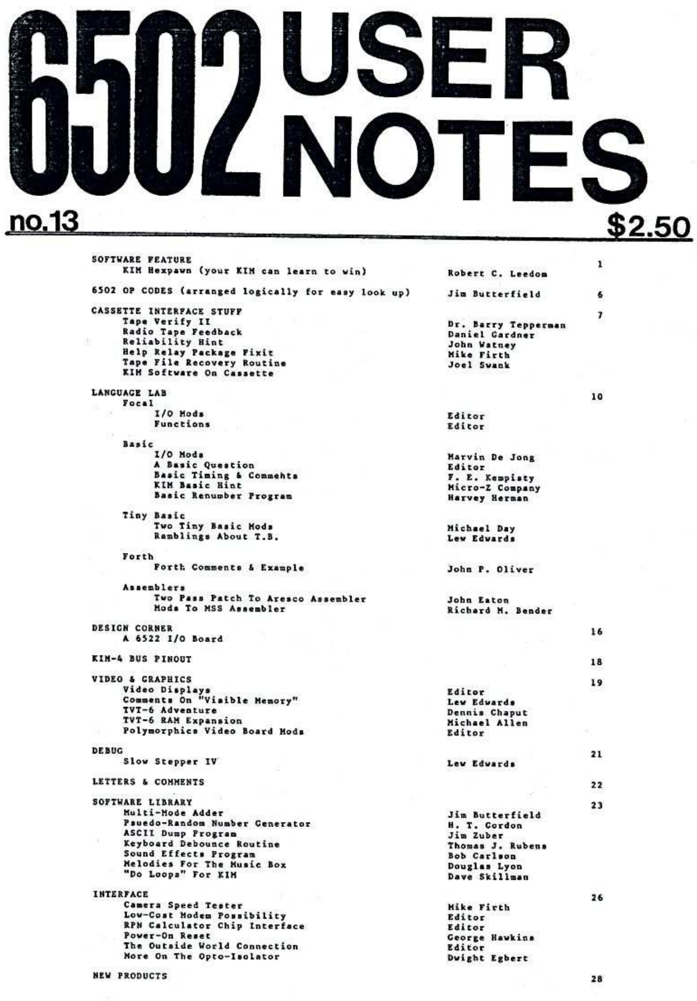
INCOMPLETE!!



6502 USER NOTES

no.13 $2.50

EDITORIAL

As you can tell already, we're back to using our old title. Although "USER NOTES: 6502 seemed like a good idea at first, old ties are hard to break - back to 6502 USER NOTES. It's easier to say anyway.

Lots of new things have been happening with the 6502 - many more are in store. The software situation has certainly gotten better - but there's still alot of room for improvement.

One problem that has slowed software development a bit is the fact that there have been no hobby mainframe systems (such as Southwest Techs 6800 machine and the IMSAI 8080 system) designed specifically for the 6502 to reach any level of popularity with aftermarket accessory manufacturers (which is a very good indication of marketplace acceptance).

By the way, I define "mainframe" as a backplane (motherboard) and a power supply in a box without an integral CPU.

Most 6502 hardware developers have gone their separate ways with regards to expansion capability. Witness the fact that there are now at least 6 bus oriented 6502 expansion systems which aren't the least bit compatible with each other.

Everybody loses in this situation. The hobbyist loses because since he will end up being locked into whatever system he purchases, he has to be sure that particular system has,(or will have) everything he has decided he needs (or will need). A very difficult decision to make for someone just getting into this hobby. One that could drive some folks away from the 6502 CPU altogether.

The manufacturer loses because with so many different 6502 expansion methods available, no self-respecting aftermarket supplier of boards would think of entering into such a diluted market. He would most likely go to the S-100 (IMSAI) or S-50 (SWTP) marketplace because of the numbers involved, the proliferation of software, and psuedo-standarization of hardware in those markets.

At this point, there is only one expansion bus which is being supported by aftermarket suppliers. That's the S-44 KIMbus from MOS Technology.

There are 6 companies (including MOS) supporting this bus in the form of accessory boards. That number is sure to increase since Synertek and Rockwell machines will also be using the S-44 KIMbus.

The S-44 KIMbus seems to hold the only real hope of popularizing the 6502 CPU and providing the consumer with an "intelligent" alternative to the S-100 bus and multiple sources of accessory boards.

When more than one company supports a particular bus in the form of accessory boards - everyone wins. The consumer now has the ability to shop around and look for the best deal on a particular board he has in mind. The supplier wins because as the market gets larger and broader - in its appeal, more consumers will enter into it and, as a result, more dollars will have a chance to reach him.

It will be interesting to watch how things develop in this marketplace.

6502 USER NOTES is published bimonthly (whenever possible) by Eric C. Rehnke, POB 33093, North Royalton, Ohio 44133. Subscription rates for Volume 3 are $13.00 (US and Canada) and $19.00 elsewhere. No part of 6502 USER NOTES may be copied for commercial purposes without the express written permission of the publisher. Articles herein may be freely reprinted by club newsletters as long as proper publication credit is given and the publisher is provided with a copy of the publication.

I think you're gonna like our new format alot. We've organized that articles to make things easier to find and are retypingall the articles (except for some program listings) to make things more consistent. Let me know your opinion. What would you like to see in our newsletter? I really enjoy feedback and look forward to YOUR comments.

They certainly are some neat new 6502 based machines entering the marketplace. Of course, I'm referring to the SYM (formaly VIM) from Synertek, the AIM from Rockwell and the Challenger 1P from OSI.

Phil Johnson (Johnson Computer) brought two OSI Challengers over to my place for a little demo so I could get an idea of what OSI was doing lately. I must say that I was impressed with the amount of capability built-in to these machines for the price. For example, for $350 you can get a machine with 8K Microsoft Basic on ROM, a 32 character/line video interface, built-in cassette interface, a metal box with built-in full size ASCII keyboard, character graphics capability, 4K RAM (expandable to 8K on board), a machine language monitor that lets you examine/change memory, and expansion capability (to OSI's bus, of course). Whether or not you can live with a 32 character display (24 character if you use an RF modulator) is up to you, but for all the obvious benefits of such a machine, that may not be a critical disadvantage.

About the only thing really missing on the Challenger 1P is a user I/O port and interval timer. These would have to be added to do any useful hacking. There is an expansion connector with the address, data and control busses but I don't know if the signals are buffered. I'll try to get more details on this for upcoming issues.

In all fairness to you, the reader, I feel it should be mentioned that I have talked to a number of people who had complaints about the level of service and support they received from OSI. If any of you have dealt with OSI lately, I'd be interested in hearing about your experiences.

The Synertek SYM certianly has some very interesting things to offer.

Its list of good points include on-board RAM, EPROM, and I/O expansion capability, a powerful monitor and a high-speed (1500 baud) cassette interface. Obviously, SYM's creators were working to update and improve on the basic KIM design.

I could tell by the number of on-board strapping options and software switching logic that this machine was meant to be as versatile as possible.

How the SYM "stacks-up" will be the subject of future articles.

Rockwells bid for marketplace superiority is called the AIM 65. This is actually a two board machine - on one board is a full size ASCII style keyboard while the other holds the rest of the system.

AIM is unique in that it contains a 20 column thermal printer besides a 20 column alphanumeric LED display. Like SYM, AIM has on-board EPROM and RAM expansion capability and an advanced monitor. Its on-board printer would make it a likely candidate for the process control and system monitoring environment.

SYM and AIM both have expansion connectors configured to fit the standard KIM-4 motherboard.

Articles on both these machines will be published in the next issue.

Hudson Digital Electronics (see back cover) has been making great advances in S-44 KIMbus compatible hardware and software products. The one thing I most admire about this firm is their way of introducing new products.

continued on page 28

software feature: KIM HEXPAWN

From Robert C. Leedom, 14069 Stevens Valley Ct., Glenwood, MD 21738

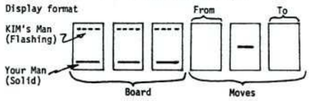
I was relieved to see (in Issue #12) that nobody's yet published a version HEXPAWN for KIM. I got my KIM in April, wrote HEXPAWN in May, and today (16 Oct 78) finally finished typing the listing. HEXPAWN first appeared in SCIENTIFIC American (Vol. 206, No. 3, Martin Gardner's "Mathematical Games"). The game is played on a 3 X 3 board. Each of the two players has three pieces, which move as chess pawns (move one square forward to vacant square, capture by moving one square diagonally to enemy piece's square). Object: get to your opponent's side of the board, or block him so that he cannot move.

This version was inspired by an article in the November 1975 BYTE, written by Bob Wier (with whom I corresponded on the subject of a "Super Star Trek" game in BASIC). Bob had written a HEXPAWN program for a 16-bit machine, and it took 4218 bytes (I assume they were 8-bit bytes). Unfortunately, (a) I have only the KIM-1 memory (and no access to an assembler), (b) the article only gave a general (top-level) flowchart and a move table, and (c) the article "Table of all Possible Board Positions and Moves" was both incomplete and incorrect, a fact I discovered only when I tried to play the game against my version of the program. Eventually, I solved problems (a), (b), and (c); here's the result:

Features of HEXPAWN for KIM-1

```
(a) Board coordinate  0 1 2  KIM's Men at 0, 1, 2
                      3 4 5
                      6 7 8  Your Men at 6, 7, 8
```

(b) Display format



(c) Program checks for (and only accepts) legal moves.

(d) KIM selects moves randomly, but learns. When When the computer loses, KIM's losing move is removed from the move table. Therefore, eventually (after 30 or so games) KIM should have only winning moves to select from!

(e) Two startup locations provided:
(1) Full initialization -- all possible KIM moves restored to move table. (Start at $100).
(2) New game initialization -- sets up board to play next game, but retains knowledge of previous bad moves. (Start AT $200).

(f) To allow tabulation/examination of the "learning" sequence, press and hold DA (Data Analysis) key at any time to display move # (0, 1, or 2 - there are three possible moves stored for each board position), Board index (see table at $10F) and Game number. Resume play upon release.

(g) Press PC (Person Concedes) to concede game to KIM

(h) After loading program, enter AD, 0100, GO. At any time, to restart the current game, press GO.

P.S. Have been using Radio Shack Supertape with a K-Mart (S.S. Kresge Co.) Model 6-33-01 cassette recorder (cost about $27) with 100% success using Hypertape program. However, in tape exchanges, others can only read my tapes about 75% of the time, and I have slightly less success reading theirs.

```
                 ; Page 0 locations used by program HEXPAWN
0000     FLSHR   RES  1    Timer for flashing KIM's men
0001     DBD     RES  3    Current board in Display format
0004     MASK    RES  3    Masks for flashing KIM's men
0007     EBD     RES  9    Current bd - Easy-to-read format
0010     WINDO   RES  6    Current 7-segment display
0017     MOVTYP  RES  1    KIM's last move (TO:FROM)
0018     TOG     RES  1    On/off indicator for KIM's men
0019     GAMNUM  RES  1    Game number
001A     BDNDX   RES  1    3*Bd # for model match/move select
001B     MOVNO   RES  1    KIM's last move # (0,1, or 2)
001C     PTO     RES  1    Person's last "to" move
001D     FROM    RES  1    If< 0, no "from" move yet;
                           if≥0, is.equal to the "from" move
001E     TMP     RES  1
001F     TMP1    RES  1
0020     POINTER RES  1
0021     POINTO  RES  1   · Page # (ADH) of MOVES
0022     MPOINT  RES  1
0023     MPOIN1  RES  1    Page # (ADH) of messages
0024     BGEBD   RES  9    Beginning bd - Easy-format:
                                 03 = KIM
                                 00 = space
                                 01 = Person
002D     MOVES   RES 99    Table of possible moves is placed
                              here by startup routine and
                              is modified as KIM "learns."


                  ; HEXPAWN for KIM-1. © Copyright May 1978 R.C.Leedom
0100 A2 6E    HXPNST LDX #$6E         Transfer moves, beginning
0102 BD 7E 01 INLP   LDA SPOINO,X        board, and pointer ADH's
0105 95 21           STA POINTO,X        to page zero.
0107 CA              DEX
0108 10 F8           BPL INLP
010A 85 19           STA GAMNUM       Set game # to zero.
010C 4C 00 02        JMP INIT
```

```
                 ; The following are the 33 board positions that
                 ; the HEXPAWN program will recognize after the
                 ; human opponent has moved.  The squares are num-
                 ; bered according to the scheme shown in the
                 ; comment field for CAPSET (location 03F8).
                 ; Here, the pieces and spaces (K=KIM, P=person,
                 ; and _=space) are packed by column -- that is,
                 ; in groups: 0,3,6; 1,4,7; 2,5,8. (For segment-
                 ; lighting, actual data is ordered 360,471,582.)
                 ;                      Bd #  BDNDX
010F 43 0B 0B BDMDL KP_,K_P,K_P         0     0
0112 0B 0B 43       K_P,K_P,KP_         1     3
0115 0B 43 0B       K_P,KP_,K_P         2     6
0118 C3 40 0B       KK_,_P_,K_P         3     9
011B 40 C3 0B       _P_,KK_,K_P         4     C
011E 43 48 03       KP_,_PP,K__         5     F
0121 43 03 48       KP_,K__,_PP         6     12
0124 08 C3 43       __P,KK_,KP_         7     15
0127 C8 43 43       _KP,KP_,KP_         8     18
012A C3 08 43       KK_,__P,KP_         9     1B
012D 43 43 C8       KP_,KP_,_KP         A     1E
0130 48 03 43       _PP,K__,KP_         B     21
0133 40 40 43       _P_,_P_,KP_         C     24
0136 08 43 03       __P,KP_,K__         D     27
0139 43 00 0B       KP_,___,K_P         E     2A
013C C0 C0 43       _K_,_K_,KP_         F     2D
013F 43 40 40       KP_,_P_,_P_         10    30
0142 C0 43 40       _K_,KP_,_P_         11    33
0145 40 43 C0       _P_,KP_,_K_         12    36
0148 C3 C0 40       KK_,_K_,_P_         13    39
014B 43 08 C3       KP_,__P,KK_         14    3C
014E 00 43 0B       ___,KP_,K_P         15    3F
0151 40 C0 C3       _P_,_K_,KK_         16    42
0154 C0 40 03       _K_,_P_,K__         17    45
0157 40 C3 00       _P_,KK_,___         18    48
015A 00 C3 40       ___,KK_,_P_         19    4B
015D 0B 00 43       K_P,___,KP_         1A    4E
0160 00 40 C3       ___,_P_,KK_         1B    51
0163 0B 40 C3       K_P,_P_,KK_         1C    54
0166 03 48 43       K__,_PP,KP_         1D    57
0169 43 C8 43       KP_,_KP,KP_         1E    5A
016C 00 43 00       ___,KP_,___         1F    5D
016F C3 40 00       KK_,_P_,___         20    60
                 ;
                 ; End-game messages
0172 3E 00 38 KWIN   DATA $3E,00,$38,$3F,$6D,$79
0175 3F 6D 79
0178 00 54 1C PWIN   DATA 00,$54,$1C,$78,$6D,00
017B 78 6D 00
      72      KWAD   EQU  @KWIN-$100
      78      PWAD   EQU  @PWIN-$100
                 ;
                 ; The following data is saved here for startup
                 ;                initialization.
017E 00       SPOINO $00
017F 00       SMPOIN $00
0180 01       SMPOI1 $01
```

```
0181 03 03 03 SBGEPD DATA 03,03,03
0184 00 00 00        DATA 00,00,00
0187 01 01 01        DATA 01,01,01
018A 31 41 52 SMVTBL DATA $31,$41,$52 BDNDX 0
018D 30 41 51        DATA $30,$41,$51       3
0190 30 40 00        DATA $30,$40,0         6
0193 40 42 63        DATA $40,$42,$63       9
0196 31 52 74        DATA $31,$52,$74       C
0199 40 42 52        DATA $40,$42,$52       F
019C 31 41 51        DATA $31,$41,$51       12
019F 51 64 74        DATA $51,$64,$74       15
01A2 51 42 00        DATA $51,$42,0         18
01A5 63 73 00        DATA $63,$73,0         1B
01A8 40 31 00        DATA $40,$31,0         1E
01AB 31 41 51        DATA $31,$41,$51       21
01AE 42 00 00        DATA $42,0,0           24
01B1 42 52 00        DATA $42,$52,0         27
01B4 52 00 00        DATA $52,0,0           2A
01B7 63 74 00        DATA $63,$74,0         2D
01BA 40 00 00        DATA $40,0,0           30
01BD 51 63 00        DATA $51,$63,0         33
01C0 31 85 00        DATA $31,$85,0         36
01C3 63 74 00        DATA $63,$74,0         39
01C6 75 85 00        DATA $75,$85,0         3C
01C9 42 52 00        DATA $42,$52,0         3F
01CC 74 85 00        DATA $74,$85,0         42
01CF 63 42 52        DATA $63,$42,$52       45
01D2 74 31 00        DATA $74,$31,0         48
01D5 74 51 00        DATA $74,$51,0         4B
01D8 30 00 00        DATA $30,0,0           4E
01DB 42 85 00        DATA $42,$85,0         51
01DE 30 40 85        DATA $30,$40,$85       54
01E1 30 40 42        DATA $30,$40,$42       57
01E4 00 00 00        DATA 0,0,0             5A
01E7 00 00 00        DATA 0,0,0             5D
01EA 63 40 00        DATA $63,$40,0         60
```

```
0200 A2 07    INIT   LDX #$07       Initialize right
0202 A9 00           LDA #$00          side
0204 95 10    INITLP STA WINDO,X       of display
0206 CA              DEX               (plus MOVTIM, MOVTYP)
0207 10 FB           BPL INITLP        to await
0209 A9 C0           LDA #$C0          person's
020B 85 14           STA WINDO+4       move.
020D A2 08           LDX #$08
020F B5 24    BDINIT LDA BGEBD,X    Transfer beginning board (in
0211 95 07           STA EBD,X         Easy-format) to current
0213 CA              DEX               board.
0214 10 F9           BPL BDINIT
0216 86 1D           STX FROM       Indicate no "from" move yet.
0218 A2 03    DISPLT LDX #$03       Clear the
021A A9 00    DSPLP  LDA #$00          "Display-format"
021C 95 00           STA FLSHR,X       board and
021E CA              DEX               the flasher-timer.
021F 10 F9           BPL DSPLP
0221 A0 02           LDY #$02       Start with 3rd char of board.
0223 18              CLC
0224 84 1E    NXDIG  STY TEMP
0226 A9 06           LDA #$06       Set up X to start with
0228 65 1E           ADC TEMP          lower segment for
022A AA              TAX               this character.
022B A9 00           LDA #$00       Clear A so can OR segments.
022D 6A 6A    NXSEG  ROR ROR        In this loop, shift the
022F 6A              ROR               segments into place.
0230 15 07           ORA EBD,X      OR 3 for KIM, 1 for person.
0232 CA CA           DEX DEX        Point to next
0234 CA              DEX               higher segment.
0235 10 [illegible]  BPL NXSEG      Loop till character done.
0237 [illegible]     STA [illegible],Y  Save completed char; so
023A 88              DEY               do next one to
023B 10 E7           BPL NXDIG         the left.
```

```
              ; Main loop begins here
023D C6 00    DISPLO DEC FLSHR       Time to flip KIM bits?
023F 10 2E           BPL LITEST      No, just show current pattern.
0241 A9 30           LDA #$30        Yes.  Reset
0243 85 00           STA FLSHR          timer.
0245 A2 02           LDX #$02        Form the
0247 B5 01    GETMSK LDA DBD,X          flasher-mask
0249 4A              LSR                patterns
024A 29 49           AND #$49           for
024C 95 04           STA MASK,X         the
024E CA              DEX                current
024F 10 F6           BPL GETMSK         board.
0251 A2 02           LDX #$02        Set X for next loop.
0253 A5 18           LDA TOG         Toggle to
0255 49 80           EOR #$80           alternate 1's and 0's
0257 85 18           STA TOG            for KIM's men.
0259 30 09           BMI WNDSET      Go do 0's.
025B A9 00           LDA #$00
025D 95 04    ZERMSK STA MASK,X      Clear masks so
025F CA              DEX                can do
0260 10 FB           BPL ZERMSK         1's.
0262 A2 02           LDX #$02
0264 B5 01    WNDSET LDA DBD,X       Use the
0266 29 49           AND #$49           masks
0268 55 04           EOR MASK,X         to flip
026A 95 10           STA WINDO,X        the bits.
026C CA              DEX
026D 10 F5           BPL WNDSET
              ; Output to KIM's 7-segment displays
026F A9 7F    LITEST LDA #$7F        Set directional
0271 8D 41 17        STA PADD           registers.
0274 A0 00           LDY #$00
0276 A2 09           LDX #$09        Start with leftmost char.
0278 B9 10 00 LITE   LDA WINDO,Y     Get character.
027B 84 FC           STY TEMP
027D 20 4E 1F        JSR CONVD+6     Output character.
0280 C8              INY
0281 C0 06           CPY #$06        Done all six yet?
0283 90 F3           BCC LITE        Not yet, continue.
0285 20 3D 1F        JSR $1F3D       Turn off digits.
              ; Keyboard input begins here
0288 D8       KEYGET CLD
0289 20 40 1F        JSR KEYIN
028C 20 6A 1F        JSR GETKEY
028F C9 13           CMP #$13        GO key?
0291 D0 03           BNE DACHK
0293 4C 00 02        JMP INIT        Yes, start new game.
0296 C9 11    DACHK  CMP #$11        DA key?
0298 D0 0E           BNE GIPROG
029A A2 02           LDX #$02        Yes, display (for Data Analysis)
029C B5 19    DALP   LDA GAMNUM,X       from left to right:
029E 95 F9           STA INH,X          Move # (00,01, or 02),
02A0 CA              DEX                Board index (Bd # * 3),
02A1 10 F9           BPL DALP           Game #. (2 digits each)
02A3 20 1F 1F        JSR SCANDS         Keep doing this till DA
02A6 10 E0           BPL KEYGET         released; then resume play.
02A8 A6 14    GIPROG LDX WINDO+4     Is game still
02AA E0 C0           CPX #$C0           in progress?
02AC D0 C1           BNE LITEST      No. Keep showing endgame msg.
02AE C9 14           CMP #$14        PC key?
02B0 F0 78           BEQ KWLINK      Yes, Person Concedes.
02B2 A6 16           LDX MOVTIM      Person's turn to move?
02B4 D0 67           BNE TIMEDS      No, go time display.
02B6 20 C0 03        JSR LEGMOV      Yes. Did he make legal move?
02B9 10 35           BPL PERLM       Yes. Go execute it.
02BB A9 08           LDA #$08        He didn't make a legal move,
02BD 85 1E           STA TMP            does he have one?  Try
02BF 20 C0 03 LMCHK  JSR LEGMOV         each position to see.
02C2 10 0D           BPL TOMVCK

02C4 C6 1E    NXFMCK DEC TMP         Try
02C6 A5 1E           LDA TKP            next
02C8 10 F5           BPL LMCHK          position.
02CA 85 16           STA MOVTIM      Tried all, no luck, no legal
02CC 85 17           STA MOVTYP         moves possible.  Set KIM
02CE 4C 3D 02 FNMVLP JMP DISPLO         win display after delay.
              ; Continue looking for valid move for person
02D1 A5 1D    TOMVCK LDA FROM        Was valid move a "to" move?
02D3 10 F9           BPL FNMVLP      Yes, he can therefore move.
02D5 86 1D           STX FROM        No. Given this "from" move,
02D7 A9 06           LDA #$06           try all possible
02D9 85 1F           STA TMP1           "to" moves.
02DB 20 C0 03 LTMCHK JSR LEGMOV      Find one?
02DE 10 0A           BPL OKMOV          Yes. He's got a move.
02E0 C6 1F           DEC TMP1        Try
02E2 A5 1F           LDA TMP1           next
02E4 10 F5           BPL LTMCHK         position.
02E6 86 1D           STX FROM        Tried all "to" moves; look
02E8 30 DA           BMI NXFMCK         for another "from" move.
              ; Have found a possible "from-to" move for person.
02EA A9 FF    OKMOV  LDA #$FF        Has got a move he could make,
02EC 85 1D           STA FROM           so restore FROM and
02EE 30 DE           BMI FNMVLP         continue the game.
```

```
                ; Person has entered a legal move.
02F0 A9 00      PERLM  LDA #$00        Clear the "to" indication
02F2 85 15             STA WINDO+5        left from KIM's move.
02F4 A5 1D             LDA FROM        Was this a "from" move?
02F6 30 1B             BMI FRMDIS      Yes, display it; save move.
02F8 86 1C             STX PTO         No, save as person's "to".
02FA A0 01             LDY #$01        Set "person" indicator.
02FC A9 FF      TMINIT LDA #$FF        Set timer; not person's move.
02FE 85 16      MAKMOV STA MOVTIM      Save move-timer.
0300 94 07             STY EBD,X       Place piece on board.
0302 BD E7 1F          LDA DIGCOD,X    Get 7-segment code for
0305 85 15             STA WINDO+5        "to" indication on board.
0307 A0 00             LDY #$00        Remove piece from
0309 A6 1D             LDX FROM           previous
030B 94 07             STY EBD,X          board position.
030D A9 FF             LDA #$FF        Prepare for next
030F 85 1D             STA FROM           "from" move.
0311 30 02             BMI DISX        Go show this "from" move.
                ; This code displays "from" moves.
0313 86 1D      FRMDIS STX FROM        Save "from" move.
0315 BD E7 1F   DISX   LDA DIGCOD,X    Use "from" in X to get 7-seg
0318 85 13             STA WINDO+3        indication.
031A 4C 18 02          JMP DISPLT      Return to main loop.
                ;
031D A5 18      TIMEDS LDA TOG         Time to decrement move timer?
031F 30 AD             BMI FNMVLP      Not yet.
0321 C6 16             DEC MOVTIM      Yes. Ready for next move?
0323 D0 A9             BNE FNMVLP      Not yet.

0325 A5 17      KWCHK  LDA MOVTYP      Has KIM moved
0327 4A                LSR                to either
0328 C9 30             CMP #$30           6,7,or 8?
032A 10 5B      KWLINK BPL KIMWIN      Yes. KIM won.
032C A5 1C             LDA PTO         Has person moved
032E C9 03             CMP #$03           to 0,1,or 2?
0330 30 44             BMI PERWIN      Yes.  Person won.
                ; Try to match current board with stored model.
0332 A0 60             LDY #$60        (#models - 1)*3 = 32*3 = 96
0334 BE 0F 01   MDLCHK LDX BDMDL,Y
0337 E4 01             CPX DBD         First column match?
0339 D0 0E             BNE NXBD       .No, try next board model.
033B BE 10 01          LDX BDMDL+1,Y   Yes, does
033E E4 02             CPX DBD+1          second column match?
0340 D0 07             BNE NXBD        No, try next board model.
0342 BE 11 01          LDX BDMDL+2,Y   Yes, does
0345 E4 03             CPX DBD+2          third column match?
0347 F0 07             BEQ GOTMDL      Yes, found model. Go get move.
0349 88 88      NXBD   DEY DEY         Point to
034B 88                DEY                next board model
034C 10 E6             BPL MDLCHK         and keep comparing.
034E 30 33             BMI PWMSG       No models found; have KIM
                                          concede the game.
                ; Pick one of the remaining moves for this position.
0350 AD 04 17   GOTMDL LDA TIMER       Use the timer to
0353 29 03             AND #$03           arbitrarily select
0355 AA                TAX                move 0, 1, or 2.
0356 F0 01             BEQ POK            (This code picks #2
0358 CA                DEX                half the time.)
0359 A9 02      POK    LDA #$02        Inititialize the counter for
035B 85 1E             STA TMP            how many moves to try (3).
035D 86 1F      MVSLLP STX TMP1        Temporary move number.
035F A9 2D      MVLP1  LDA ADMVTB
0361 18                CLC
0362 65 1F             ADC TMP1        Set ADL of pointer to pick
0364 85 20             STA POINTER        up this move.
0366 B1 20             LDA (POINTER),Y
0368 D0 30             BNE GOTMOV      Got a valid move -- use it!
036A C6 1E             DEC TMP
036C 30 08             BMI PERWIN      No moves left; KIM resigns.
036E C6 1F             DEC TMP1        Try next move
0370 10 ED             BPL MVLP1          in the set. (May try
0372 A2 02             LDX #$02           in order 2,1,0; 1,0,2;
0374 10 E7             BPL MVSLLP         or 0,2,1.)
0376 A4 1A      PERWIN LDY BDNDX       Person has won.  ************
0378 18                CLC             Compute
0379 A9 2D             LDA ADMVTB         the ADL
037B 65 1B             ADC MOVNO          of KIM's
037D 85 20             STA POINTER        last move.(POINTER+1 = 0)
037F A9 00             LDA #$00        Wipe out the last
0381 91 20             STA (POINTER),Y   move KIM made.
0383 A9 78             LDA PWAD        Get address of "person won"
0385 10 02             BPL STAD           message.
0387 A9 72      KIMWIN LDA KWAD        Get "KIM won" msg address.
0389 85 22      STAD   STA MPOINT      Point to message address (ADL)

                ; Display end-of-game message
038B A0 05             LDY #$05
038D B1 22      FILWIN LDA (MPOINT),Y Store the six-letter
038F 99 10 00          STA WINDO,Y        message in the window.
0392 88                DEY
0393 10 F8             BPL FILWIN
0395 E6 19             INC GAMNUM      Increment game number.
0397 4C 6F 02          JMP LITEST      Show msg and wait for GO.
```

```
                   ; Make KIM's chosen move
039A 85 17     GOTMOV STA MOVTYP     Save move for later checks.
039C 84 1A            STY BDNDX      Save board pointer.
039E A6 1F            LDX TMP1       Pick up
03A0 86 13            STX MOVNO         move # (=0,1, or 2).
03A2 48 48            PHA PHA        Save 2 copies of move type.
03A4 4A 4A            LSR LSR
03A6 4A 4A            LSR LSR
03A8 AA               TAX            Place "to" move in X.
03A9 68               PLA
03AA 29 0F            AND #$0F       Extract and save
03AC 85 1D            STA FROM          "from" move.
03AE A0 03            LDY #$03       Indicate KIM move being made.
03B0 68 4A            PLA LSR        Is upper half-byte of
03B2 C9 30            CMP #$30          move a 6,7, or 8?
03B4 30 03            BMI NOKWIN     No. KIM hasn't won.
03B6 4C FC 02         JMP TIMINIT    Yes. Show winning move.
03B9 A9 00     NOKWIN LDA #$00       Indicate it's person's
03BB 4C FE 02         JMP MAKMOV        move, and make KIM's.
               ; Subroutine to test for legal player move.
               ; Call: with move in A.
               ; Returns: with X = $FF if illegal move
               ;                   X = move if legal
03BE 00 00
03C0 C9 09     LEGMOV CMP #$09       Is move 0 to 8?
03C2 B0 22            BCS MOVNFG     No, illegal.
03C4 AA               TAX
03C5 B5 07            LDA EBD,X      Extract player indicator from
03C7 6A 6A            ROR ROR           board: 1 player, 3 KIM.
03C9 10 27            BPL TOCHK      Nobody here, but OK if "to".
03CB A5 1D            LDA FROM       Is this a "from" move?
03CD 10 04            BPL PTOMOV     No. Go see if legal "to".
03CF B0 15            BCS MOVNFG     Yes, but KIM's here! Bad.
03D1 8A        MOVOK  TXA            Return. Legal move was made
03D2 60               RTS               (or found possible).
03D3 8A        PTOMOV TXA            Place "to" move in A.
03D4 90 10            BCC MOVNFG     Person here! Can't capture!
03D6 0A 0A            ASL ASL        KIM here. A capture. Can
03D8 0A 0A            ASL ASL           only be one of 8 possible
03DA 05 1D            ORA FROM          moves. Format "TO:FROM",
03DC A0 07            LDY #$07          and test against
03DE D9 F8 03 CAPCHK CMP CAPSET,Y       each possibility.
03E1 F0 EE            BEQ MOVOK      Found it! Move is OK.
03E3 88               DEY
03E4 10 F8            BPL CAPCHK
03E6 A2 FF     MOVNFG LDX #$FF       Move illegal. Set indicator.
03E8 60               RTS            (Second of two return points.)
03E9 8A        NOCAP  TXA            If here, not capture, C=0.
03EA 69 03            ADC #$03       Is move + 3 = FROM?
03EC C5 1D            CMP FROM
03EE D0 F6            BNE MOVNFG     No.  Illegal move.
03F0 F0 DF            BEQ MOVOK
03F2 A4 1D     TOCHK  LDY FROM       Space here. "To" move?
03F4 30 F0            BMI MOVNFG     No. Illegal move.
03F6 10 F1            BPL NOCAP      Yes. See if valid move.
               ;
               ; Set of all possible "capture" moves by person,
               ;   packed in "TO:FROM" format.
               ;
               ;   Captures are from to              Board format
03F8 13        CAPSET DATA $13     3   1             !   !   !   !
03F9 04               DATA $04     4   0             ! 0 ! 1 ! 2 !
03FA 24               DATA $24     4   2             !___!___!___!
03FB 15               DATA $15     5   1             !   !   !   !
03FC 46               DATA $46     6   4             ! 3 ! 4 ! 5 !
03FD 37               DATA $37     7   3             !___!___!___!
03FE 57               DATA $57     7   5             !   !   !   !
03FF 48               DATA $48     8   4             ! 6 ! 7 ! 8 !
                                                     !___!___!___!
```

HEXPAWN HEX DUMP

```
      0  1  2  3  4  5  6  7  8  9  A  B  C  D  E  F
0100 A2 6E BD 7E 01 95 21 CA 10 F8 85 19 4C 00 02 43
0110 0B 0B 0B 0B 43 0B 43 0B C3 40 0B 40 C3 0B 43 48
0120 03 43 03 48 08 C3 43 C8 43 43 C3 08 43 43 43 C8
0130 48 03 43 40 40 43 08 43 03 43 00 0B C0 C0 43 43
0140 40 40 C0 43 40 40 43 C0 C3 C0 40 43 08 C3 00 43
0150 0B 40 C0 C3 C0 40 03 40 C3 00 00 C3 40 0B 00 43
0160 00 40 C3 0B 40 C3 03 48 43 43 C8 43 00 43 00 C3
0170 40 00 3E 00 38 3F 6D 79 00 54 1C 78 6D 00 00 00
0180 01 03 03 03 00 00 00 01 01 01 31 41 52 30 41 51
0190 30 40 00 40 42 63 31 52 74 40 42 52 31 41 51 51
01A0 64 74 51 42 00 63 73 00 40 31 00 31 41 51 42 00
01B0 00 42 52 00 52 00 00 63 74 00 40 00 00 51 63 00
01C0 31 85 00 63 74 00 75 85 00 42 52 00 74 85 00 63
01D0 42 52 74 31 00 74 51 00 30 00 00 42 85 00 30 40
01E0 85 30 40 42 00 00 00 00 00 00 63 40 00
```

```
0200 A2 07 A9 00 95 10 CA 10 FB A9 C0 85 14 A2 08 B5
0210 24 95 07 CA 10 F9 86 1D A2 03 A9 00 95 00 CA 10
0220 F9 A0 02 18 84 1E A9 06 65 1E AA A9 00 6A 6A 6A
0230 15 07 CA CA CA 10 F6 99 01 00 88 10 E7 C6 00 10
0240 2E A9 30 85 00 A2 02 B5 01 4A 29 49 95 04 CA 10
0250 F6 A2 02 A5 18 49 80 85 18 30 09 A9 00 95 04 CA
0260 10 FB A2 02 B5 01 29 49 55 04 95 10 CA 10 F5 A9
0270 7F 8D 41 17 A0 00 A2 09 B9 10 00 84 FC 20 4E 1F
0280 C8 C0 06 90 F3 20 3D 1F D8 20 40 1F 20 6A 1F C9
0290 13 D0 03 4C 00 02 C9 11 D0 0E A2 02 B5 19 95 F9
02A0 CA 10 F9 20 1F 1F 10 E0 A6 14 E0 C0 D0 C1 C9 14
02B0 F0 78 A6 16 D0 67 20 C0 03 10 35 A9 08 85 1E 20
02C0 C0 03 10 0D C6 1E A5 1E 10 F5 85 16 85 17 4C 3D
02D0 02 A5 1D 10 F9 86 1D A9 06 85 1F 20 C0 03 10 0A
02E0 C6 1F A5 1F 10 F5 86 1D 30 DA A9 FF 85 1D 30 DE
02F0 A9 00 85 15 A5 1D 30 1B 86 1C A0 01 A9 FF 85 16
```

```
0300 94 07 BD E7 1F 85 15 A0 00 A6 1D 94 07 A9 FF 85
0310 1D 30 02 86 1D BD E7 1F 85 13 4C 18 02 A5 18 30
0320 AD C6 16 D0 A9 A5 17 4A C9 30 10 5B A5 1C C9 03
0330 30 44 A0 60 BE 0F 01 E4 01 D0 0E BE 10 01 E4 02
0340 D0 07 BE 11 01 E4 03 F0 07 88 88 88 10 E6 30 33
0350 AD 04 17 29 03 AA F0 01 CA A9 02 85 1E 86 1F A9
0360 2D 18 65 1F 85 20 B1 20 D0 30 C6 1E 30 08 C6 1F
0370 10 ED A2 02 10 E7 A4 1A 18 A9 2D 65 1B 85 20 A9
0380 00 91 20 A9 78 10 02 A9 72 85 22 A0 05 B1 22 99
0390 10 00 88 10 F8 E6 19 4C 6F 02 85 17 84 1A A6 1F
03A0 86 1B 48 48 4A 4A 4A 4A AA 68 29 0F 85 1D A0 03
03B0 68 4A C9 30 30 03 4C FC 02 A9 00 4C FE 02 00 00
03C0 C9 09 B0 22 AA B5 07 6A 6A 10 27 A5 1D 10 04 B0
03D0 15 8A 60 8A 90 10 0A 0A 0A 0A 05 1D A0 07 D9 F8
03E0 03 F0 EE 88 10 F8 A2 FF 60 8A 69 03 C5 1D D0 F6
03F0 F0 DF A4 1D 30 F0 10 F1 13 04 24 15 46 37 57 48
      0  1  2  3  4  5  6  7  8  9  A  B  C  D  E  F
```

6502 OP CODES

Arranged in logical order by Jim Butterfield, Toronto

	IMM 2	ZPAG 2	Z,X 2	Z,Y 2	ABS 3	A,X 3	A,Y 3
ASL		06	16		0E	1E	
ROL		26	36		2E	3E	
LSR		46	56		4E	5E	
ROR		66	76		6E	7E	
STX		86		96	8E		
LDX	A2	A6		B6	AE		BE
DEC		C6	D6		CE	DE	
INC		E6	F6		EE	FE	

Op Code ends in -2, -6, or -E

	IMM 2	ZPAG 2	Z,X 2	ABS 3	A,X 3
BIT		24		2C	
STY		84	94	8C	
LDY	A0	A4	B4	AC	BC
CPY	C0	C4		CC	
CPX	E0	E4		EC	

Misc. -0, -4, -C

	IMM 2	ZPAG 2	Z,X 2	(I,X) 2	(I),Y 2	ABS 3	A,X 3	A,Y 3
ORA	09	05	15	01	11	0D	1D	19
AND	29	25	35	21	31	2D	3D	39
EOR	49	45	55	41	51	4D	5D	59
ADC	69	65	75	61	71	6D	7D	79
STA		85	95	81	91	8D	9D	99
LDA	A9	A5	B5	A1	B1	AD	BD	B9
CMP	C9	C5	D5	C1	D1	CD	DD	D9
SBC	E9	E5	F5	E1	F1	ED	FD	F9

Op Code ends in -1, -5, -9, or -D

BPL	10	BMI	30
BVC	50	BVS	70
BCC	90	BCS	B0
BNE	D0	BEQ	F0

Branches -0

	ABS	(IND)
JSR	20	
JMP	4C	6C

Jumps

	0-	1-	2-	3-	4-	5-	6-	7-	8-	9-	A-	B-	C-	D-	E-	F-
-0	BRK				RTI		RTS									
-8	PHP	CLC	PLP	SEC	PHA	CLI	PLA	SEI	DEY	TYA	TAY	CLV	INY	CLD	INX	SED
-A	ASL-A		ROL-A		LSR-A		ROR-A		TXA	TXS	TAX	TSX	DEX		NOP	

Single-byte Op Codes -0, -8, -A

Another OP-CODE chart? Yes, but there is a reason.

This chart groups the codes logically. This way, you get three benefits.

First, you get to see how the codes are classified and decoded. A glance at the chart shows that LDA and ADC, for example, are close cousins: same addressing modes, same timing, and quite similar OP-CODES; on the other hand, LDA and LDX are noticeably different. The classification idea can be useful to those who want to dig into op-codes, say to write an assembler or a disassembler.

Secondly, it's handy for looking up an OP-CODE-maybe easier than an alphabetical list. You'll very quickly learn to look at the right box and spot the code you want right away. As you get used to the groupings, you'll also develop a feel for the addressing modes that are allowed.

Thirdly, you'll find it convenient for identifying an unknown op-code---("What the heck is CE, anyway?")

Jim B.

EDITORS NOTE: I have found this chart to be extremely useful in designing opcode decode algorithms etc.

TAPE FILE RECOVERY ROUTINE

Joel Swank
4655 SW 142nd #186
Beaverton, Or 97005

Ever have a tape file with a dropout? One that fails on the same byte every time. There must be good data behind that dropout, but how to get at it?

The normal tape read routine quits when it gets an invalid character. Instead the recovery routine flags the error by storing an asterisk(*) in memory, and begins reading bits looking for a valid character. When it gets one it resumes reading the file. The only problem is that there is no way of telling whether the first valid character is the first half of a byte or the last half of a byte. To overcome this problem the routine uses an external flag byte (HALF) to determine what to do with the first valid character after a dropout. A bit is shifted out of the high order end of HALF each time a recovery is attempted. If the bit is zero the first valid character is ignored. If it is one the first valid character is used to form the first valid byte. Upon each entry into recovery mode the counter ERRC is incremented. If it wraps to zero the program is aborted.

To recover a file initialize $17F5-$17F9 as usual and set HALF ($CB) to $00. Start the program at $200. When $FFFF appears look at ERRC ($C7). If the count is low then examine the data to find the errors marked '*' ($2A). Determine where the data is a half byte out of sync and set one bits in HALF accordingly. Rerun the program and the data, minus the dropouts should be in memory. I have recovered files with two dropouts, it should work for as many as 8.

If you have a file that has a dropout in the sync pattern and won't sync-up it may be recovered by using SCAN ($298) as entry point, effectively starting in recovery mode. You must first initialize VEBB ($17EC) with $8Dnnnn60, where nnnn is the address where the data is to be stored. Also zero ERRC. The ID and start address will be read and stored like data. It is also possible to begin reading files in the middle in this manner. The routine also performs the special tape read functions (ID=00 or FF). Thanks to Jim Butterfield for use of his synchronizaiton code.

```
1090                    :  ZERO PAGE STORAGE
1100                    :
1110                    HALF .DL 00CB
1120                    ERRC .DL 00C7     ERROR COUNT
1130                    INH  .DL 00F9
1140                    :
1150                    :  EXTERNAL LABELS
1160                    :
1170                    VEBB .DL 17EC
1180                    SAL  .DL 17F5
1190                    SAH  .DL 17F6
1200                    EAL  .DL 17F7
1210                    EAH  .DL 17F8
1220                    ID   .DL 17F9
1230                    INTV .DL 1932
1240                    END0 .DL 1925
1250                    ENDF .DL 1929
1260                    INVB .DL 1932
1270                    RDBY .DL 19F3
1280                    RDCH .DL 1A24
1290                    CHKT .DL 194C
1300                    INCV .DL 19EA
1310                    SBD  .DL 1742
1320                    PAKT .DL 1A00
1330                    RDBT .DL 1A41
1340                    CHKL .DL 17E7
1350                    CHKH .DL 17E8
1360                    :
1370                    :  ENTRY POINT
1380                    :
1390 0200 A9 8D         RECV LDA 8D       OPCODE FOR STA
1400 0202 8D EC 17           STA VEBB     INTO VEBB
1410 0205 A9 00              LDA 00
1420 0207 85 C7              STA *ERRC    INIT COUNT
1430 0209 C9 FF              CMP 0FF
1440 020B 20 32 19           JSR INVB     INIT VEBB
1450 020E A9 07              LDA 07       DIRECTIONAL REG
1460 0210 8D 42 17           STA SBD
1470 0213 20 41 1A      SYN  JSR RDBT     GET A BIT
1480 0216 46 F9              LSR *INH
1490 0218 05 F9              ORA *INH     SHIFT INTO LEFT OF INH
1500 021A 85 F9              STA *INH
1510 021C C9 16         TST  CMP 16       SYNC CHARACTER?
1520 021E D0 F3              BNE SYN      NO - KEEP LOOKIN
1530 0220 20 24 1A           JSR RDCH     GET A CHARACTER
1540 0223 C6 F9              DEC *INH     COUNT 22 SYNCS
1550 0225 10 F5              BPL TST
1560 0227 C9 2A              CMP '*       * FLAGS START OF RECORD
1570 0229 D0 F1              BNE TST      IF NOT - THEN MUST BE SYNC
1580 022B 20 F3 19           JSR RDBY     GET BYTE
1590 022E CD F9 17           CMP ID       CORRECT RECORD?
1600 0231 F0 1C              BEQ LOAD     YES - READ IT
1610 0233 AD F9 17           LDA ID       ID=0?
1620 0236 F0 13              BEQ LOAD     YES - READ IT ANYWAY
1630 0238 C9 FF              CMP 0FF      ID=FF?
1640 023A D0 D7              BNE SYN      NO TRY NEXT
1650 023C 20 F3 19           JSR RDBY     YES IGNORE SA OF TAPE
1660 023F 20 4C 19           JSR CHKT
1670 0242 20 F3 19           JSR RDBY
1680 0245 20 4C 19           JSR CHKT
1690 0248 38                 SEC
1700 0249 B0 12              BCS BYTE     RELATIVE JUMP
1710 024B 20 F3 19      LOAD JSR RDBY     READ START ADDRESS AND SAVE
1720 024E 8D ED 17           STA VEBB+01
1730 0251 20 4C 19           JSR CHKT
1740 0254 20 F3 19           JSR RDBY
1750 0257 8D EE 17           STA VEBB+02
```