STRPRT
;
RLP:    CALL    KPRSS                   ;check for a keypress
        JP      Z,RLP
;
        CALL    KYGET                   ;get it
        CP      X                       ; an X?
        RET     Z                       ;  then leave
;
        LD      HL,NAKSTR               ;tell 'em we're initializing
        CALL    STRPRT
;
        LD      HL,EXPSTR               ; tell 'em we're expecting something
        CALL    STRPRT
;
NAKST:  LD      A,(RECNM)               ; put it up
        CALL    BYTPRT
;
        LD      HL,BS                   ;get ready for the next one
        CALL    STRPRT
;
NAKLP:  LD      A,NAK                   ;initialize
        CALL    XMIT
;
        CP      SOH                     ;ready?
        JP      Z,PACREC                ;  then get the rest of the packet
;
        CP      CAN                     ;cancelled?
        JP      NZ,NAKLP                ; if not, try again
;
```

```
        LD      HL,CANMSG       ;  if so, tell 'em
        CALL    STRPRT
;
        RET                     ;then leave
;
PACREC: LD      B,083H          ;set up to get some bytes
        LD      HL,RCBF
;
PACLP:  CALL    SERIN           ;get one
        LD      (HL),A          ;save it
        INC     HL              ;bump the counter
        DJNZ    PACLP           ;loop back
;
        LD      A,(ONESCP)      ;check the record number
        LD      HL,RCBF         ;get the record number in HL
;
        CPL                     ; complement the complement
        CP      (HL)            ;see if they're the same
;
        JP      NZ,NAKST        ;if not, tell 'em to retransmit
;
        LD      HL,RECNM        ;get what we're expecting
        CP      (HL)            ; see if they're the same
;
        JP      NZ,NAKST        ;if not, try again
;
CKCK:   LD      A,0             ;check the checksum
        LD      HL,DBF
        LD      B,080H
CKLP:   ADD     A,(HL)          ;get what there really is
        INC     HL
        DJNZ    CKLP
;
        LD      HL,CKSM         ;then compare with what they say
        CP      (HL)
;
        JP      NZ,NAKST        ;if bad, ask for it again
;
        LD      HL,DBF          ; if all the above is OK
        LD      DE,(STRT)       ;   save it in the working buffer
        LD      BC,080H
        LDIR
;
        LD      (STRT),DE       ; then save the next block address
;
        LD      A,(RECNM)       ;  & then increment the expected block #
        INC     A
        LD      (RECNM),A       ;then save it
;
        CALL    BYTPRT          ;and put it up
;
        LD      HL,BS           ;get ready for the next one
        CALL    STRPRT
;
ACKSND: LD      A,ACK           ;send an ACK
```

```
        CALL    XMIT
;
        CP      SOH                     ;see if we got an SOH back
        JP      Z,PACREC                ;if so, start again
;
        CP      EOT                     ;EOT?
        JP      NZ,ACKSND               ;if not, try another ACK
;
        LD      A,ACK                   ;if so, ACK it...
        CALL    SNDCH
;
        LD      HL,CMPL                 ;say we're done...
        CALL    STRPRT
;
        RET                             ;then leave
;
;
;
REC1:   IN      A,(0FDH)                ;doesn't wait for character
        BIT     1,A
;
        JP      Z,RCND
;
        CALL    SERIN
        RET
;
RCND:   LD      A,0FFH
        RET
;
;
;
XMIT:   CALL    SNDCH                           ;send what's in A
;
        LD      HL,05555H                        ;set up for loop
;
XLP:    CALL    REC1                            ;call the one that doesn't wait
;
        CP      0FFH                            ;a character?
        RET     NZ                              ; then leave
;
XCNT:   DEC     HL                              ;if not, drop the counter
        LD      A,H                             ; & check for 0...
        CP      0
;
        JP      NZ,XLP
;
        LD      A,L
        CP      0
;
        JP      NZ,XLP
;
        RET
;
;
;
```

```
;       the equate table...
;
;
SOH     EQU     01
EOT     EQU     04
ACK     EQU     06
CLR     EQU     0CH
CR      EQU     0DH
NAK     EQU     015H
X       EQU     'X'
CAN     EQU     018H
;
STRPRT  EQU     0E051H
PRT     EQU     0E00CH
BYTPRT  EQU     0E1EDH
KPRSS   EQU     0E03FH
KYGET   EQU     0E018H
SNDCH   EQU     0E012H
SERIN   EQU     0E00FH
;
STRBUF  EQU     0BC00H
STRT    EQU     STRBUF+5
ND      EQU     STRT+2
STOR    EQU     ND+2
COUNT   EQU     STOR+1
RECNM   EQU     COUNT+2
LSTREC  EQU     RECNM+1
RCBF    EQU     LSTREC+1
ONESCP  EQU     RCBF+1
DBF     EQU     ONESCP+1
CKSM    EQU     DBF+080H
;
BS      DB      1,1,0
STRTST  DB      CR,CR,'  Receive file from RS-232...'
        DB      CR,' Address to start file-----> ',0
WARN    DB      CR,CR,'  Press any key to initiate or X to eXit...',0
NAKSTR  DB      CR,CR,' ****** Initializing ******',0
EXPSTR  DB      CR,'      --------> Awaiting record # ',0
CANMSG  DB      CR,CR,' **** Routine cancelled by request ****',0
CMPL    DB      CR,CR,'  Transfer Completed',0
;
        END
;
;
;
```