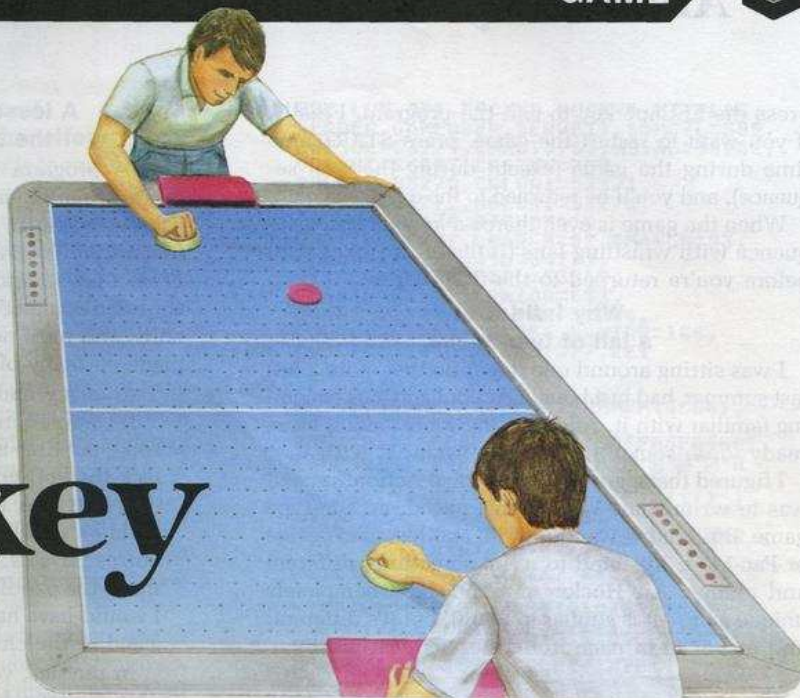# Air Hockey

by Chris Page

Air Hockey is written in Action! and must be compiled off of a disk or tape (the source and object code won't fit in memory together). So type it in, SAVE it, clear the editor, go to the monitor and RUN it, thusly: R "FILENAME" (substituting the device and filename you saved it under for "FILENAME"—I used "D:AIRHOCKE.Y").

Once you have it running properly, you should see the title screen and hear the title music (the "Peter Gunn" bass line). Press START.

Now you should see the options screen (it has the word *OPTIONS* at the top). You can use the OPTION key to highlight a different option, SELECT to change the option and START to play the game.

The options available in **Air Hockey** are:

Friction — This can be ON or OFF. If the friction is on, the puck will have a tendency to slow down while travelling across the board. You may notice that it sometimes curves as it slows down. This is because I used integer values instead of floating point. This means that the motion is not 100 percent accurate, resulting in the curved motion of the puck.

Velocity — This can be 2 through 9 and indicates the maximum velocity of the puck. Option 2 is slowest; 9 is fastest.

Bounce — This can be 0 through 9, indicating the amount of "bounce" to the puck, or how well it re-tains velocity after hitting the side of the board. A 9 means that the puck will not slow down on collision; 0 indicates very little bounce and will cause the puck to slow down considerably if it hits the sides.

Win — This is the score up to which the player(s) will play. It can be from 10 to 90, in increments of 10.

Players — Either 1 or 2. If one player is selected, then the player should use joystick port 1 and control the top player; the computer will control player two, the bottom player. If two players are selected, then it's the same, except that player two will be controlled by joystick port 2.

Playing.

Once your options are set (or left alone, if you like the default settings) you may press START to play. You'll then see a vertical air hockey board with the scores displayed at the top, along with the score necessary to win the game. The puck will appear in front of the serving player's paddle. That player must hit the puck to start the game.

The game is something like **Pong**, with forward motion as well as side-to-side. Players control the paddles by moving the joystick in the direction they wish to move. The buttons do nothing. A score is made when the puck goes into the yellow goal area of a player, and the other player becomes the server.

If the puck gets stuck between players, as it can in real air hockey, you may re-serve by pressing the SPACE BAR. Also, while in the play mode, you may

press the ESCape key to exit the program. Finally, if you want to restart the game, press START anytime during the game (except during the goal sequence), and you'll be returned to the options screen.

When the game is over, there's a long cheering sequence with whistling fans (if the crowd likes you) before you're returned to the title screen.

Why I did it or a tail of two ducks.

I was sitting around one day (I do that quite a lot) last summer, had just bought Action! and was becoming familiar with it. After writing some demos, I was ready to do something more substantial with it.

I figured that a good way to utilize Action!'s speed was to write some kind of fast-paced, arcade-type game. But I didn't want to write another space game or **Pac-Man**. I wanted to write something different and unique. **Air Hockey** may not be a completely unique game (it *is* similar to **Pong**), but it's different, and a change of pace from "Laser the Aliens!"

A lesson in compromise or the quacker in the rye.

This program is an example of inventiveness, procrastination and compromise. "Inventive" because. . .well, it's a matter of opinion, but *I* think it's inventive. "Procrastination" because I dropped the project for several months at a time (check out the start and completion dates in the source listing). Finally, and most decidedly, "compromise" because I made so many of these concerning **Air Hockey**.

Two of my main compromises were:

1) I wanted more options and a complete title song, but time and a willingness to work (or the lack thereof) got in the way.

2) Everyone, including myself, thought that the paddles should have been round. But that requires physics. . .I barely passed physics. If I'd made the paddles round, as they are in real life, I would have had to resort to "real" physics instead of the chintzy method I did use.

In physics, you use vectors to describe how objects move, but this is a difficult thing in integers (well, difficult for me), which is what I was stuck with in Action! So I simply gave a horizontal and vertical speed and a horizontal and vertical direction.

These two compromises, however, were not as difficult or as important as my final compromise. I had to compromise on the one aspect, the most difficult thing, that every programmer must: completion of my goal. I had to stop work on a program which I felt was incomplete and short of my goals, and call it finished.

I realized this when I was telling my friends that I'd have to add the treble line to the title music before I'd submit it for publishing. As I told them this, I realized that the game is rather simple (as it was supposed to be) and that a full-blown song was superfluous. I then realized that other things I wanted to add were also not needed.

Actually, I had already met my goal (design a simple game as an exercise in Action!), but in the process, I'd created other goals—like adding the treble line.

I realized that I would continue creating goals as long as I was working on it; I would never finish the game. I'd sit, perma-bonded to my video screen, for the next ten years working on **Air Hockey** until it was 3-D, talked and had instant replay, a high score list, a theme song to put "Flight of the Valkyries" to shame, and a thousand other things. . .and I would still want to change something.

Program design and
some ducks thrown in for effect.

I think the important thing here is to realize that, when you want to write a program, you should decide exactly what it will be like, so that you can say it is finished when it meets the description. I certainly did not. I designed and wrote it as I went along (this is painfully evident to me in the lack of unity and consistency in the program, the "patchwork quilt" look).

This has also led to my big problem: because the program is so disorganized, I invariably come to some sort of dead end and drop the project. I completely gave up on **Air Hockey** many months back, but, at the urging of two of my friends (D.S. and D.B.), I picked it up again and trudged through the tangled code to finish it . . . finally. This is what has kept me from finishing the other hundred or so projects I have stored away in dusty disk files.

I'm sure that if it were not for this fact, there would be thousands more programs available for computers through other users, magazines and distributors. Next time you start to put something off because it seems too difficult, back up and try again.

Oh yes, the ducks.

The ducks? Well, I thought I'd try to be a little different from the other articles gracing this magazine's fine pages. (You wondered about them, didn't you?) Have a duck, you'll feel better. ⊞

Chris Page is an eighteen-year-old from San Diego, who's studying for an A.A. degree in electronics at I.T.T. Technical Institute. He has worked with computers for seven years and owned an Atari 800 for four. His primary computing interests are in sound, graphics and human interfacing.

<div align="center">

Listing 1.
Action! listing.

</div>

```
; ==================
; =  Air Hockey  =
; =      by      =
; =  Chris Page  =
; ==================

; Copyright (c) 1985 ANALOG Computing

; Special Thanks to:
; David Sullivan & David Becker

DEFINE
  OPTION="3",SELECT="5",START="6",
  NONE="7",LEFT="96",RIGHT="60",
  TOP="56",BOT="144"

BYTE
  NINDEX,VOLUME,FRICTION=[1],
  BOUNCE=[90],WIN=[10],PLAYERS=[2],
```

```
  HUE,LUM,OPT,PUCKXD,PUCKYD,HITFLAG,
  SERVER,GAMEOVER,SERVEIT,SDMCTL=559,
  CONSOL=53279,CHACT=755,WSYNC=54282,
  VCOUNT=54283,CRSINH=752,
  COLOR0=53270,COLOR1=53271,
  COLOR2=53272,COLOR3=53273,
  COLOR4=53274,RTCLOK=20,
  DMACTL=54272,LMARGN=82,RMARGN=83,
  CHBAS=756,PMBASE=54279,
  HITCLR=53278,P2PL=53262,
  GRACTL=53277,GPRIOR=623,
  RANDOM=53770,COLPM0=53266,
  COLPM1=53267,CH=764,RAMTOP=106,
  AUDCTL=53768,ATRACT=77,KEY

BYTE ARRAY
  DLIST,SCRMEM,RAMFONT,PMMEM($800),
  BAR(0)=[$FF$FF],
  PUCK(0)=[$60$F0$F0$F0$F0$F0$F0$60],
  TTOP(0)=['Q'R'R'R'R'R'R'S'S'S'S'R
          'R'R'R'R'R'E]
  TMID(0)=['R'T'T'T'T'T'T'T'T'T'T'T
          'T'T'T'T'T'R],
  TBOT(0)=['Z'R'R'R'R'R'R'S'S'S'S'R
          'R'R'R'R'R'C],
  NOTE(0)=[243 243 217 243 204 243
          193 204],
  YTOP(0)=[6 80],YBOT(0)=[62 144],
  SCORE(2),PDLX(2),PDLY(2),OSTIK(2),
  ROMSET($400)=$E000,HPOSP(4)=53248,
  HPOSM(4)=53252,PCOLR(4)=704

CARD
  PUCKXV,PUCKYV,PUCKX,PUCKY,
  MAXV=[500],DLISTL=560,SAVMSC=88,
  XITVBV=$E462

; --- Miscellaneous Procedures ---

PROC SETVBV=$E45C(BYTE CMD,VBIHI,
                       VBILO)


PROC VBI()
; VBI to play music
  SOUND(0,NOTE(NINDEX),10,VOLUME)
  SOUND(1,NOTE(NINDEX)-2,10,VOLUME)
  VOLUME==-1
  IF VOLUME=0 THEN
    VOLUME=15
    NINDEX==+1
    IF NINDEX=8 THEN
      NINDEX=0
    FI
  FI
; JMP XITVBV
  [$4C XITVBV]
RETURN


PROC INITVBI()
; Initialize music VBI
  NINDEX=0
  VOLUME=15
  SNDRST()
; set deferred vbi vector
  SETVBV(7,VBI RSH 8,VBI)
RETURN


PROC DEBOUNCE()
CARD I
; Debounce console keys
  FOR I=0 TO 5000 DO
    DO
      UNTIL CONSOL=NONE
    OD
```

```
    OD
RETURN

; --- Title Screen ---

PROC INITTITLE()
BYTE I
; Initialize title screen
    GRAPHICS(0)
    GPRIOR=17
    GRACTL=0
    SDMCTL=0
    CRSINH=1
    HUE=0
    DLIST=DLISTL
    DO
        UNTIL VCOUNT=0
    OD
    FOR I=3 TO 5 DO
        DLIST(I+7)=DLIST(I)
    OD
    SETBLOCK(DLIST,10,$70)
    FOR I=13 TO 25 STEP 2 DO
        DLIST(I)=0
    OD
    SETBLOCK(DLIST+27,2,$70)
    SETCOLOR(1,0,14)
    SETCOLOR(2,0,8)
    POSITION(11,0)
    PRINT("Air Hockey")
    POSITION(1,1)
    PRINT("By: Chris Page")
    POSITION(29,1)
    PRINT("Thanks: D.S. and D.B.")
    POSITION(17,2)
    PRINT(
    "June 30, 1984 - August 9, 1985")
    POSITION(7,4)
    PRINT("Copyright (c) 1984")
    POSITION(34,5)
    PRINT("Press  START")
    SDMCTL=33
RETURN


PROC TITLECOLORS()
BYTE I
; Mid-screen color changes
    HUE==+2
    IF HUE&2 THEN
        CHACT==+1&3
    FI
    FOR I=0 TO 30 DO
        DO
            WSYNC=0
            COLOR4=VCOUNT LSH 1+HUE
            IF VCOUNT=48 THEN
                COLOR1=0
            FI
            UNTIL VCOUNT&128
        OD
    OD
RETURN


PROC TITLE()
; Display title screen
    INITTITLE()
    INITVBI()
    DO
        TITLECOLORS()
        UNTIL CONSOL=START
    OD
    SDMCTL=0
RETURN
```

```
; --- Game Options ---

PROC INITOPTIONS()
; Initialize procedure OPTIONS()
    GRAPHICS(17)
    SDMCTL=0
    GRACTL=0
    DO
        UNTIL VCOUNT=0
    OD
    DEBOUNCE()
    SCRMEM=SAVMSC
    DLIST=DLISTL
    DLIST(3)==+1
    SETCOLOR(0,3,14)
    SETCOLOR(2,0,14)
    PRINTDE(6,"     GAME OPTIONS")
    POSITION(0,2)
    PRINTD(6,"OPTION - NEXT OPTION")
    PRINTDE(6,"SELECT - CHOOSE")
    PRINTDE(6," START - PLAY GAME")
    POSITION(3,6)
    PRINTD(6,"FRICTION: O")
    IF FRICTION THEN
        PRINTD(6,"N")
    ELSE
        PRINTD(6,"FF")
    FI
    POSITION(3,8)
    PRINTD(6,"VELOCITY: ")
    PRINTBD(6,MAXV/100)
    POSITION(3,10)
    PRINTD(6,"BOUNCE  : ")
    PRINTBD(6,BOUNCE/10)
    POSITION(3,12)
    PRINTD(6,"WIN AT  : ")
    PRINTBD(6,WIN)
    POSITION(3,14)
    PRINTD(6,"PLAYERS : ")
    PRINTBD(6,PLAYERS)
    SDMCTL=34
    OPT=0
RETURN


PROC OPTIONCOLORS(BYTE OPT)
; Mid-screen color changes
; OPT=option line to hi-light
    DO
        WSYNC=0
        UNTIL VCOUNT=15
    OD
    LUM=0
    WSYNC=0
    DO
        WSYNC=0
        COLOR0=LUM&$0F%$20
        LUM==+2
        UNTIL VCOUNT=25
    OD
    WSYNC=0
    COLOR0=0
    COLOR4=6
    DO
        WSYNC=0
        UNTIL VCOUNT=40
    OD
    COLOR0=$F8
    OPT==LSH 3+41
    DO
        WSYNC=0
        UNTIL VCOUNT=OPT
    OD
    COLOR0=$FE
    DO
        WSYNC=0
        UNTIL VCOUNT=OPT+8
```

```
      OD
      COLOR0=$F8
   RETURN


   PROC OPTIONS()
   CARD I
   ; Get game options from player(s)
      INITOPTIONS()
      DO
         FOR I=0 TO 10 DO
            OPTIONCOLORS(OPT)
            UNTIL CONSOL=START
         OD
         IF CONSOL=OPTION THEN
            OPT==+1
            IF OPT=5 THEN
               OPT=0
            FI
         FI
         IF CONSOL=SELECT THEN
            IF OPT=0 THEN
               FRICTION==!1
               IF FRICTION THEN
                  SCRMEM(134)=46
                  SCRMEM(135)=0
               ELSE
                  SCRMEM(134)=38
                  SCRMEM(135)=38
               FI
            ELSEIF OPT=1 THEN
               IF MAXV=900 THEN
                  SCRMEM(173)==-7
                  MAXV=200
               ELSE
                  SCRMEM(173)==+1
                  MAXV==+100
               FI
            ELSEIF OPT=2 THEN
               IF BOUNCE=90 THEN
                  SCRMEM(213)==-9
                  BOUNCE=0
               ELSE
                  SCRMEM(213)==+1
                  BOUNCE==+10
               FI
            ELSEIF OPT=3 THEN
               IF WIN=90 THEN
                  SCRMEM(253)==-8
                  WIN=10
               ELSE
                  SCRMEM(253)==+1
                  WIN==+10
               FI
            ELSE
               IF PLAYERS=2 THEN
                  SCRMEM(293)==-1
                  PLAYERS=1
               ELSE
                  SCRMEM(293)==+1
                  PLAYERS=2
               FI
            FI
         FI
         UNTIL CONSOL=START
      OD
      SDMCTL=0
      SNDRST()
   RETURN


   ; --- Play Air Hockey ---

   PROC MAKEFONT()
   BYTE I
   CARD J
   ; Change character set
```

```
   BYTE ARRAY
      CDAT(8)=[$55$55$55$55$54$54$50$40],
      EDAT(8)=[$40$50$54$54$55$55$55$55],
      QDAT(8)=[$01$05$15$15$55$55$55$55],
      RDAT(8)=[$55$55$55$55$55$55$55$55],
      SDAT(8)=[$FF$FF$FF$FF$FF$FF$FF$FF],
      TDAT(8)=[$AA$AA$2A$AA$AA$AA$A2$AA],
      ZDAT(8)=[$55$55$55$55$15$15$05$01]

      RAMFONT=(RAMTOP-8)*$100
      MOVEBLOCK(RAMFONT,ROMSET,$400)
      ZERO(RAMFONT+536,192)
      CHBAS=RAMTOP-8
      SDMCTL=61
      FOR I=0 TO 7 DO
         FOR J=0 TO 3000 DO OD
         RAMFONT(536+I)=CDAT(I)
         RAMFONT(552+I)=EDAT(I)
         RAMFONT(648+I)=QDAT(I)
         RAMFONT(656+I)=RDAT(I)
         RAMFONT(664+I)=SDAT(I)
         RAMFONT(672+I)=TDAT(I)
         RAMFONT(720+I)=ZDAT(I)
      OD
   RETURN


   PROC POSPLAYER(CARD PLAYER
                        BYTE X,Y,LENGTH
                        BYTE ARRAY SHAPE)
   ; Position Player
      HPOSP(PLAYER)=X+LEFT
      PLAYER==*$100+$400
      MOVEBLOCK(PMMEM+PLAYER+Y+TOP,
                    SHAPE,LENGTH)
   RETURN


   PROC POSPDL(BYTE PADDLE,X,Y)
   ; Position paddle
      POSPLAYER(PADDLE,X,Y,2,BAR)
   RETURN


   PROC POSPUCK(CARD X,Y)
   ; Position puck
      X==/100
      Y==/100
      POSPLAYER(2,X,Y,8,PUCK)
   RETURN


   PROC ERASEPDL(CARD PADDLE  BYTE Y)
   ; Erase paddle
      PADDLE==*$100+$400
      ZERO(PMMEM+PADDLE+Y+TOP,2)
   RETURN


   PROC ERASEPUCK(CARD Y)
   ; Erase puck
      Y==/100+TOP
      ZERO(PMMEM+$600+Y,8)
   RETURN


   PROC ERASEALL()
   ; Clear Player memory
      ERASEPDL(0,PDLY(0))
      ERASEPDL(1,PDLY(1))
      ERASEPUCK(PUCKY)
   RETURN


   PROC INITPMG()
   ; Initialize PMG
      PMMEM=(RAMTOP-16)*$100
      Zero(PMMEM,$800)
```

```
      PCOLR(0)=$76
      PCOLR(1)=$76
      PCOLR(2)=$36
      PMBASE=RAMTOP-16
      GRACTL=3
RETURN

PROC INITPLAY()
CARD I
; Initialize game
      GRAPHICS5(0)
      SDMCTL=0
      DO
         UNTIL VCOUNT=0
      OD
      SETVBV(7,$E4,$62)
      SNDRST()
      DEBOUNCE()
      INITPMG()
      SCRMEM=SAVMSC
      SCORE(0)=0
      SCORE(1)=0
      OSTIK(0)=15
      OSTIK(1)=15
      SERVER=0
      GAMEOVER=0
      CRSINH=1
      DLIST=DLISTL
      DLIST(2)=DLIST(3)+4
      DLIST(3)=DLIST(4)
      DLIST(4)=DLIST(5)
      DLIST(5)=$30
      DLIST(7)=$30
      SETBLOCK(DLIST+8,21,4)
      SETCOLOR(0,3,6)
      SETCOLOR(1,0,14)
      SETCOLOR(2,0,4)
      SETCOLOR(3,2,14)
      SETCOLOR(4,0,6)
      POSITION(3,0)
      PRINTE("air hockey")
      SAVMSC==+16
      POSITION(0,0)
      PRINTF(
      " One : 00 | Win : %B | Two : 00",
      win)
      MOVEBLOCK(SCRMEM+55,TTOP,18)
      FOR I=87 TO 663 STEP 32 DO
         MOVEBLOCK(SCRMEM+I,TMID,18)
      OD
      MOVEBLOCK(SCRMEM+695,TBOT,18)
      MAKEFONT()
      SOUND(3,0,0,3)
      KEY=0
      CH=$FF
RETURN

PROC SERVE(BYTE PLAYER)
CARD I
; Initialize positions
      ERASEALL()
      PDLX(0)=28
      PDLX(1)=28
      PDLY(0)=YTOP(0)
      PDLY(1)=YBOT(1)
      PUCKX=3000
      PUCKY=4000+6800*PLAYER
      PUCKXV=0
      PUCKYV=0
      POSPDL(0,PDLX(0),PDLY(0))
      POSPDL(1,PDLX(1),PDLY(1))
      POSPUCK(PUCKX,PUCKY)
      HITCLR=0
      HITFLAG=0
      VOLUME=0
RETURN


PROC MOVEPADDLE(BYTE P)
BYTE STIK
; Move paddle
      ERASEPDL(P,PDLY(P))
      STIK=STICK(P)
      ; move puck 2 for one player game
      IF PLAYERS=P THEN
         STIK=$F
         IF PDLX(1)+2<PUCKX/100 THEN
            STIK==-8
         ELSE
            STIK==-4
         FI
         IF PDLY(1)-6<PUCKY/100 THEN
            STIK==-2
         ELSEIF PDLY(1)-8>PUCKY/100 THEN
            STIK==-1
         ELSE
            STIK==-2
            IF RAND(2) THEN
               STIK==+1
            FI
         FI
      FI
      ; save stick position
      OSTIK(P)=STIK
      ; move paddle verticaly
      IF (STIK&1)=0 THEN
         PDLY(P)==-2
         IF PDLY(P)<YTOP(P) THEN
            PDLY(P)=YTOP(P)
         FI
      ELSEIF (STIK&2)=0 THEN
         PDLY(P)==+2
         IF PDLY(P)>YBOT(P) THEN
            PDLY(P)=YBOT(P)
         FI
      FI
      ; move paddle horizontaly
      IF (STIK&8)=0 THEN
         PDLX(P)==+2
         IF PDLX(P)>RIGHT-4 THEN
            PDLX(P)=RIGHT-4
         FI
      ELSEIF (STIK&4)=0 THEN
         PDLX(P)==-2
         IF PDLX(P)>240 THEN
            PDLX(P)=0
         FI
      FI
      POSPDL(P,PDLX(P),PDLY(P))
RETURN


PROC REVERSEPX()
; Reverse horizontal puck direction
      VOLUME=14
      PUCKXD==!1
      IF PUCKXV<(90-BOUNCE) THEN
         PUCKXV=0
      ELSE
         PUCKXV==-(90-BOUNCE)
      FI
RETURN


PROC REVERSEPY()
; Reverse vertical puck direction
      VOLUME=14
      PUCKYD==!1
      IF PUCKYV<(90-BOUNCE) THEN
         PUCKYV=0
      ELSE
         PUCKYV==-(90-BOUNCE)
      FI
RETURN
```

```
PROC MOVEPUCK()
BYTE PADDLE,XDIF,YDIF,STIK,ABOVE
CARD ARRAY
  XVELOC(0)=[400 140 100  80  40   0
              40  80 100 140 400]
; Move the puck
  ERASEPUCK(PUCKY)
; check for paddle collisions
  PADDLE=0
  IF PUCKY/100>70 THEN
    PADDLE=1
  FI
  STIK=OSTIK(PADDLE)
  IF P2PL THEN
    IF HITFLAG=0 THEN
      VOLUME=14
    ; new x velocity & direction
      XDIF=PUCKX/100+3-PDLX(PADDLE)
      PUCKXV=XVELOC(XDIF)
      PUCKXD=0
      IF XDIF>5 THEN
        PUCKXD=1
      FI
    ; new y velocity & direction
      YDIF=PUCKY/100-PDLY(PADDLE)
      ABOVE=0
      IF PADDLE THEN
        IF PUCKY/100<PDLY(1) THEN
          ABOVE=1
        FI
      ELSE
        IF PUCKY/100+8<PDLY(0) THEN
          ABOVE=1
        FI
      FI
    ; paddle not moving
      IF (STIK&3)=3 THEN
        PUCKYD==!1
    ; puck not moving
      ELSEIF PUCKYV=0 THEN
        PUCKYV=200
        PUCKYD=0
        IF (STIK&3)=1 THEN
          PUCKYD=1
        FI
    ; puck and paddle equal y coord
      ELSEIF PUCKY/100+3=PDLY(PADDLE)
      THEN
      ; do nothing
      ELSE
      ; moving puck and paddle down
        IF PUCKYD=1 AND (STIK&3)=1
        THEN
          IF ABOVE THEN
            PUCKYV==-200
            IF PUCKYV>200 THEN
              PUCKYD==!1
            FI
          ELSE
            PUCKYV==+200
          FI
        ELSEIF PUCKYD=0 AND
               (STIK&3)=2 THEN
          IF ABOVE=0 THEN
            PUCKYV==-200
            IF PUCKYV>200 THEN
              PUCKYD==!1
            FI
          ELSE
            PUCKYV==+200
          FI
        ELSEIF PUCKYD=1 AND
               (STIK&3)=2 THEN
          IF ABOVE THEN
            PUCKYD==!1
            PUCKYV==+200
          FI
        ELSEIF PUCKYD=0 AND
```

```
               (STIK&3)=1 THEN
          IF ABOVE=0 THEN
            PUCKYD==!1
            PUCKYV==+200
          FI
        FI
      FI
      HITFLAG=1
    ELSE
      HITFLAG=0
    FI
    HITCLR=0
  ; move horizontaly
    IF PUCKXV>MAXV THEN
      PUCKXV=MAXV
    FI
    IF PUCKXD THEN
      PUCKX==+PUCKXV
    ELSE
      PUCKX==-PUCKXV
    FI
  ; check boundaries
    IF PUCKX>24000 THEN
      REVERSEPX()
      PUCKX=0
    ELSEIF PUCKX>RIGHT*100 THEN
      REVERSEPX()
      PUCKX=RIGHT*100
    FI
    IF PUCKYV>MAXV THEN
      PUCKYV=MAXV
    FI
  ; move verticaly
    IF PUCKYD THEN
      PUCKY==+PUCKYV
    ELSE
      PUCKY==-PUCKYV
    FI
  ; check boundaries
    IF PUCKY>24000 THEN
      REVERSEPY()
      PUCKY=0
    ELSEIF PUCKY>BOT*100 THEN
      REVERSEPY()
      PUCKY=BOT*100
    FI
  ; handle friction
    IF PUCKXV THEN
      PUCKXV==-FRICTION
    FI
    IF PUCKYV THEN
      PUCKYV==-FRICTION
    FI
  ; fading collision sound
    IF VOLUME THEN
      VOLUME==-2
      SOUND(0,10,8,VOLUME)
      SOUND(1,10,10,VOLUME)
    ELSE
      SOUND(0,0,0,0)
      SOUND(1,0,0,0)
    FI
    POSPUCK(PUCKX,PUCKY)
RETURN

PROC GOAL(BYTE PLAYER)
BYTE I
CARD J
; Inc score, check for a winner
  SNDRST()
  ERASEPUCK(PUCKY)
  VOLUME=0
  SERVEIT=1
  SERVER=PLAYER
  SCORE(PLAYER)==+1
  IF SCORE(PLAYER)=WIN THEN
```

```
      GAMEOVER=1
    FI
  ; flash score
    FOR I=0 TO 5 DO
      SETBLOCK(SCRMEM+23+22*PLAYER,2,0)
      FOR J=0 TO 5000 DO OD
      SCRMEM(23+22*PLAYER)=
             16+SCORE(PLAYER)/10
      SCRMEM(24+22*PLAYER)=
             16+SCORE(PLAYER) MOD 10
      SOUND(0,20,10,8)
      FOR J=0 TO 5000 DO OD
      SOUND(0,0,0,0)
    OD
  ; cheering
    IF GAMEOVER=0 THEN
      FOR I=0 TO 30 DO
        FOR J=0 TO 1000 DO OD
        SOUND(0,10,8,I RSH 1)
      OD
      FOR J=0 TO 40000 DO OD
      FOR I=0 TO 30 DO
        FOR J=0 TO 1000 DO OD
        SOUND(0,10,8,15-I RSH 1)
      OD
    FI
    SNDRST()
    SOUND(3,0,0,3)
  RETURN


  PROC MOVEALL()
  ; Move paddles and puck
  ; keep attract mode at bay
    ATRACT=0
  ; check for goal
    IF PUCKX>2400 AND PUCKX<3700 THEN
      IF PUCKY=0 THEN
        GOAL(1)
      ELSEIF PUCKY=BOT*100 THEN
        GOAL(0)
      FI
    FI
    IF GAMEOVER=0 THEN
      MOVEPUCK()
      MOVEPADDLE(0)
      MOVEPADDLE(1)
    FI
  RETURN


  PROC ENDGAME()
  BYTE I
  CARD J,K
  ; Cheer profusly and end game
    SNDRST()
    FOR I=0 TO 30 DO
      FOR J=0 TO 1000 DO OD
      SOUND(0,10,8,I RSH 1)
    OD
    FOR J=0 TO 200 DO
      FOR K=0 TO 500 DO OD
      IF RAND(130)=0 THEN
        FOR I=0 TO 15 DO
          FOR K=0 TO 1200 DO OD
          SOUND(1,30-I,10,I)
        OD
        FOR I=0 TO 15 DO
          FOR K=0 TO 1200 DO OD
          SOUND(1,15+I,10,15-I)
        OD
      FI
    OD
    FOR I=0 TO 30 DO
      FOR J=0 TO 1000 DO OD
      SOUND(0,10,8,15-I RSH 1)
    OD
    FOR J=0 TO 40000 DO OD
```

```
  RETURN


  PROC PLAY()
  ; Play Air Hockey
    INITPLAY()
    SERVE(SERVER)
    DO
      DO
        UNTIL VCOUNT=100
      OD
      IF CH<>$FF THEN
        KEY=GETD(1)
      FI
      IF KEY=32 OR SERVEIT=1 THEN
        SERVE(SERVER)
        KEY=0
        CH=$FF
        SERVEIT=0
      FI
      MOVEALL()
      UNTIL GAMEOVER=1 OR KEY=27 OR
            CONSOL=6
    OD
    IF KEY<>27 AND CONSOL<>6 THEN
      ENDGAME()
    FI
    SNDRST()
  RETURN

  ; --- Main Procedure ---

  PROC MAIN()
    LMARGN=0
    CLOSE(1)
    OPEN(1,"K:",4,0)
    DO
      TITLE()
      WHILE CONSOL=6 DO
        OPTIONS()
        PLAY()
      OD
      UNTIL KEY=27
    OD
    CLOSE(1)
    GRAPHICS(0)
    GRACTL=0
  RETURN
  •
```