R.O.T.O.

At 25, Mike Stortz is the P.D. Librarian for GRASP, the Richmond! Virginia Atari users group. Seemingly unable to find work in the programming field, he is working on about thirty projects at once, including a graphic arcade/adventure game.

1996: Kara Hyke leaves show business after losing out in a disputed Oscar award.
1997: Kara Hyke enrolls at M.I.T.
2010: Kara Hyke discovers semi-matter.
2016: Hyke-Grey effect discovered.
2030: Hyke and Grey found Arcadia on Proxima III.
2065: Hamner scout ship attacks Arcadia and is driven off after extensive damage to the city. Work on the shield begins.
Today: Hamner fleet attacks Arcadia...

Deep in caverns under Arcadia, fuel cannisters of semi-matter ore to power the city’s defensive shield have been cached away against the day when the aliens would return. Composed of mixed-charge matter, hyke (as it is called) is easily persuaded to annihilate itself in the manner of matter-antimatter reactions, but is more easily stored.

Now that the attack has begun, brave volunteer retrievers must don helipacks, fly down into the caves and bring up the ore, so that Arcadia may not fall.

R.O.T.O. is a game of coordination, reflexes, timing and a hint of strategy that should keep you going for a while before Arcadia can claim victory.

Plug in your Action! cartridge and type in Listing 1. Action! is forgiving about case, spacing and line divisions, so you need not slavishly follow the format of the listing (which is compressed somewhat for purposes of publication).

SAVE it before attempting to RUN it! If you try to run R.O.T.O. from memory, the source code will be overwritten and ruined, causing much gnashing of teeth. Because R.O.T.O. is so large, it must be compiled off of disk or cassette.

For disk.

After typing R.O.T.O. in, save it with the command CTRL-W and type in the filename "ROTO.ACT" then RETURN. SHIFT-CLEAR the editor, enter the monitor with the CTRL-M command, enter C "ROTO.ACT" and RETURN. This will compile R.O.T.O. into machine language. Now, save the compiled version by typing W "ROTO.AML" and RETURN. AML stands for Action! machine language. To run R.O.T.O. now, just type R and RETURN. In the future, simply type CTRL-M to enter the monitor, D and RETURN to go to DOS, and then binary loading the file ROTO.AML from DOS by using the L command in DOS 2 or DOS 3?or type LOAD ROTO.AML if you have DOS-XL.
For cassette.

Type in R.O.T.O. and save it with the CTRL-W command. Do not use the "Screen Off?" option; it will upset the tape timing. Rewind your tape, press PLAY and RECORD, give the filename 'C:' and RETURN.

Go have lunch while the source code's being saved (about fifteen minutes). Come back, clear memory with the SHIFT CLEAR, and enter the monitor with the CTRL-M command.

Rewind the tape containing the R.O.T.O. source code, press PLAY, type in R "C:" and RETURN. Have some more lunch. The computer is rereading the source code, compiling it as it does. When it's finished compiling, the game will automatically start.

In the future, you may play R.O.T.O. by inserting the tape containing the source code into the recorder and typing the R "C:" command.

Playing R.O.T.O.

After beginning, you should see the R.O.T.O. logo and your man flying about it, while an explanatory message scrolls beneath. You may begin by pressing the START button or the fire button on joystick 1. You will see a portion of a cavernous network and four green blocks with Hs on them. This is the fuel intake.

Cannisters of hyke are scattered about the caves. They look like the fuel intake, except that they're glowing. Pick up these cannisters by touching them, then return to the intake and touch it. This advances your score and charges the shield in accordance with however many cannisters you've deposited.

Each cannister is worth fifty points. Returning ore also refuels your helipack. Picking up more than ten cannisters before depositing them will cause their magnetic fields to interact with explosive results.

Your man moves up, down, left, right and diagonally in all directions, although he moves downwards faster than up. Moving against the screen's border will scroll more caverns into view, although you'll automatically stop at the far ends of the caves.

Don't run into a wall, or you'll lose a helipack. Watch your fuel. too. Running out will produce the expected effect.

Your retriever is also equipped with a molecular debonding device to facilitate going through rock. Press the fire button to let off a shot. The debonder will vaporize any chunk of rock you fire on, but you will lose a point for even' piece of the caves you eliminate (because you're reducing their structural integrity).

Unfortunately, the debonder will also affect a fuel cannister. Rupturing the magnetic bottle containing the hyke will produce a large explosion and prevent anyone else from retrieving fuel.

Also, attacks from the alien fleet will shake the caves periodically, causing rocks to break loose from the ceiling. Shooting rocks is worth one point each (for cleaning up). Don't run into them, and be careful that the tremors don't send you into a wall.

Pressing any key while a game is in progress will pause it. Press another key to resume play. If you want to begin again, press START.
If you play well and retrieve enough cannisters to top 1000 points, the shield will have stayed up long enough for reinforcements to arrive—and the city will be saved.

On the other hand, if you wreck while hauling cannisters, you could deplete the ore supply so that victory is impossible.

Remember which portions of the caverns you've mined out, and definitely recall the way back to the fuel intake. The cave network is generated randomly each game, so expect variety.

R.O.T.O. may end in five ways:

1. Most frequently, you run out of helipacks (while there are many volunteers, there are only three of the sophisticated flying apparatuses).
2. The shield is battered down. This happens when you don't retrieve enough ore.
3. A fuel cannister is shot.
4. You carry more than ten cannisters at once.
5. The least common ... Arcadia holds out, and the aliens are defeated? this time!

That's it!#

Action! deserves a word of praise here. R.O.T.O. was designed half in advance and half as I thought of another feature to put in. The excellent editor made even major reshuffling and splitting of routines easy.

I shudder to think what I would have gone through using a conventional assembler. Even when the source code became too large to co-reside with the object code, I could compile off of my Axlon RAMDisk with little loss of development time.

The author would appreciate any letters of business offers, extravagant praise, or, failing that, constructive comment. Have fun and save Arcadia!

; R.O.T.O.  by Mike Stortz
SET $000E=$4000
SET $0491=$4000
DEFINE bytes="64",lines="80",
    rock="194",pmb_page="128",
    cb_page="128",cb_adr="32768",
    dl_page="136",dl_adr="34816",
    misc_page="137",misc_adr="35072",
    sc_page="140",sc_adr="35840"
BYTE
   rtclok=$14,atract=$4D,lmargin=$52,
   rowcrs=$54,dindex=$57,sdmctl=$22F,
   gprior=$26F,crsinh=$2F0,
   ch=$2FC,gractl=$D01D,
   hitclr=$D01E,consol=$D01F,
   audctl=$D208,skstat=$D20F,
   pmbase=$D407,wsync=$D40A,
   vcount=$D40B,nmien=$D40E,
   chbas=$2F4,chbase=$D409,
   hscrol=$D404,vscrol=$D405,
   pcolr0=$2C0,pcolr1=$2C1,
   pcolr2=$2C2,pcolr3=$2C3,
   colpm0=$D012,colpm1=$D013,
   colpm2=$D014,colpm3=$D015,
BYTE ARRAY hposp=$D000,mxpf=$D000,
hposm=$D004,pxpf=$D004

CARD colcrs=$55,savmsc=$58,
vdslst=$200,sdslst=$230,
txtmsc=$294

BYTE i,j,k,l,cx,cy,x,y,xs,xsm,ys,joy,
phase,mc,face,flag,bak,fore,
fuel,packs,enable,whine,
carried,end,cans,fallc,shake,
shakec

BYTE ARRAY logo=
[85 85 88 88 88 88 88 85 85
128 96 88 88 88 88 88 96 128
0 5 21 88 88 88 88 88 88
0 128 96 88 88 88 88 88 88
0 149 149 2 2 2 2 2
0 85 85 80 80 80 80 80 80
2 9 37 37 37 37 37 37 37
80 84 37 37 37 37 37 37 37
85 89 88 88 88 88 88 0 0
128 96 88 88 88 88 88 88 1
88 88 88 88 88 88 88 21 5
88 88 88 88 88 88 88 96 128
2 2 2 2 2 2 2 2 2
80 80 80 80 80 80 80 80 80
37 37 37 37 37 9 2 64
37 37 37 37 37 84 80 0 ],

cset=
[ 0 124 198 198 198 198 198 254 124
0 56 120 120 56 56 124 254
0 124 206 28 56 112 254 254
0 254 28 56 28 206 254 124
0 28 60 124 220 254 28 28
0 254 192 252 14 206 254 124
0 124 192 252 206 206 254 124
0 254 14 28 56 112 112 112
0 124 198 124 198 198 254 124
0 124 206 126 14 30 124 120
0 56 124 206 206 222 206 206
0 252 14 252 206 206 254 252
0 124 254 198 192 198 254 124
0 248 220 206 206 222 220 216
0 126 252 0 240 192 252 126
0 126 252 0 252 248 224 224
0 126 224 224 238 230 254 126
0 230 230 230 230 230 230 230]
man0=
[ 0 254 16 16 28 24 28 20
24 36 42 46 52 0 0 8
8 16 16 0 0
0 127 8 8 56 24 56 40
24 36 84 116 44 0 0 16
16 8 8 0 0],

man1=
[ 0 254 16 16 16 20 20 20
16 48 49 48 48 8 24 16
16 32 32 32 32
0 127 8 8 8 40 40 40
8 12 140 12 12 16 24 8
8 4 4 4 4],

rotor=[254 124 56 16 56 124
127 62 28 8 28 62],

can=[24 60 126 90 195 195 219 255],

dldata=
[ 112 112 112 68 0 misc_page
4 6 11 6 139 48],
PROC Vblank()

    [$48] ; PHA
    vscrol=ys  hscrol=xs+xsm
    [$68] ; PLA
    [$4C $62 $E4] ; JMP XITVBV
    RETURN

PROC Dli()

BYTE d

    [$48
      $8A $48 $98 $48 $A5 $AC $48
      $A5 $AE $48 $A5 $AF $48 ]

; PHA
; TXA PHA TYA PHA LDA $AC PHA
; LDA $AE PHA PHA LDA $AF PHA

wsync=0
IF vcount<64 THEN
    chbase=chbas+2
    colpf0=fore
    colpf4=bak
    IF enable=1 THEN
        colpf1=202
        colpf2=(rtclok RSH 1)&$3A
        colpf3=6
    ELSE
        colpf1=0
        colpf2=0
        colpf3=0
    FI
ELSE
    chbase=chbas
    colpf1=202
    colpf4=64
FI
PROC Wait(BYTE w)
BYTE w1

w1=rtclok
DO until rtclok=W1+w OD
RETURN

PROC PmSet()

sdmctl=62 gractl=3 hitclr=0
pcolr0=152 pcolr1=118 gprior=33
pmbase=pmb_page chbas=cb_page
RETURN

PROC ZeroOut()

Zero(missile,1280)
FOR i=0 TO 3 DO
  hposp(i)=0 hposm(i)=0
OD
SndRst()
RETURN

PROC DoPhase()

phase==+1
IF phase=6 THEN phase=0 FI
RETURN

PROC DoScore()

IF score>1000 THEN end=5 FI
dindex=2
rowcrs=4 colcrs=14 PrintBD(6,packs)
rowcrs=6 colcrs=14 PrintD(6,""")
rowcrs=6 colcrs=14 PrintID(6,score)

dindex=6
Plot(fuel,5) Plot(shield,7)
RETURN

PROC ChargeShield()
IF carried<>0 THEN

    shield==+carried LSH 2
    score==+carried*50 DoScore()
    SndRst()
    FOR i=1 TO 250 step 10 DO
        Sound(3,250-i,10,6)
        Wait(1)
    OD
    Sound(3,0,0,0)
    carried=0 fuel=50 whine=0
    color=1
    Plot(0,5) DrawTo(fuel,5)
    Plot(0,7) DrawTo(shield,7)
    color=0
FI
hitclr=0
RETURN

PROC CheckShake()

    IF Rand(0)=255 AND
       Rand(5)=0 AND
       shake=0 THEN

        shake=Rand(10)+10
        shield==-Rand(20)
        IF shield<0 THEN shield=0 FI
        IF shield=0 THEN end=2 FI
        Plot(159,7) DrawTo(shield,7)
    FI
    IF shakec<>0 THEN
        shakec==-1
    ELSE
        shakec=60
        IF shakec<>0 THEN
            shakec=-1
            j=Rand(10)
            IF fall(j)=0 THEN
                a=table(cy)+cx+Rand(20)
                IF screen(a)=0 THEN
                    fall(j)=a id(j)=rock
                    screen(a)=rock
                FI
            FI
            Sound(2,255-shake,2,6)
            xsm=Rand(5)
        ELSE
            xsm=0
            Sound(2,0,0,0)
        FI
    FI
RETURN

PROC CheckFuel()
IF (rtcloc=0 or rtcloc=128) AND fuel<>0 THEN
  DoScore() fuel=-1
FI
RETURN

PROC EndGame()

Zero(Misc_adr+80,80) ZeroOut()

dindex=2 rowcrs=4 colcrs=0

IF end=1 THEN
  PrintDE(6," NO PACKS LEFT")
ELSEIF end=2 THEN
  PrintDE(6," SHIELD DEPLETED")
ELSEIF end=3 THEN
  PrintDE(6," CANNISTER RUPTURED")
ELSEIF end=4 THEN
  PrintD(6," TOO MANY CANNISTERS")
ELSEIF end=5 THEN
  PrintDE(6," ARCADIA THANKS YOU")
FI
PutDE(6) PrintDE(6,"      game over")

FOR a=1 TO 400 DO
  Sound(0,a RSH 1,8,6)
  DO UNTIL vcount=128 OD
  FOR i=0 TO 60 DO
    colpf0=vcount+rtcloc
    wsync=0
  OD
OD
RETURN

PROC GetDir()

  joy=15!Stick(0) xd=0 yd=0
  IF (joy&8)<>0 THEN xd=1 FI
  IF (joy&4)<>0 THEN xd=-1 FI
  IF (joy&2)<>0 THEN yd=2 FI
  IF (joy&1)<>0 THEN yd=-1 FI
  IF xd<>0 or yd<>0 THEN
    oxd=xd oyd=yd
  FI
RETURN

PROC Scroll()

  IF (joy&4)<>0 THEN xs==+1 x==+1
  IF xs=8 THEN
    IF cx=0 THEN xs=-1
    ELSE cx=-1 xs=0
  FI
  FI
ELSEIF (joy&8)<>0 THEN xs=-1 x=-1
IF xs=255 THEN
  IF cx=44 THEN xs==+1
  ELSE cx==+1 xs=7
  FI
FI
FI
IF (joy&2)<0 THEN ys==+1 y==2
IF ys=8 THEN
  IF cy=68 THEN ys==1
  ELSE cy==+1 ys=0
  FI
FI
ELSEIF (joy&1)<0 THEN ys==1 y==1
IF ys=255 THEN
  IF cy=0 THEN ys==1
  ELSE cy==-1 ys=7
  FI
FI
FI
DO UNTIL vcount=128 OD

ary=@a a=screen+table(cy)+cx j=12
FOR i=0 TO 17 DO
  dlist(j+1)=ary(0)
  dlist(j+2)=ary(1)
  j==+3
  a==+bytes
OD
RETURN

PROC MoveMan()

  Zero(pmb+y,26) Zero(pmb+256+y,26)
  x==+xd y==+yd
  hposp(0)=x hposp(1)=x
  IF xd>0 THEN face=0
  ELSEIF xd<0 THEN face=1 FI
  a=pmb+y+(phase RSH 2)
  temp=manadr(face)
  MoveBlock(a,man0+temp,21)
  MoveBlock(a+256,man1+temp,21)

  a=pmb+y+1
  a==+phase RSH 2
  i=rotor(rotoradr(face)+phase)
  Poke(a,i) Poke(a+256,i)

  Sound(0,phase LSH 2-(yd LSH 3),8,2)
RETURN

PROC GoBoom()

  SndRst()
  Zero(missile,256) mx(0)=0 mx(1)=0
  Wait(30)
ary=pmb+y

FOR i=0 TO 170 DO
    FOR j=1 to 20 DO
        colpm0=64+Rand(8) LSH 1
        colpm1=64+Rand(8) LSH 1
        wsync=0
    OD
    k=Rand(24) ary(k)==&Rand(0)
    k=Rand(24) ary(k+256)==&Rand(0)
    Sound(1,i,4,6)
    Wait(1)
OD
Zero(pmb,512)
SndRst() pcolr0=152 pcolr1=118
Wait(20) enable=0
FOR i=0 TO 14 step 2 DO
    fore=46-i
    Wait(5)
OD
fore=0 Wait(60) hitclr=0
carried=0 whine=0 shake=0 face=0
packs=-1
IF packs=0 THEN end=1 FI
FOR i=0 TO 19 DO
    screen(fall(i))=0
    fall(i)=0
OD

fuel=50 color=1
Plot(0,5) DrawTo(fuel,5)
color=0

x=84 y=110 cx=0 cy=0 xs=7 ys=0
DoScore() Scroll() MoveMan()
fore=36 enable=1
RETURN

PROC GetCan()

i=x-35 j=y-50
i==RSH 3 j==RSH 3
a=table(j+cy)+i+cx
IF screen(a)=159 THEN
    screen(a)=0
    carried==+1
    IF carried=11 THEN end=4 FI
    whine=200 hitclr=0
FI
RETURN

PROC Falling(CARD bb)

j=screen(bb-64)
IF j=159 OR j=rock THEN
    FOR k=10 TO 19 DO
        IF fall(k)=0 THEN
            fall(k)=1
            whine=200 hitclr=0
        FI
    OD
PROC ZapIt(BYTE zz)

attract=0
l=mxd(zz)+2
j=mx(zz)-31-l RSH 2-xs
k=my(zz)-72+ys
j==RSH 3  k==RSH 3

missile(my(zz))==$&255-mdata(zz)
mx(zz)=0

a=table(cy+k)+cx+j
IF screen(a)=159 THEN end=3 FI
IF screen(a)=rock THEN score==$+2 FI
bak=70 fore=12
FOR j=0 TO 10 DO
  screen(a)=65
  FOR k=1 TO 100 DO OD
  screen(a)=0
  FOR k=1 TO 100 DO OD
  Sound(1,200,2,15-j)
OD
bak=0 fore=36 screen(a)=0
Sound(1,0,0,0)
hitclr=0 score=-1 DoScore()
Falling(a)
RETURN

PROC Bump()

i=pxpf(0)  j=pxpf(1)

IF (i&1)<<0 OR (j&1)<<0 OR
  (i&8)<<0 OR (j&8)<<0 THEN
  GoBoom()
ELSEIF (i&2)<<0 OR (j&2)<<0 THEN
  ChargeShield()
ELSEIF (i&4)<<0 OR (j&4)<<0 THEN
  GetCan()
FI
IF mxpf(0)<<0 THEN ZapIt(0) FI
IF mxpf(1)<<0 THEN ZapIt(1) FI
RETURN
PROC StartMiss()

IF Strig(0)=0 AND flag=0 THEN
    flag=1 mc==!1
    IF mx(mc)=0 THEN
        missile(my(mc))==&(255!mdata(mc))
        my(mc)=y+10
        missile(my(mc))==%mdata(mc)
        mx(mc)=(x+4+face RSH 3)&254
        mxd(mc)=face LSH 2-2
    FI
FI
flag=Strig(0)!1
RETURN

PROC MoveMiss()

j=2
FOR i=0 TO 1 DO
    temp=mx(i)
    IF temp<>0 THEN
        temp=-mxd(i) hposm(i)=temp
        IF x>temp THEN
            k=x-temp
        ELSE
            k=temp-x
        FI
        Sound(1,k,12,8)
    ELSE
        j==-1
    FI
    mx(i)=temp
    IF j=0 THEN Sound(1,0,0,0) FI
OD
RETURN

PROC MoveRocks()

FOR i=0 TO 19 DO
    temp=fall(i)
    IF temp<>0 THEN
        IF screen(temp)=0 THEN
            temp=0
        ELSE
            a=temp+64
            IF screen(a)<>0 THEN
                temp=0
                IF id(i)=159 THEN end=3 FI
            ELSE
                screen(temp)=0
                screen(a)=id(i)
                temp==+64
            FI
        FI
    FI
fall(i)=temp
PROC CheckRocks()
    fallc=-1
    IF fallc=0 THEN
        fallc=20 MoveRocks()
    FI
    RETURN

PROC DrawWall(CARD st BYTE cc, in, len)
    BYTE ii, jj
    CARD tt

    screen(st)=1
    tt=st+in
    FOR ii=1 TO len-2 DO
        jj=Rand(2)
        screen(tt)=cc+jj
        tt=tt+in
    OD
    screen(tt)=1
    RETURN

PROC DrawCaves()
    sdmctl=0 sdstlst=dlist
    Zero(sc_adr,5120)
    Zero(misc_adr,512)

    FOR i=0 TO 11 DO
        dlist(i)=dldata(i)
    OD

    a=screen j=12
    FOR i=0 TO 17 DO
        dlist(j)=64+32+16+6
        dlist(j+1)=a&$FF
        dlist(j+2)=a RSH 8
        j==+3
        a==+bytes
    OD
    dlist(j-3)=128+64+16+6

    FOR i=0 to 7 DO
        dlist(j+i)=dldata2(i)
    OD

    txtmsc=misc_adr
    FOR i=0 TO 7 DO
        Poke(misc_adr+17+i,79+i)
        Poke(misc_adr+57+i,87+i)
    OD
    rowcrs=2  colcrs=0
    PrintE("fuel packs:")
color=0
PrintE("shield score:")
PrintE(" by mike stortz")

dindex=6 color=1
Plot(0,5) DrawTo(fuel,5)
Plot(0,7) DrawTo(shield,7)

a=0
FOR i=0 TO 7 DO
  FOR j=0 TO 15 DO
    k=Rand(32)
    IF (k&16)<0 THEN k==%4 FI
    IF (k&1)<0 THEN
      DrawWall(a,3,1,4) FI
    IF (k&2)<0 THEN
      DrawWall(a+3,5,bytes,10) FI
    IF (k&4)<0 THEN
      DrawWall(a+576,7,1,4)
      IF Rand(5)=0 THEN
        screen(a+514)=rock
      FI
    IF (k&8)<0 THEN
      DrawWall(a,9,bytes,10) FI
    IF (k&16)<0 AND j>0 AND j<15 THEN
      screen(a+513)=159 FI
    a==+4
  OD
  a==+576
OD

FOR a=8 TO 68 STEP 10 DO
  i=(Rand(14)+1) LSH 2
  FOR j=i+1 TO i+2 DO
    screen(table(a)+j)=0
    screen(table(a+1)+j)=0
    screen(table(a+2)+j)=0
  OD
OD

screen(69)=95 screen(70)=95
screen(133)=95 screen(134)=95
PmSet()
RETURN

PROC Title()
BYTE t

Graphics(21)
PmSet() ZeroOut()
Zero(missile,1280)
Zero(misc adr,3000)
screen=savmsc dlist=sdslst
colr0=150 colr1=146
FOR i=6 TO 13 DO
    FOR j=8 TO 15 DO
        screen(j*20+i)=logo(k)
        k+=1
    OD
OD
FOR i=6 TO 13 DO
    FOR j=16 TO 23 DO
        screen(j*20+i)=logo(k)
        k+=1
    OD
OD

dlist(31)=32
dlist(32)=64+32+6
dlist(33)=0
dlist(34)=misc_page
b=misc_adr

FOR i=35 TO 43 DO
    dlist(i)=32+6
OD
dlist(44)=6
FOR i=45 TO 52 DO
    dlist(i)=0
OD
dindex=0 lmargin=1
dlist(10)=6
savmsc==+100
PrintD(6," last ") PrintID(6,score)
colcrs=10
PrintD(6,"high ") PrintCD(6,high)

savmsc=misc_adr+300
rowcrs=0 colcrs=1
PrintDE(6," reserve ore ")
PrintDE(6,"transport operation")
PrintDE(6," THE CITY OF ")
PrintDE(6,"ARCADE IS UNDER ")
PrintDE(6,"ATTACK. YOUR JOB ")
PrintDE(6,"IS TO RECOVER FUEL")
PrintDE(6,"CANNISTERS OF HYKE")
PrintDE(6,"AND RETURN THEM TO")
PrintDE(6,"THE UPPER LEFT END")
PrintDE(6,"OF THE Caverns. IF")

savmsc==+400 rowcrs=0 colcrs=1
PrintDE(6,"YOUR SCORE EXCEEDS")
PrintDE(6,"1000, ARCAIDE HAS ")
PrintDE(6,"HELD OUT LONG ")
PrintDE(6,"ENOUGH FOR HELP TO")
PrintDE(6,"ARRIVE. DON'T ")
PrintDE(6,"SHOOT A CANNISTER ")
PrintDE(6,"OR CARRY MORE THAN")
10 AT A TIME, AND 
DON'T RUN INTO A 
WALL. GOOD LUCK! 

savmsc==+400 rowcrs=0 colcrs=1
PutDE(6)
PRINTDE(6,"          press START")
PRINTDE(6," to play")

x=86 y=58 yd=-1 xd=0 ys=0 xs=0
phase=0 l=0

DO
  IF yd=-1 THEN
    yd=0 xd=2 gprior=36
  ELSEIF yd=1 THEN
    yd=0 xd=-2 gprior=33
  ELSEIF xd=-2 THEN
    xd=0 yd=-1
  ELSE xd=0 yd=1
  FI
FOR t=0 TO 39 DO
  IF xd=-2 and x=150 THEN
    gprior=36
  FI
  IF xd=-2 and x=116 THEN
    gprior=33
  FI
  MoveMan()
  DoPhase()
  l+=+1
  IF l=2 THEN
    l=0 ys+=+1
  FI
  IF ys=8 THEN
    ys=0
    b+=+20
    IF b=misc_adr+1320 THEN
      b=misc_adr FI
    DO UNTIL vcount=128 OD
    dlist(33)=Peek(@b)
    dlist(34)=Peek(@b+1)
  FI
  Wait(2)
  IF consol=6 OR Strig(0)=0 THEN
    EXIT
  FI
UNTIL consol=6 OR Strig(0)=0  OD
ch=255 SndRst()
RETURN

PROC Init()

  skstat=3 audctl=0 high=0 score=0
MoveBlock(cb_adr, 57344, 1024)
MoveBlock(cb_adr+128, cset, 80)
MoveBlock(cb_adr+264, cset+80, 208)
MoveBlock(cb_adr+512, cset+288, 120)
MoveBlock(cb_adr+632, logo, 128)
MoveBlock(cb_adr+760, can, 8)

pmb=pmb_page*256+1024
missile=pmb-256

a=0
FOR i=0 TO 79 DO
  table(i)=a
  a=+bytes
OD

SetVbv(7, Vblank RSH 8, Vblank&$FF)
vdslst=Dli nmien=192
RETURN

PROC LoopInit()
  colr0=68 colr1=40 colr4=64
  screen=sc_adr dlist=dlist_adr
  savmsc=misc_adr
  PmSet() ZeroOut()
  phase=0 face=0 mc=1 flag=0
  cx=0 cy=0 xs=7 ys=0 x=84 y=110
  bak=0 fore=36 enable=1 fallc=1
  packs=3 score=0 fuel=50 shield=50
  whine=0 carried=0 end=0
  shake=0 shakec=0 xsm=0
  FOR i=0 TO 19 DO
    id(i)=0 fall(i)=0
  OD
  mx(0)=0 mx(1)=0
RETURN

PROC GameLoop()
  vdslst=Dli nmien=192
  GetDir()
  IF fuel=0 THEN
    yd=2 joy==%2
  ELSE
    DoPhase()
  FI
  MoveMan()

  IF x<70 OR x>176 OR
    y<90 OR y>172 THEN
    Scroll()
  ELSE
    Wait(1)
  FI

  CheckFuel()
CheckShake()
CheckRocks()
StartMiss() MoveMiss()
Bump()

IF whine<>0 THEN
  whine=-1
  Sound(3,whine,10,4)
  IF whine=0 THEN
    Sound(3,0,0,0)
  FI
FI

IF ch<255 THEN
  ch=255  SndRst()
  DO UNTIL ch<255 OR consol<>7 OR Strig(0)=0 OD
  ch=255
FI
RETURN

PROC Game()

Init()
DO
  Title()
  Graphics(0) crsinh=1
  LoopInit()
  DrawCaves()
  DoScore()
  color=0

  DO
    GameLoop()
    UNTIL consol<>7 OR end<>0 OD

  IF end<>0 THEN EndGame() FI
  IF score>0 AND score>high THEN
    high=score
  FI
OD
RETURN