H.C.Version

INSTRUCTIONS TO STAR FIRE

CODE: Basic/Machine language

The object of this game is to shoot down all the moving characters that appear on the screen. They appear one at a time going steadily in a chosen direction, you move your controller so that you can follow the target as if you were sitting in a ship. You move the controller to the right and your ship moves to the right while the target moves to the left. You push the stick foreward and your ship goes into a dive downward as your target moves upward etc. To get familar with these operations, just move the controller in any direction and watch the background moving stars. It may take you a while to get used to the movement, but those who know the operation of flight shouldn't have any difficulty. The amount of time remaining is in the top-right corner, it starts at 200 and decrements. This is the variable 'U' which can be changed in line #1. At the bottom-left are messages, bottom-center is the number of points that shot was worth, bottom-right is the total accumulated points. Points start at zero and advance in increments of 46 minus the number of loops it took you to hit the target. So the faster you react, the more points you get along with saving time. Foreground color changes along with point values in increments of 32, maximum FC value is 166-light green. If the enemy hits you, you lose 10 points for this hit and 32 points for energy lost the next time you hit something. So getting hit just once will really hurt you in points and energy. Points will now go positive again. There is a one out of nine chance that the enemy will fire at you in any given loop. If he fires at you, you will see a rectangular box getting larger and moving down-screen. You <u>can</u> fire at the same time the enemy is fireing on you, if you think you can hit him in 4 loops. The enemy takes 6 loops before he hits you. (A loop is each time a character erases and replaces itself) To evade the enemy's shot, simply make sure the top of his ship is below the center of the screen, when his shot dissappears, push foreward on the stick and dive down to fire on him.

One of three characters will show up on the screen, a fighter, a refueling station, and a meteor/planet. The fighter and refueling station can both fire on you and the background color will be a deep blue, the meteor will not fire and the background color will be black. There will always be two X and Y boxes that line up your target alongside the grid so that you don't even have to watch your target in order to hit it in the center.(Points 1 & 2 on the Specs. page) After the program loops 12 times the target appears 2 times the size in the effect that you are getting closer to the object therefore he gets larger. If the object goes off-screen, you have to go after him in the same direction he went off.

The machine language program takes 31.9 seconds to load, directly after the Basic program loads. It uses only 60 bytes, but not of basic memory, this program is stored in the Line Input Buffer but <u>after</u> the first 20 bytes of the buffer so you can still use the keypad (up to 20 bytes) without erasing the machine language program. Listing of this program is given (in HEX).

As you can see, this program can do many things at once and the subroutines should have great educational value in themselves. The more you look, the more you will find in this program.Remember, it tool 4 months to develop. Happy hunting!

NOTE: The <u>Assembler</u> (the program that wrote the machine lang. prog.) is avaliable for $2.50

STAR FIRE -listing

```
1 CLEAR ;W=0;U=200;Q=0;NT=0;GOTO 100
2 BOX F-40,I-20,1,1,3;BOX F-30,I+23,1,1,3;BOX F+30,I+18,1,1,3;
  BOX F+42,I-25,1,1,3;RETURN
5 L=-10
10 XY=-10831;LINE -42,-22,3;XY=11087;LINE 42,22,3;XY=-10929;
   LINE 42,-22,3;XY=11185;LINE -42,22,3;RETURN
20 XY=-5418;LINE -16,-8,3;XY=5674;LINE 16,8,3;XY=-5590;LINE 16,
   -8,3;XY=5846;LINE -16,8,3;L=-25;RETURN
30 XY=-1808;LINE 16,8,3;XY=2288;LINE 16,-8,3;L=0;R=0;G=0;GOSUB
   75;IF (H>80)-(V>45)-(H<50)-(V<24)X=1;RETURN
40 &(16)=255;&(17)=255;&(18)=135;&(19)=63;&(21)=255;&(22)=255;
   &(23)=255;IF JRETURN
50 FOR K=4TO 50STEP 5;BOX 0,0,K,K,3;BOX 0,0,K-2,K-2,3;BC=-KxRND
   (Kx3);XY=RND (9950)-4975;LINE 0,0,3
60 BOX 0,0,K,K+2,1;BOX 0,0,K+2,K,3;NEXT K;GOSUB 75;&(19)=63;
   FOR K=256TO OSTEP -4;FC=7xRND (36);&(23)=K
65 BOX RND (152)-76,RND (60)-26,16,17,2;NEXT K;CY=-40;P=P+W;
   Q=Q+P;PRINT "__DESTROYED!",#5,P,Q,;GOSUB 75;W=W+32;T=1;RETURN
70 BOX 0,-20,41,1,3;BOX 20,0,1,41,3;FOR K=-20TO 16STEP 4;BOX K,
   -18,1,3,3;BOX 18,K-4,4,1,3;NEXT K;RETURN
75 &(16)=0;&(17)=0;&(18)=0;&(22)=0;&(23)=0;RETURN
80 BOX Z-75,43-Y-A,A,A-4,3;A A-3;BOX Z-75,43-Y-A,A,A-4,3;IF A
   <18RETURN
85 A=0;IF V>43BOX Z-75,24-Y,19,23,3;RETURN
90 J=1;GOSUB 40;&(9)=0;FOR K=1TO 70;&(0)=RND (7);&(1)=RND (7)+77;
   NEXT K;FOR K=255TO OSTEP -3;&(23)=K;&(22)=K;NEXT K;&(9)=42;
   GOSUB 75
95 CY=-40;P=-10;Q=Q+P;PRINT "__YOU'RE HIT",#5,P,Q,;W=-32;J=0
--------Processing starts here:---------
100 BC=8;F=0;I=0;A=0;FC=134+W;BOX 0,5,159,78,2;M=0;T=0;GOSUB 2;
    H=RND (130)+15;BOX 0,-35,159,1,1;IF FC>166FC=166;W=32
120 V=RND (55)+15;C=RND (6)-3;D RND (6)-3;E RND (3);BOX H-73,-22,
    1,5,3;BOX 22,40-V,5,1,3;GOSUB 70;IF E=1M=76;BC=0
140 &(19)=0;J=0;%(20213)=128;IF E=2%(20213)=129
160 %(20204)=10240;FOR P=20TO 8STEP -1
180 GOSUB 2;BOX H-73,-22,1,5,3;BOX 22,40-V,5,1,3;F=F-JX(1)x2;
    I=I-JY(1)x2;H=H+C+F;V=V+D+I
200 IF V>70V=70;GOTO 240
210 IF V<0V=0;GOTO 240
220 %(20203)=Vx256+H;CALLB;IF P=7IF MBOX H-71,35-V,10,18,3;GOTO 240
230 IF MBOX H-M,39-V,5,9,3
240 GOSUB 2;U=U-1;CY=30;CY=40;PRINT U;BOX H-73,-22,1,5,3;BOX 22,
    40-V,5,1,3
280 IF TR(1)R=1
290 IF R_G=G+10;&(23)=64;GOSUB G+L;IF XGOSUB 20;GOSUB 30;X=0
300 IF TGOTO 100
310 IF MGOTO 360
320 IF A=1GOSUB 80;GOTO360
340 A=RND (9)-8;IF A=1Z=H;Y=V;BOX Z-75,43-Y-A,A,A-4,3;GOSUB 80
360 IF U<0GOTO 500
370 IF V=70GOTO S
380 IF V=0GOTO S
400 IF P=7CALLB;IF MBOX H-71,35-V,10,18,3
410 IF P=7GOTO S
420 CALLB;IF MBOX H-M,39-V,5,9,3
440 NEXT P;%(20204)=26624+V;GOTO S
500 CLEAR;PRINT;PRINT Q,"_PTS!
S=180;&(21)=255
```

MACHINE LANGUAGE LISTING For STAR FIRE CHARACTERS
(in HEX)

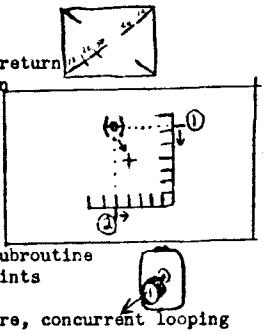
```
   D5 FF 35 16 00 28 F2 4E D1 C9 70 F8 4E 80 00 36
   80 09 0F 02 09 FF 4E 22 00 41 00 49 00 94 80 EB
   80 94 80 49 00 41 00 22 00 03 80 1F F0 36 D8 FF
   FE C2 86 07 C0 08 20 00 00 00 00 00 FF FF FF FF
```

NOTE: These programs are protected against copying, the copied program will automatically RESET when RUN.

SPECIFICATIONS

DESCRIPTION: User-Defined-Character Generating, Multi-Processing Program
VARIABLES: uses all except letter 'o'
Fixed storage: S=180 (line #)
Input Buffer: 20200-20258 (60 Bytes)
Memory left Ø Bytes, SZ 1 (last byte cannot be used)
SUBROUTINES 2-95 (kept under 100 for calculation)

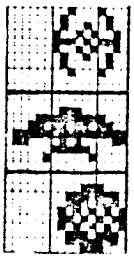
2 BACKGROUND Moving stars
10
20 > FIRE ⟶
30 calculates hit or miss & return
40 Explosion sound switched on
50 Enemy explodes-Graphics
60 Explosion dissipates
65 Score
70 Graticule/Laser sight
75 Explosion off, noise
80 Enemy's torpedeo shot
85 Return if shot missed
90 You explode, Overlapping subroutine
95 Subtract from score -32 points



VARIABLES
A Percentage the enemy will fire, concurrent looping
B Fixed addressing
C Enemy movement Horizontal
D Enemy movement Vertical
E RND(3) value for choosing target
F KN(1) value
G Next laser position
H Horizontal position-Character
I KN(2) value
J For enemy's shot
K Looping variable
L Which laser SUBROUTINE is next
M Meteor, false if E≠1
N Size of boxes in explosion
O Not used
P Points for that shot
Q Points accumulated
R Trigger status
S =180, GOTO S
T RETURN from multiple GOSUBs, clears stack memory
U #of loops, Game time
V Vertical position
W FC color & points for next shot
X TO erase miss-fire
Y Vertical position for enemy fire
Z Horizontal position of enemy fire

STRINGS- not used

CHARACTERS:



ENEMY FIGHTER

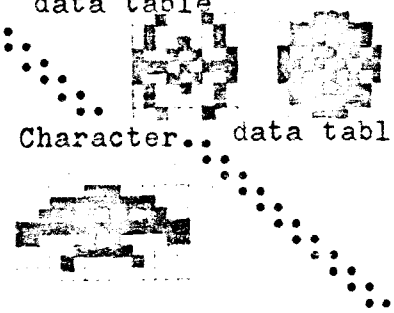
ENEMY
REFUELING STATION

METEOR/PLANET

SUBROUTINES are kept at the begining because it takes less time for the pointer to find them there and the program runs faster.

STAR FIRE ASSEMBLER
Listing

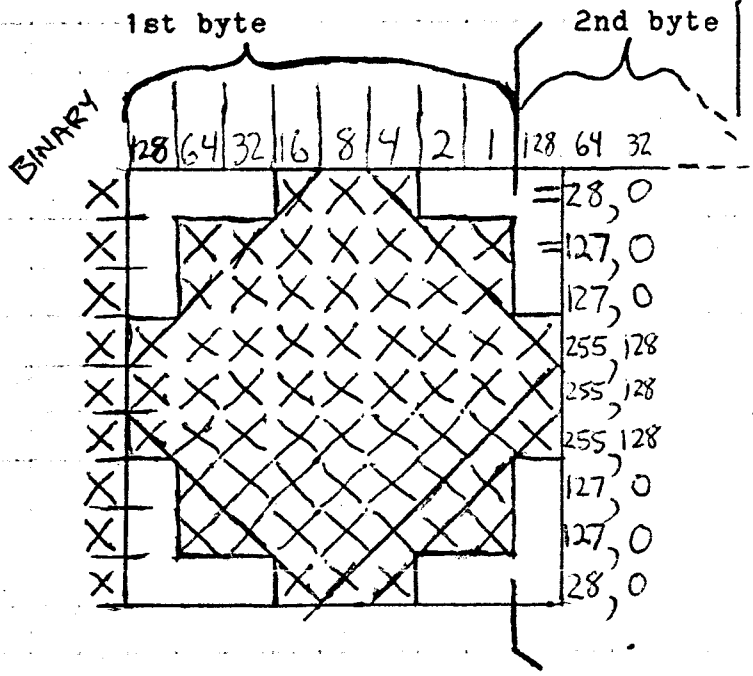
```
  1 .
 20 .ASSEMBLER FOR STAR FIRE
 30 :RETURN; CLEAR ; NT=Ø
 40 A=20200; B=A;C=310                          C&D are line #s
 45 D=320
 50 X=-43;GOSUB C                               Poke Subroutine #
 60 X=6965;GOSUB C                              General setup
 70 X=10240;GOSUB C
 80 X=20210;GOSUB C                             address of character string
 90 X=-13871;GOSUB C                            non-standard character
100 X=112;GOSUB D                               Char display perameters
110 X=20216;GOSUB C                             address of character spec table
120 X=128;GOSUB C                               Poke 1 byte space
130 A=20216                                     Character spec table start
140 X=128;GOSUB D                               Subtract from char. to get table address
150 X=9;GOSUB D                                 Horizontal size
160 X=15;GOSUB D                                Vertical size
170 X=2;GOSUB D                                 (Horizontal size-1)/8+1
180 X=9;GOSUB D                                 Vertical sizex((Horiz.-1)/8+1)
190 X=20223;GOSUB C                             Address of character table
200 A=20223                                     Character table start
210 X=34;GOSUB C;X=65;GOSUB C;X=73;GOSUB C;     Tie-fighter character
    X=148;GOSUB D;X=128;GOSUB D;X=235;GOSUB     data table
    D;X=128;GOSUB D
220 X=148;GOSUB D;X=128;GOSUB D;X=73;GOSUB
    C;X=65;GOSUB C;X=34;GOSUB C
230 X=3;GOSUB D;X=128;GOSUB D;X=31;GOSUB D;     UFO Character.. data table
    X=240;GOSUB D;X=54;GOSUB D;X=216;GOSUB
    D;X=255;GOSUB D;X=254;GOSUB D
240 X=194;GOSUB D;X=134;GOSUB D;X=7;GOSUB D;
    X=192;GOSUB D;X=8;GOSUB D;X=32;GOSUB D;
    X=0;GOSUB D;X=0;GOSUB D;X=0;GOSUB D
260 PRINT " READY";A=KP                         Wait for user to press keypad
270 :PRINT ;TV=31;TV=31;TV=31;TV=13;A=KP;       TV=31-erases ?'s, TV=13-GO Char.
    PRINT "CLEAR ";CLEAR
275 NT=2                                        NT=2--not too fast
280 FOR S=20200 TO 20258 STEP 2                 Start & end addresses of mach.lan
290 PRINT #1,"%(",S,")=",%(S)                   Print out %(20200)=-43
295 FOR Q=1TO 35;NEXT Q                         Wait loop, input slower than out
297 NEXT S;PRINT "%(20080)=20200                Poke CALL address
299 PRINT ":RETURN;RUN ";GOTO 390               Auto start
310 %(A)=X;A=A+2;RETURN                         C&D Subroutines  Pokes into input buffer
320 %(A)=X;A=A+1;RETURN
390 B=20200;F=80;I=40;CLEAR                     F&I are X&Y location
395 :RETURN ;NT=0                               speed up
400 F=F+JX(1)x2;I=I+JY(1)x2                     read joystick
410 IF I>7ØI=Ø                                  check for off screen @ bottom
420 IF I<ØI=7Ø                                  check for off screen @ top
430 %(20203)=Ix256+F                            Poke location of character
440 %(20204)=10240+I;IF TR(1)%(20204)=26624+I   If trig is pressed char 2x size
450 CALLB                                       Gosub machine language program
460 BC=&(28);FC=BC+4                            Background color knob control
470 P=P+1                                       bump loop
480 CALLB
490 IF P>15%(20213)=RND (2)+127;P=0             Choose random character
500 GOTO 400
```



STAR FIRE ASSEMBLER
CHARACTER SPECS

The STAR FIRE program uses a user-defined subroutine located in the BALLY's 10K ROM, this assembler is a program which assembles (writes) a machine language program to store on tape for use in a Basic program, which CALLs the machine language program from it's location in the Input Buffer. The user can define his own character, and use it any way he wants in the area of graphics. Because the program is in machine language, it runs many times faster and is much easier to use than puting BOXes on the screen with BASIC.

Example shows bowling ball character table 9x9 size



size of character is specified in lines 150 (Horiz) & 160 (Vert). Character table (@ left) is stored in lines 210 to 250. GOSUB C increments by 2, GOSUB D increments by 1. You can use table at left by using GOSUB D and inputing (X=Number), starting with (shown @ left) X=28;GOSUBD;X=0 GOSUB D;X=127;GOSUB D;X=0;GOSUB D; X=127.....etc.
Remember to change the size perameters @ 150, 160.

If you like what you see that you can do, you can order my TUTORIAL on the use of the Internal Subroutines Character Generator. This will cover the complete use of this software, and show the user through examples, how he can CALL his own characters onto the screen and make them move around in REAL TIME , total Machine Language control.TUT.CHAR.GEN.$7.95

52 07CA E,D,C,L,H
String display routine.
Call with E = Horizontal position (0-159D).
D = Vertical position (0-95D).
C = Character display parameters. (See Appendix A.)
HL = Address of character string to display.
CHARACTER STRING:
Bit 7 0 = use standard ASCII characters defined in Character table.
1 = Use non-standard display. IX must contain address of specification table.
If bit 7 = 0, bits 6-0 00 = end of string.
01-1F = Number of Character spaces to skip.
20-64 = Standard character to display.
65-7F = Input parameter setup.
Bit 0 = E (Horizontal position)
Bit 1 = D (Vertical position)
Bit 2 = C (Character display parameters. See Appendix A.)
Bit 4 = IX (Address of display spec table)
If Bit 7 = 1, Bits 6-0 00-7F = Non-standard character to display.

CHARACTER SPECIFICATION TABLE

7 Byte table as follows:

Byte 0	20	Amount to subtract from character so result becomes table entry number.
1	8	Horizontal Character allotment
2	8	Vertical Character allotment.
3	1	Number of bytes per table entry. (Number of bytes/Horizontal line.)
4	7	No. of table entries per character.
5	EA	Address of Character table.
6	09	" " " "

Subroutines 50, 52, 54 and 82 use the C register to pass display parameters (character size and type of character display, character and background color) as follows:

Bits 7-6	Character size. 0-3 = multiplication factors 1, 2, 4 and 8 respectively.
Bits 5-4	0 = Obliterate character field with background color specified by bits 0 and 1.
	1 = Only characters will obliterate existing display; data not actually in conflict with character will remain intact.
	2 = Characters displayed in such a manner as to leave existing display legible. Existing display in conflict with character area will change to a contrasting color. Alternate requests to display the same character in the same area will result in the characters alternately appearing and disappearing.
	3 = A combination of 1 and 2 resulting in the area of the CRT required for character display being blanked with no character displayed.
Bits 3-2	Color port (08) index to be used for characters.
Bits 1-0	Color port (08) index to be used for character area background.