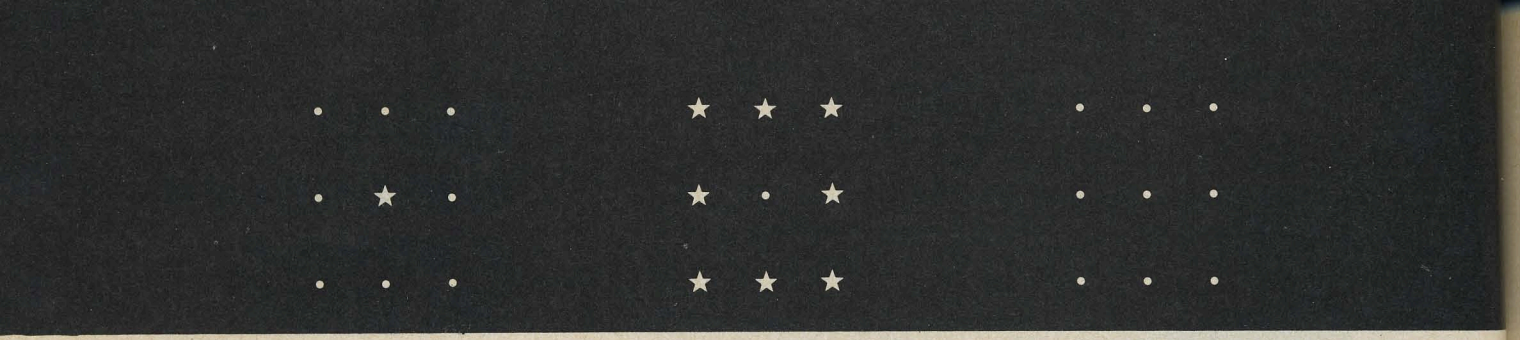


Figure 1: Three special patterns of stars and black holes. The game begins with a single star representing the Big Bang theory (left), and is won when the pattern of only one central black hole is achieved (center). The pattern shown on the right represents a loss and terminates the game.

SHOOTING STARS

Willard I Nico
Delta t
11020 Old Katy Rd, Suite 204
Houston TX 77043

There are probably as many reasons to have a computer in the home as there are computers in homes. For whatever reason you have one though, it's only human nature to want to show it off to other people.

Say you have a super program called "Investment Portfolio Analysis and Statistical Summary" (IPASS) up and running on your Scelbi 8H or whatever. It took months to write and debug the program and it involved several unique concepts of which you are justifiably proud. You can picture the furious activity going on inside the little heart of the computer and would dearly love to show off your skill to Mr and Mrs Nexdor and bask in their admiration. So you invite them over for cocktails.

The program runs flawlessly and, as the results flash on the display screen, you step back slightly to receive your praise. Mr Nexdor looks at you with a blank expression and says, "But will it grind pepper?"

That actually happened to me. One way around this problem is to save IPASS for your own enjoyment and have a game program or two available to show off. Of course, for some people game programs are the primary interest in having a home computer. Whatever your games interest, I think you'll find SHOOTING STARS an interesting addition to your library.

I started my quest for a "show-off" game about a year ago, searching everywhere for one that was just right. I learned a very interesting fact quickly: My computer doesn't speak BASIC, and to date many games have been written and published in that language.

So I had to do it myself. The result is SHOOTING STARS, a game with enough challenge to intrigue, enough variables to make learning to win difficult (but not impossible), and a couple of goodies thrown in to involve the player with the computer.

A complete program listing for 8008 computer is included, as well as the various messages that allow the computer to interact with the player.

The Game

Nine dot or asterisk characters are arranged in a 3 by 3 matrix on the playing field which may be shown on a CRT screen. The matrix represents the universe; asterisks are stars and dots are black holes. The player shoots stars which die and turn into black holes. When a star dies, it affects other stars and black holes in its particular galaxy.

How To Play

Each position in the universe is assigned a number (see figure 2). The computer outputs the current composition of the universe and asks YOUR SHOT? The player responds by typing the position number of the star he decides to shoot. Then the new constellation is displayed for the next shot.

Effect Of Shooting A Star

When a star dies, it affects the stars and black holes of its particular galaxy. The *effect* is that fragments of the star move into black holes to become new stars and other fragments collide with other stars and knock them out of orbit producing black holes. Each star has its own galaxy as shown in figure 3.

The Program

The game proceeds in an orderly manner which is shown in the Flow Chart of figure 4. The heading, rules and interactive messages require approximately 1600 B of memory. I use a Delta t Digital Recorder for message storage and retrieval since it operates in the reverse as well as forward incremental modes. Each message is prefaced with a message number surrounded with STX and ETX characters. A search routine in the main program finds the first address, decides whether the desired message is ahead or behind the current tape position, and rewinds or spins forward as necessary.

Table 1 is a list of the interactive messages. For computers with limited memory the essential messages are in the first portion of the table; the fancy heading is next, and the rules of the game occupy the largest number of bytes at the end of the text.

When the program is entered at address 014000, the 8008's H and L pointers are set to the beginning of the heading. Then the message control routine is called. It outputs sequentially each character of the message until the EM delimiter is encountered which returns control to the main program.

The status of the universe is stored in the B and C registers. Universe positions 1 through 4 and 6 through 9 are represented by the eight bits in the B register. A one bit represents a star, and a zero a black hole. Bit 0 of the C register keeps track of position 5.

The universe is set up in the beginning by clearing the B register and setting C to 001

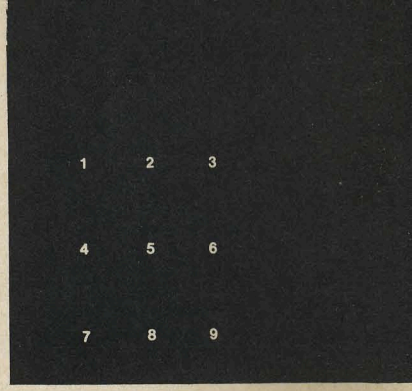


Figure 2: Positions in the universe are identified by numbers.

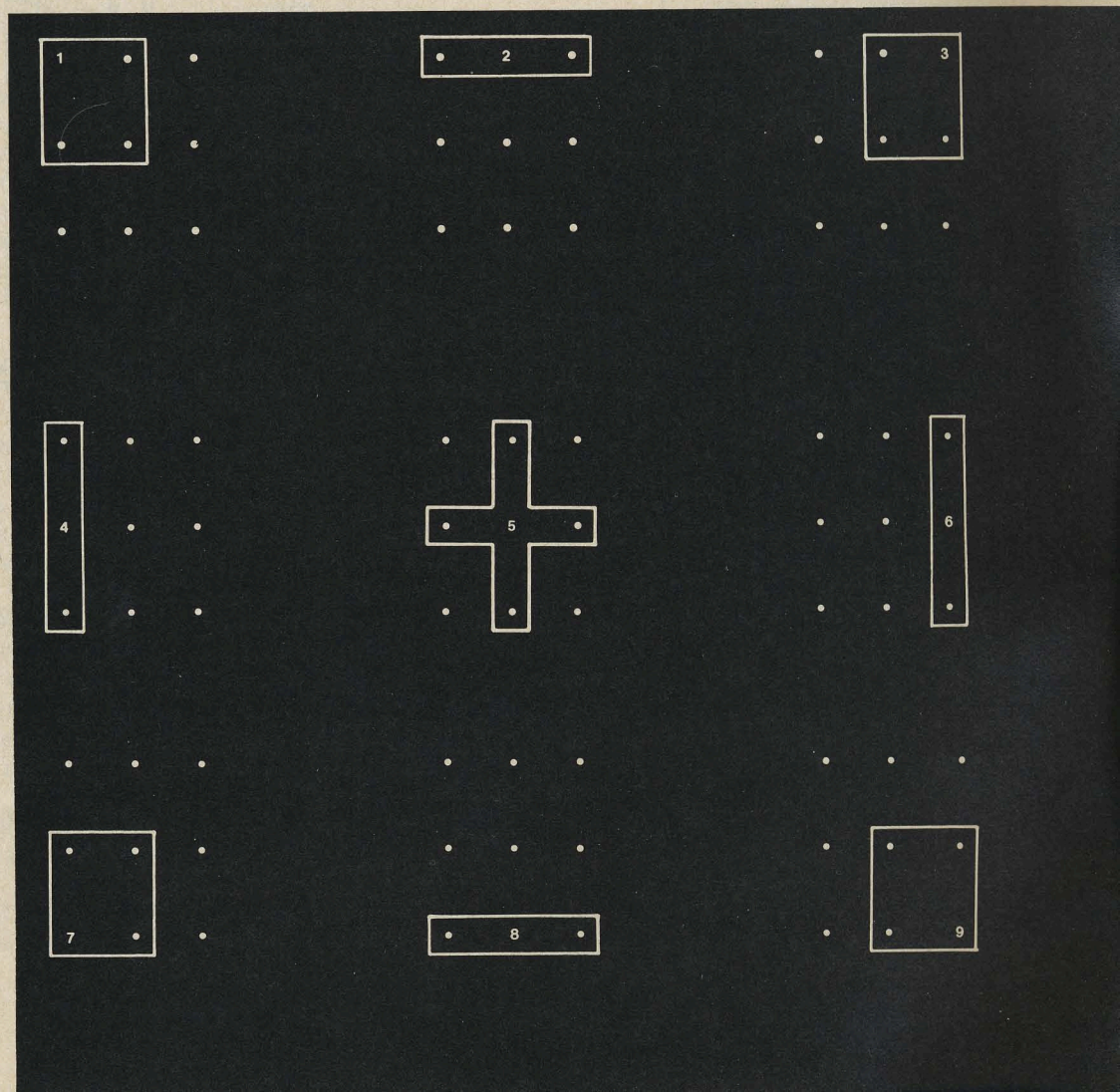


Figure 3: A complete set of galaxies which are associated with every star or black hole position. Stars or black holes within a galaxy are affected whenever the respective position has been chosen.

octal. The D register, which will tally the number of shots fired, is also cleared as part of the initialization process. Each time the print universe routine is entered after a valid shot, the D register is incremented to count the shot.

Displaying The Universe

First, the print universe routine is entered. This routine sets the E register to octal 012 and will decrement the register each time the print loop is executed. The E register tells the program when it needs to insert a couple of linefeeds for spacing, when it needs to branch to the position 5 special routine, and when it has finished printing the universe. These events occur at the following E register exception counts:

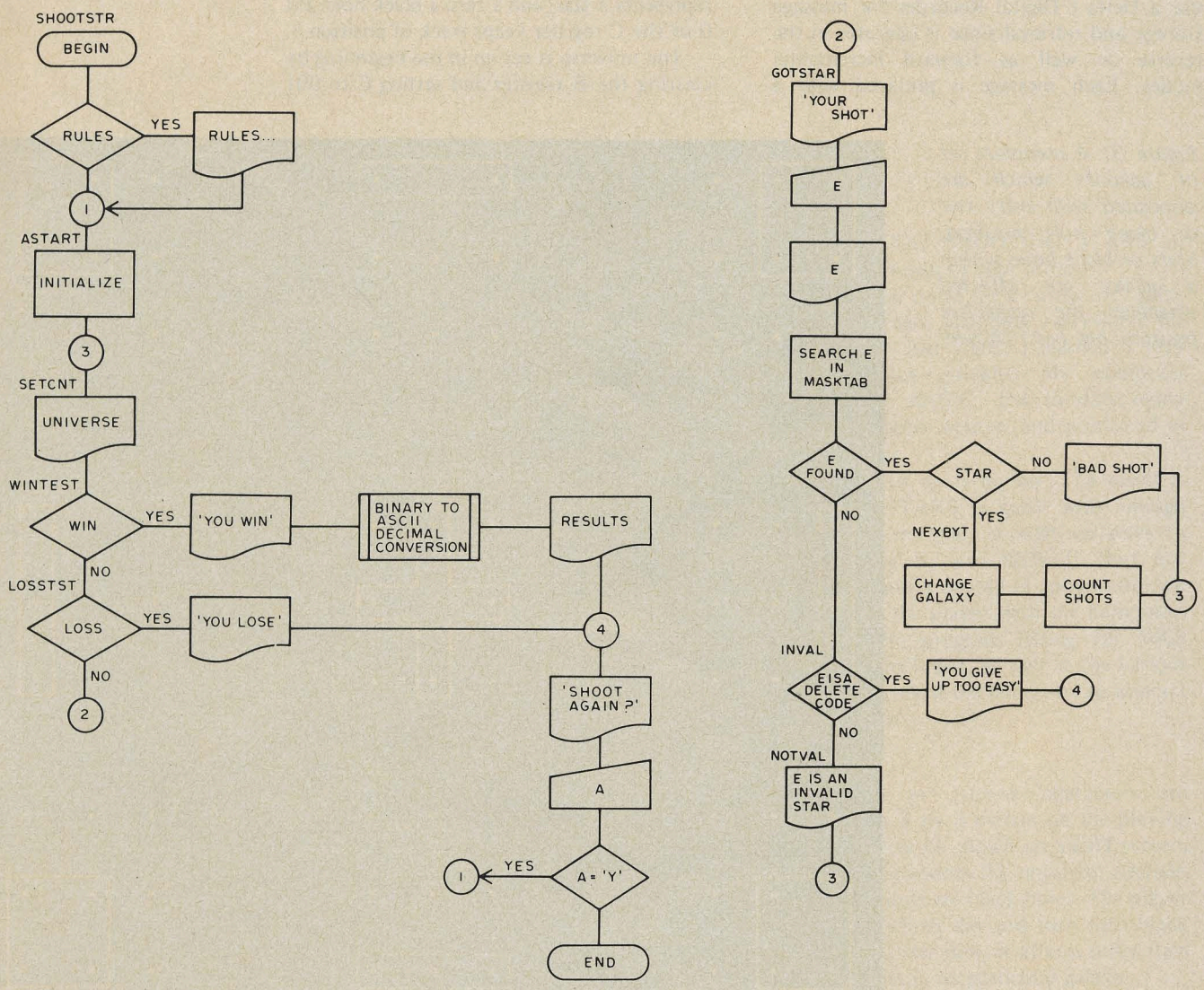
006 – Insert two linefeeds
005 – Go to position 5 subroutine
003 – Insert two linefeeds
000 – Done Print; exit

In normal processing, the positions represented by the bits in the B register are inspected one-by-one for star or black hole status, and the corresponding symbol is printed. It's done like this: The B register is loaded to A and rotated one place to the right. The rotated byte is loaded into B to be ready for the following position next time around in the loop. The carry flag is then tested for a one or zero. If the carry is zero, the program jumps to the dot output section. A one in the carry bit causes the asterisk output to be executed.

At the exception counts, further processing is required.

Thus when the E register count indicates that position 5 is the next one to be printed, the program loads the C register to A and

Figure 4: A flow chart of the SHOOTING STARS program acts as a guide to the listing. The labels indicated on this flow chart correspond to the labels found in table 3.



rotates the least significant bit to carry. The program then jumps back to the asterisk and dot output portion of the loop. Note that the rotated C register content is not loaded again to C, since we are only interested in the least significant bit.

Shoot A Star

When the universe has been displayed, the message YOUR SHOT? is printed and the computer waits for the player to type a number from 1 to 9 which indicates the star he wants to shoot. The ASCII code for the number the player types is compared to the first byte in each group of four contained in the MASKTAB table 2. The number of tries at the table is monitored by the E register, which starts at 011 and is decremented each time around the "test for match" loop. If the E register gets to 000 without finding a match, the input is tested for code 177 (delete), indicating that the player gives up and wants to start over. If a match still can't be found, the NOT A VALID STAR NUMBER message is printed, and the universe displayed again. If this happens, the print universe routine is entered just after the instruction that causes the shot to be counted, so the player won't be charged for his mistake.

When a find is made in the MASKTAB table, the program is ready to process the player's shot. First, it must make sure the player is following the rules and hasn't shot a black hole. The second byte of the four byte group is used as a "mask" to blank out all the positions of the universe except the one that has been shot. Figure 5 shows how the mask is used with the Boolean AND function to isolate the bit representing the shot position from among the eight bits of the B register. After masking out all but the selected position, the resultant byte is tested to see if it is zero. If it is, the shot position was a black hole and the message HEY! YOU CAN ONLY SHOOT STARS, NOT BLACK HOLES! is printed. If this happens, the universe is displayed again without counting the shot.

If the mask itself is zero, it indicates that position 5 was selected, and so the program

Table 1: Program Messages. This table lists all the messages used by SHOOTING STARS. Each message entry in the table starts with a symbolic name and an absolute address. The text should be stored at ascending memory address locations, and terminated with an end of message (EM) delimiter of octal 031, which is printed as ■. The symbolic names in this table are referenced by table 3.

```
MESS1:
    016000
    HEY! YOU CAN ONLY SHOOT STARS,
    NOT BLACK HOLES.
    TRY AGAIN!
    ■

MESS2:
    016077
    THAT WASN'T A VALID STAR NUMBER!
    TRY AGAIN!
    ■

MESS3:
    016156
    YOU LOST THE GAME!
    WANT TO SHOOT SOME MORE STARS?  ■

MESS4:
    016243
    YOU WIN! GOOD SHOOTING!
    YOU FIRED  ■

MESS5:
    016310
     SHOTS.
    BEST POSSIBLE SCORE IS 11 SHOTS.
    WANT TO SHOOT AGAIN, DEADEYE?  ■

MESS6:
    017022
    YOU GIVE UP TOO EASILY!
    WANT TO SHOOT SOME MORE STARS?.  ■

MESS7:
    017114
    YOUR SHOT?  ■

HMESS:
    017131
    S H O    SSS    TTT    AAA    RRR    SSS
             S       T      A A    R R    S
    O * T    SSS     T      AAA    RRR    SSS
               S     T      A A    RR       S
    I N G    SSS     T      A A    R R    SSS
    ••••••••••••SHOOTING STARS••••••••••••
                A BRAIN TEASER GAME
    WANT THE RULES?  ■

PAGE1:
    020147
    THERE ARE STARS:  *
    AND BLACK HOLES:  .
    IN THE UNIVERSE:      * * *
                          * * *
                          * * *

    YOU SHOOT A STAR
    (NOT A BLACK HOLE)
    BY TYPING ITS NUMBER      1 2 3
                              4 5 6
                              7 8 9
    THAT CHANGES THE STAR TO A BLACK HOLE!
    (TO SEE MORE RULES, TYPE ANY KEY.) ■

PAGE2:
    021277
    EACH STAR IS IN A GALAXY. WHEN YOU
    SHOOT A STAR, EVERYTHING IN ITS GALAXY
    CHANGES. ALL STARS BECOME BLACK HOLES
    AND ALL BLACK HOLES BECOME STARS.
    GALAXIES:
    1 * .   * 2 *   . * 3   * . . . * .
    * * .   . . .   . * *   4 . .   * 5 *
    . . .   . . .   . . .   * . .   . * .
    . . *   . . .   . . .   . . .
    . . 6   * * .   . . .   . * *
    . . *   7 * .   * 8 *   . * 9
    (TYPE ANY KEY FOR LAST PAGE OF RULES.) ■

PAGE3:
    023137
    THE GAME STARTS          . . .
    WITH THE UNIVERSE        . * .
    LIKE THIS                . . .

    YOU WIN WHEN YOU         * * *
    CHANGE IT TO THIS        * . *
                             * * *

    YOU LOSE IF YOU          . . .
    GET THIS                 . . .
                             . . .

    READY TO PLAY. TYPE ANY KEY TO START
    THE GAME. GOOD LUCK! ■
```

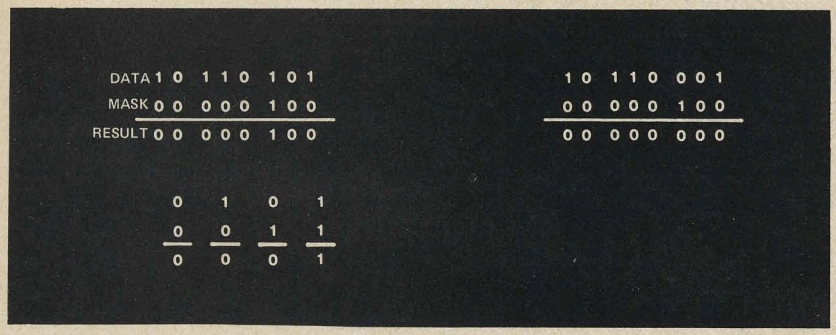


Figure 5: The AND function of Boolean logic is used to mask the current universe in order to select one position for testing each shot.

	LOCATION	SHOT	POSITION MASK	GALAXY MASK	CENTER MASK
MASKTAB	015070	061	001	013	001
	015074	062	002	007	000
	015100	063	004	026	001
	015104	064	010	051	000
	015110	065	000	132	001
	015114	066	020	224	000
	015120	067	040	150	001
	015124	070	100	340	000
	015130	071	200	320	001

Table 2: MASKTAB, a table of masks to test and alter galaxies. This table gives the data needed for memory locations 015/070 to 015/133 in the SHOOTING STARS program. This table is used to check the shot fired for a valid star number and to change the portion of the universe which is affected by the star's change.

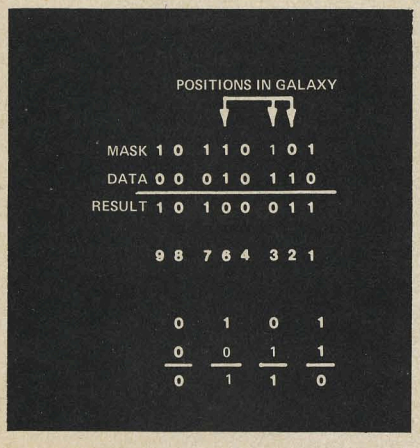


Figure 6: The EXCLUSIVE OR function of Boolean logic is used to complement bits selected according to the galaxy information stored for the position just shot.

tests the C instead of the B register for a star.

Change A Galaxy

Once the program has determined that the shot was valid, it can use the next byte in the MASKTAB table to change the dots and stars in the galaxy of the "shot" star. Again, the table entry is a mask, but this time the Boolean EXCLUSIVE OR function is used. The result is that the selected positions are *complemented;* one bits are changed to zero bits and the zeros are changed to ones. Figure 6 shows how the mask does this neat trick. After the change is made, the new universe is stored in the B register.

Byte four of the MASKTAB table entry contains a mask that is used to EXCLUSIVE OR the C register to change position 5 if required. If star 5 is to be complemented, the mask will be octal 001; if not, it will be octal 000.

After the universe in the B and C registers is changed, the new universe is displayed and the cycle repeats until a win or a loss is detected, or until the player gives up.

Win Or Loss Test

Each time the universe is displayed, it is tested for a win or a loss. If both the B and C registers contain the octal number 000, the YOU LOST THE GAME message is printed, and the opportunity to play again is offered.

If the B register contains octal 377 and C is octal 000 a win is detected. After displaying the proper message, the binary content of the D register is converted to decimal numbers and the number of shots fired is printed. The calculation is performed by the binary to decimal conversion subroutine.

Binary To Decimal Conversion

The B, C and E registers are assigned the functions of summing the hundred, ten and unit digits of the score respectively. The process is one of repetitively adding a one to the three digit number while subtracting a one from the shots fired register (D). Looping continues until all shots fired have been counted in the 3 digit decimal form.

The somewhat unusual feature of the binary to decimal conversion is that it is done directly in ASCII numeric code. The three registers B, C and E are intially loaded with octal 060, which is the ASCII numeric character zero. After each increment, the least significant digit register (E) is tested to see if it contains octal 072. If it does, the register has counted 060, 061 ... 071, which is 0 through 9 in ASCII, and has just been incremented one more to 072. When

the register has 072, a carry condition exists. When this condition is detected, the register is reset to 060 and the next register in line (C) is incremented. After incrementing, the second register is tested for a carry in the same manner, and so on. When all the shots have been counted, the registers B, C and E will not only represent the decimal equivalent of the shots fired, but will contain the proper ASCII codes for the decimal digits of the count.

Print The Shots

To suppress leading zeros, the hundreds digit (B) is tested for octal 060. If it contains any other code, the contents of all three registers will be printed. If it contains octal 060, the tens register (C) is similarly tested and the output will be one digit if it is at zero (code 060) and two digits if it is not.

Figure 7 contains a flow chart of the binary to decimal conversion program. You may find use for it in some of your other programs.

Program Listing Conventions

Table 3 contains the complete program as it was implemented in my 8008 system using the SCELBI 8H computer. The listing is in symbolic assembly language with absolute octal address and memory contents.

The 8008 computer has 8 possible restart instructions which are one byte calls to locations in the first portion of memory address space. These are used to access utility subroutines needed by the SHOOTING STARS program. The required restarts are as follows:

RST0: User's input routine, starting at location 000/000 which is used to wait for one character input from the keyboard device.

RST1: Exit Routine, starting at location 000/010. This is a return address to the system monitor for the computer.

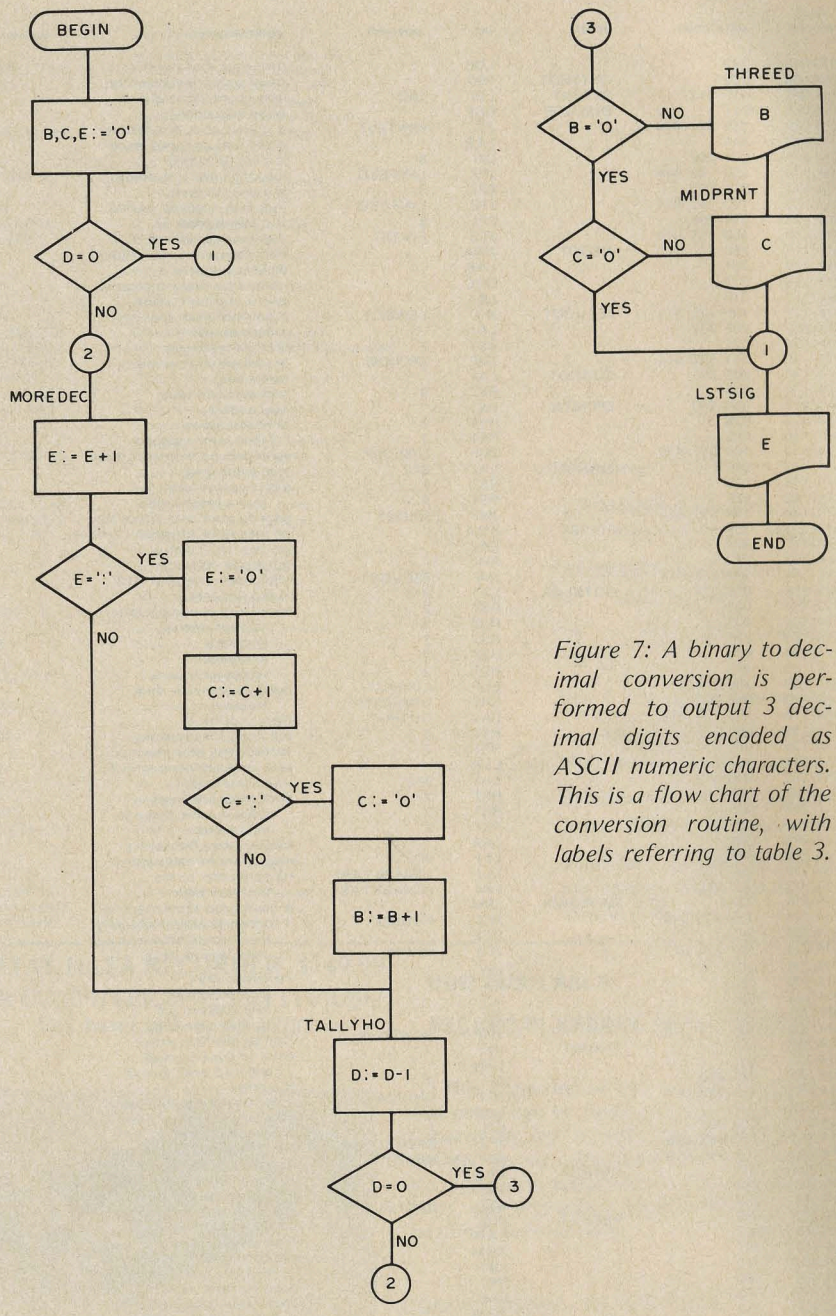


Figure 7: A binary to decimal conversion is performed to output 3 decimal digits encoded as ASCII numeric characters. This is a flow chart of the conversion routine, with labels referring to table 3.

octal address	octal code	label	op.	operand	commentary
014/000	006 012	SHOOTSTR	LAI	012	display a linefeed to
014/002	025		RST	2	initialize display;
014/003	066 131		LLI	L(HMESS)	set address pointers
014/005	056 017		LHI	H(HMESS)	to heading message;
014/007	106 134 015		CAL	OUTPUT	print message & return;
014/012	106 151 015		CAL	INPUT	call input looper;
014/015	074 116		CPI	'N'	is first letter 'N'?
014/017	150 052 014		JTZ	ASTART	if so then plunge into game;
014/022	066 147		LLI	L(PAGE1)	if not then point to first
014/024	056 020		LHI	H(PAGE1)	page of rules text;
014/026	106 134 015		CAL	OUTPUT	and go output rules message;
014/031	075		RST	7	wait for goahead;
014/032	066 277		LLI	L(PAGE2)	point to second page of
014/034	056 021		LHI	H(PAGE2)	rules text;
014/036	106 134 015		CAL	OUTPUT	display second page of rules;
014/041	075		RST	7	wait for goahead;
014/042	066 137		LLI	L(PAGE3)	point to third page of
014/044	056 023		LHI	H(PAGE3)	rules text;
014/046	106 134 015		CAL	OUTPUT	display third page of rules;
014/051	075		RST	7	wait for goahead;
014/052	006 012	ASTART	LAI	012	set up linefeed;
014/054	025		RST	2	display one linefeed,
014/055	025		RST	2	then a second linefeed,
014/056	025		RST	2	then a third;
014/057	016 000		LBI	0	initialize the universe
014/061	026 001		LCI	1	to starting pattern;

Table 3: The SHOOTING STARS program specified in symbolic assembly language with an absolute listing of addresses and codes for the author's system.

```
octal
address    octal code      label     op.     operand      commentary

014/063    331                       LDB                  then clear shot counter;
014/064    030             CNTSHOT   IND                  count a shot (anticipatory);
014/065    046 012         SETCNT    LEI     10D          loop count 10 iterations;
014/067    041             DISLOOP   DCE                  is the loop done?
014/070    150 321 014               JTZ     WINTEST      if so then go to win testing;
014/073    304                       LAE                  if not then continue display;
014/074    074 006                   CPI     6            is it fourth cycle?
014/076    150 142 014               JTZ     LINFEED      if so then new line needed;
014/101    074 003                   CPI     3            is it seventh cycle?
014/103    150 142 014               JTZ     LINFEED      if so then new line needed;
014/106    074 005                   CPI     5            is it star number 5?
014/110    150 151 014               JTZ     FIVTST       if so then go test star 5;
014/113    250             NEDOT     XRA                  clear the carry (and A too);
014/114    301                       LAB                  move universe to A;
014/115    012                       RRC                  rotate next place into carry;
014/116    310                       LBA                  save it in B for a while;
014/117    100 130 014     PSEUDOT   JFC     LOADOT       if dot then go output dot;
014/122    006 052                   LAI     '*'          otherwise load a star;
014/124    025                       RST     2            then print the star;
014/125    104 133 014               JMP     SPCNOW       branch around dot logic;
014/130    006 056         LOADOT    LAI     '.'          load a dot;
014/132    025                       RST     2            then print the dot;
014/133    006 040         SPCNOW    LAI     ' '          load a space;
014/135    025                       RST     2            print one space,
014/136    025                       RST     2              then print a second;
014/137    104 067 014               JMP     DISLOOP      waltz around loop once more;
014/142    006 012         LINFEED   LAI     012          load a line feed;
014/144    025                       RST     2            display a line feed,
014/145    025                       RST     2              then a second one;
014/146    104 113 014               JMP     NEDOT        back to print next dot or star;
014/151    250             FIVTST    XRA                  no operation intended - leftover;
014/152    302                       LAC                  get position 5 status;
014/153    012                       RRC                  put status into carry;
014/154    104 117 014               JMP     PSEUDOT      rejoin main line after RRC;
014/157    006 012         GOTSTAR   LAI     012          load a line feed;
014/161    025                       RST     2            have finished universe print,
014/162    025                       RST     2              so print several
014/163    025                       RST     2              line feeds
014/164    025                       RST     2              to separate
014/165    025                       RST     2              successive rounds;
014/166    066 114                   LLI     L(MESS7)     point to the 'your shot'
014/170    056 017                   LHI     H(MESS7)       message;
014/172    106 134 015               CAL     OUTPUT       then go print it;
014/175    005                       RST     0            call input for character;
014/176    025                       RST     2            immediately echo the input;
014/177    340                       LEA                  save input temporarily in E;
014/200    006 012                   LAI     012          load a line feed;
014/202    025                       RST     2            print three line feeds to
014/203    025                       RST     2              space out the response
014/204    025                       RST     2              a bit more;
014/205    304                       LAE                  recover input for testing;
014/206    046 011                   LEI     9D           loop count for table search;
014/210    066 070                   LLI     L(MASKTAB)   set up pointer to the
014/212    056 015                   LHI     H(MASKTAB)     the mask table;
014/214    277             NEXGRUP   CPM                  is input equal table character?
014/215    150 233 014               JTZ     FOUND        if so then go alter structure of
014/220    041                       DCE                    the universe otherwise just
014/221    150 273 014               JTZ     INVAL          check end of loop;
014/224    060                       INL                  increment the L
014/225    060                       INL                    register pointer
014/226    060                       INL                    four times to get
014/227    060                       INL                    to next table entry;
014/230    104 214 014               JMP     NEXGRUP      then go test next entry;
014/233    060             FOUND     INL                  point to position mask
014/234    307                       LAM                    and load mask into A;
014/235    074 000                   CPI     0            is it zero?
014/237    110 253 014               JFZ     UNIV2A       if not then fringe position;
014/242    302                       LAC                  otherwise the center position;
014/243    074 001                   CPI     1            is a star in center?
014/245    110 165 015               JFZ     BADFELO      if not then have wrong move;
014/250    104 260 014               JMP     NEXBYT       if so then go process star;
014/253    301             UNIV2A    LAB                  rest of universe to A;
014/254    247                       NDM                  AND with mask to isolate star;
014/255    150 165 015               JTZ     BADFELO      if not star then wrong move;
014/260    060             NEXBYT    INL                  point to the galaxy mask;
014/261    301                       LAB                  fetch universe again;
014/262    257                       XRM                  and complement the universe
014/263    310                       LBA                    on a fine performance;
014/264    060                       INL                  point to center mask;
014/265    302                       LAC                  fetch center of universe;
014/266    257                       XRM                  complement center if required;
014/267    320                       LCA                  save center of universe;
014/270    104 064 014               JMP     CNTSHOT      go display a new universe;
014/273    074 177         INVAL     CPI     177          was invalid shot a 'delete'?
014/275    110 307 014               JFZ     NOTVAL       if not then recycle bad star;
014/300    066 022                   LLI     L(MESS6)     otherwise point to giving up
014/302    056 017                   LHI     H(MESS6)       message;
014/304    104 034 015               JMP     PRNTIT       display then test for restart;
014/307    066 077         NOTVAL    LLI     L(MESS2)     point to the invalid star
014/311    056 016                   LHI     L(MESS2)       number message
014/313    106 134 015     OUTMES    CAL     OUTPUT       output a message then
014/316    104 065 014               JMP     SETCNT         go display the universe again;
014/321    301             WINTEST   LAB                  move universe to A;
014/322    074 377                   CPI     11111111B    are all fringe stars present?
014/324    110 050 015               JFZ     LOSSTST      if not see if player has lost;
014/327    302                       LAC                  fetch center of universe;
014/330    074 000                   CPI     0            is center of universe empty?
014/332    110 157 014               JFZ     GOTSTAR      is full then not win;
014/335    066 243                   LLI     L(MESS4)     no star! got a win, folks
014/337    056 016                   LHI     H(MESS4)       so point to win message;
014/341    106 134 015               CAL     OUTPUT       then display win message;
014/344    046 060                   LEI     '0'          begin binary to decimal conversion
014/346    314                       LBE                    by setting all three working
014/347    324                       LCE                    register to (ASCII) zero;
014/350    031                       DCD                  get rid of last shot;
014/351    303                       LAD                  move shot count to A for test;
014/352    074 000                   CPI     0            test for zero (not needed in
014/354    150 026 015               JTZ     LSTSIG         SHOOTING STARS but generally
                                                            useful with conversions);
014/357    006 072                   LAI     '9'          need compare to ASCII '9' + 1;
014/361    040             MOREDEC   INE                  count up one in 1.s. digit;
014/362    274                       CPE                  is it equal to overflow code?
014/363    110 000 015               JFZ     TALLYHO      if not then tally and continue;
014/366    046 060                   LEI     '0'          else reset 1's digit to zero
014/370    020                       INC                    and carry into next digit;
```

RST2: User's output routine, starting at location 000/020. This routine prints or displays one character on the output device for the system. The character to be output is in the A register when RST2 is entered.

RST7: A "do Nothing" keyboard input acknowledgement routine, starting at location 000/070. Any character typed on the keyboard causes return from this subroutine.

For the optimum use of the program, the output device should be a cathode ray tube terminal with a scrolling feature.

Game Background

I first saw the SHOOTING STARS game in the September, 1974, issue of PCC† as a program called TEASER. If you are an analytical person, you can figure out all of the possible positions.

PCC Editor, Bob Albrecht, told me that the program was contributed to the Hewlett-Packard software library, and originally written in BASIC.■

---

†PCC is People's Computer Company which publishes a tabloid size computer hobbyist newspaper five or more times during the school year. It's filled with games written in BASIC, art, and computer news. If you are interested, write to People's Computer Company, PO Box 310, Menlo Park CA 94025.

Symbol table, in order of appearance

SHOOTSTR	014 000
ASTART	014 052
CNTSHOT	014 064
SETCNT	014 065
DISLOOP	014 066
NEDOT	014 113
PSEUDOT	014 117
LOADOT	014 130
SPCNOW	014/133
LINFEED	014/142
FIVTST	014 151
GOTSTAR	014 157
NEXGRUP	014/214
FOUND	014/233
UNIV2A	014/253
NEXTBYT	014 260
INVAL	014 273
NOTVAL	014 307
OUTMES	014/313
WINTEST	014/321
MOREDEC	014/361
TALLYHO	015 000
THREED	015/023
MIDPRNT	015/025
LSTSIG	015 026
RECYC	015/032
PRNTIT	015/034
LOSSTST	015/050
MASKTAB	015/070
OUTPUT	015/134
INPUT	015/151
GETNEXT	015/154
BADFELO	015/165
MESS1	016/000
MESS2	016/077
MESS3	016/156
MESS4	016/243
MESS5	016/310
MESS6	017/022
MESS7	017 144
HMESS	017/131
PAGE1	020/147
PAGE2	021/277
PAGE3	023/137

octal address	octal code	label	op.	operand	commentary
014/371	272		CPC		is it equal to overflow code too?
014/372	110 000 015		JFZ	TALLYHO	if not then tally and continue;
014/375	026 060		LCI	'0'	else reset middle digit to zero
014/377	010		INB		and carry into m.s. digit;
015/000	031	TALLYHO	DCD		decrement score counter for tally;
015/001	110 361 014		JFZ	MOREDEC	if not zero then keep loopin;
015/004	301		LAB		fetch leading digit to A;
015/005	074 060		CPI	'0'	is it (ASCII) zero?
015/007	110 023 015		JFZ	THREED	if not go display three digits;
015/012	302		LAC		fetch middle digit to A;
015/013	074 060		CPI	'0'	is it (ASCII) zero too?
015/015	110 025 015		JFZ	MIDPRNT	if not go display two digits;
015/020	104 026 015		JMP	LSTSIG	if so display only one;
015/023	025	THREED	RST	2	display three digits, left first;
015/024	302		LAC		fetch middle digit to A;
015/025	025	MIDPRNT	RST	2	display two digits, left first;
015/026	304	LSTSIG	LAE		fetch 1's digit;
015/027	025		RST	2	display remaining digit;
015/030	066 310		LLI	L(MESS5)	point to first part of you win;
015/032	056 016	RECYC	LHI	H(MESS5)	second part of MESS5/MESS6 pointer;
015/034	106 134 015	PRNTIT	CAL	OUTPUT	display the message;
015/037	106 151 015		CAL	INPUT	fetch a character for continue
015/042	074 131		CPI	'Y'	query, is it "yes"?
015/044	150 052 014		JTZ	ASTART	if so then continue game;
015/047	015		RST	1	otherwise call EXIT;
015/050	074 000	LOSSTST	CPI	0	is fringe universe all black holes?
015/052	110 157 014		JFZ	GOTSTAR	if not then continue game;
015/055	302		LAC		if so then test center position;
015/056	074 000		CPI	0	is center also black hole?
015/060	110 157 014		JFZ	GOTSTAR	if not then continue game;
015/063	066 156		LLI	L(MESS3)	else point to loss message;
015/065	104 032 015		JMP	RECYC	and go print loss;
015/070	see Table II	MASKTAB	BLK	036D	36 bytes of mask table;
015/134	307	OUTPUT	LAM		fetch next message byte;
015/135	074 031		CPI	031	is it a delimiter?
015/137	053		RTZ		return when delimiter found;
015/140	025		RST	2	otherwise display byte;
015/141	060		INL		point to next byte;
015/142	110 134 015		JFZ	OUTPUT	is it page boundary?
015/145	050		INH		if so increment page;
015/146	104 134 015		JMP	OUTPUT	and then recycle;
015/151	005	INPUT	RST	0	get next character;
015/152	340		LEA		save it in E;
015/153	025		RST	2	echo on display;
015/154	005	GETNEXT	RST	0	get next character;
015/155	025		RST	2	echo on display;
015/156	074 012		CPI	012	was it a line feed?
015/160	110 154 015		JFZ	GETNEXT	if not continue scan;
015/163	304		LAE		if so, restore first input;
015/164	007		RET		and then return to caller;
015/165	066 000	BADFELO	LLI	L(MESS1)	point to the error message
015/167	056 016		LHI	H(MESS1)	admonishing bad 'star';
015/171	104 313 014		JMP	OUTMES	and go display error;

Notation:

L(HMESS) = low order 8 bits of address of HMESS;

H(HMESS) = high order 8 bits of address of HMESS;

'N' = the ASCII character "N";

9D = the decimal number 9;

7 = the octal number 7 (with high order zeros as needed);

mnemonics are from original Intel 8008 documentation;

octal code is shown in ascending address order top to bottom, left to right;

MODEL CC-7 SPECIFICATIONS:

A. Recording Mode: Tape saturation binary. This is not an FSK or Home type recorder. No voice capability. No Modem.

B. Two channels (1) Clock, (2) Data. OR, Two data channels providing four (4) tracks on the cassette. Can also be used for NRZ, Bi-Phase, etc.

C. Inputs: Two (2). Will accept TTY, TTL or RS 232 digital.

D. Outputs: Two (2). Board changeable from RS 232 to TTY or TTL digital.

E. Runs at 2400 baud or less. Synchronous or Asynchronous. Runs at 4800 baud Synchronous (simple external synchronizer diagram furnished.) Runs at 3.1"/sec. Speed regulations ± .5%.

F. Compatability: Will interface any computer or terminal with a serial I/O. (Altair, Sphere, M6800, PDP8, LSI 11, etc.)

G. Other Data: (110-220 V), (50-60 Hz); 2 Watts total; UL listed 955D; three wire line cord; on/off switch; audio, meter and light operation monitors. Remote control of motor optional. Four foot, seven conductor remoting cable provided. Uses high grade audio cassettes.

H. Warrantee: 90 days. All units tested at 110 and 2400 baud before shipment. Test cassette with 8080 software program included. This cassette was recorded and played back during quality control.

ALSO AVAILABLE: MODEL CC-7A with variable speed motor. Uses electronic speed control at 4"/sec. or less.
Runs at 4800 baud Synchronous or Asynchronous without external circuitry.
Recommended for quantity users who exchange tapes. Comes with speed adjusting tape to set exact speed.

DIGITAL DATA RECORDER $149.95
FOR COMPUTER or TELETYPE USE
Any baud rate up to 4800

Uses the industry standard tape saturation method to beat all FSK systems ten to one. No modems or FSK decoders required. Loads 8K of memory in 17 seconds. This recorder, using high grade audio cassettes, enables you to back up your computer by loading and dumping programs and data fast as you go, thus enabling you to get by with less memory. Can be software controlled.

Master Charge & BankAmericard accepted.

On orders for Recorders and Kits please add $2.00 for Shipping & Handling.
(N.J. Residents add 5% Sales Tax)

NATIONAL multiplex
CORPORATION

3474 Rand Avenue, Box 288
South Plainfield, New Jersey 07080
(201) 561-3600

NOW AVAILABLE
RECORD/PLAYBACK AMPLIFIER KIT

This expanded version of our Computer Aid board can be used with your own deck (cassette or reel to reel). Go to 9600 baud on reel to reel with suitable heads. Digital in, digital out, serial format. Kit includes all parts, case and power supply. Includes high baud rate synchronizer. $59.95

COMING SOON – IN KIT FORM

* I/O Board for use with Computer Aid or other digital recorders. Variable baud rate selectable on externally located unit by one knob. Can load computer or accept dumps without software, thus providing Turnkey Operation. For any 8 bit computer.

* Hexadecimal or Octal Keyboard – Load programs direct from keyboards' 20 keys and verifying display. Does not use Computer I/O. Can be wired Octal or Hex. – Your choice.

* Interested in these? Send your name and address for brochure when released.

Send One dollar for Cassette Operating and Maintenance Manual with Schematics and Software control data for 8080 and 6800. Also applies to Kit above. (Postpaid)