Star Raiders source code by Lorenz Wiest#

Lorenz Wiest did a tremendous must see job with a complete commented Star Raiders source code at the highest level possible! Highly recommended!!!
Every assembly source code should look like this. The reader can take his work as a template on how to do it the right way!

Source Code in TXT-Format#

- StarRaiders.source.txt ; size: 840 KB ; incredible source code listing **with** assembled hex bytes!!! A must see!!!
- StarRaiders.source.asm.txt ; size: 644 KB ; incredible source code listing **without** assembled hex bytes, ready to use in an assembler!!! A must see!!!

Read Me from Lorenz Wiest#

Reverse engineered and fully documented STAR RAIDERS source code
STAR RAIDERS is a seminal computer game, published by Atari Inc. in 1979 as one of the first titles for the original Atari 8-bit Home Computer System (Atari 400 and 800).
During the last years, as a hobby afterhours project, I reverse engineered a complete, extensively documented assembly language source code of STAR RAIDERS. I reverse engineered the source code directly from the binary file of the ROM cartridge, finishing it in September 2015.
This repo contains both the reverse enginered assembly source code input StarRaiders.source.asm.txt and output StarRaiders.source.txt.
You may find this useful if you are curious about the internals of STAR RAIDERS or if you are planning to make your own, modified version.
After very positive reader feedback on the initial release of my reverse engineered source code I reviewed it again, resulting in a minor update.
Enjoy -- Lorenz
P.S. I was absolutely thrilled to learn that in October 2015 scans of the original STAR RAIDERS source code re-surfaced.
To my delight, inspection of the original source code confirmed the findings of my reverse engineered version and caused only a few trivial corrections.
Even more, the documentation of my reverse engineered version adds a substantial amount of information - from overall theory of operation down to some tricky details - to the understanding of the often sparsely commented original (quite expected for source code never meant for publication).
So, if you are interested in learning how STAR RAIDERS works, my reverse engineered source code could be of help to you.

Extras#

Genome Sequence#

The picture below shows the 'genome sequence' of the reverse engineered STAR RAIDERS 8 KB ROM (which I prepared for a publication. I stacked the 8192 bytes vertically, with each byte represented by a tiny, solid horizontal line of 8 pixels. Then, I split the stack into strips of 192 bytes, arranged side-by-side. The segments of the stack are color-coded:

- Code (main game loop and subroutines) is in alternating shades of blue.
- Data (lookup tables, texts, etc.) is in alternating shades of green.
- Bitmap data (Player-Missile shapes and character set) is in alternating shades of purple and the solid line of a byte is replaced by its actual bit pattern.

STAR RAIDERS 8KB ROM Genome Sequence ; thank you so much Lorenz, that is totally incredible! :-)

ColorSheets#

- STAR RAIDERS Color Sheets ; size: 250 KB

DisplayListSheets#

- STAR RAIDERS Display List Sheets ; size: 250 KB

References#

- Original post of the source code
- AtariAge Forum - Fully documented, reverse-engineered STAR RAIDERS source code available
- Link at github of the source code
- Wikipedia 'Star Raiders'
- ANTIC The Atari 8-bit Podcast
- Star Raiders Tribute Page
- Brainwagon - Deconstructing the Classic Atari Game: Star Raiders
- Article 'Reverse Engineering Star Raiders', PoC GTFO 0x13, p. 5-20, October 2016

Source Code Listing#

```
00001 ;*****************************************************************
00002 ;*
00003 ;*                            S T A R   R A I D E R S
00004 ;*
00005 ;*                    for the Atari 8-bit Home Computer System
00006 ;*
00007 ;*       Reverse engineered and documented assembly language sourc
00008 ;*
00009 ;*                                      by
00010 ;*
00011 ;*                                 Lorenz Wiest
00012 ;*
00013 ;*                               (lo.wiest(at)web.de)
00014 ;*
00015 ;*                                 First Release
00016 ;*                                  22-SEP-2015
00017 ;*
00018 ;*                                  Last Update
00019 ;*                                  10-DEC-2016
00020 ;*
00021 ;*                  STAR RAIDERS was created by Douglas Neubauer
00022 ;*                    STAR RAIDERS was published by Atari Inc.
00023 ;*
00024 ;*****************************************************************
00025
00026 ; I wrote this document out of my own curiosity. When STAR RAIDERS
00027 ; in 1979 it became the killer app for the Atari 8-bit Home Comput
00028 ; Since then I have always been wondering what made it tick and ho
00029 ; time) spectacular 3D graphics worked, especially the rotating st
00030 ; Impressed by "The Atari BASIC Source Book" I decided to reverse
00031 ; STAR RAIDERS 8KB ROM cartridge to recreate a fully documented as
00032 ; language source code file. I had no access to the original sourc
00033 ; only way to succeed was a combination of educated guesses, trial
```

```
00034 ; and patience. Eventually, I made it.
00035 ;
00036 ; Essential in preparing this document were three programs I wrote
00037 ;
00038 ; (1) A 6502-cross-assembler based on the syntax of the MAC/65 ass
00039 ;     Atari 8-bit Home Computer System to create the binary file t
00040 ;     against the binary of the original ROM cartridge.
00041 ;
00042 ; (2) A text formatter to layout the source code file with its cop
00043 ;     sections. This was a big time saver, because as the document
00044 ;     source code had to be reformatted over and over again.
00045 ;
00046 ; (3) A symbol checker to verify that the ubiquitous symbol-value
00047 ;     documentation match the corresponding symbol values produced
00048 ;     assembler.
00049 ;
00050 ; This assembly language source code file is compatible with the M
00051 ; assembler for the Atari 8-bit Home Computer System. I was able t
00052 ; on an emulated Atari running MAC/65, producing the identical bin
00053 ; cartridge.
00054 ;
00055 ; Your feedback is welcome! Send feedback to lo.wiest(at)web.de.
00056 ;
00057 ; Enjoy! -- Lorenz
00058
00059 ;*****************************************************************
00060 ;*
00061 ;*                              N O T A T I O N
00062 ;*
00063 ;*****************************************************************
00064
00065 ; BITS AND BYTES
00066 ;
00067 ; o   A "byte" consists of 8 bits. They are numbered B7..0. Bit B0
00068 ;     significant bit.
00069 ;
00070 ; o   A "word" consists of 16 bits. They are numbered B15..B0. Bit
00071 ;     least significant bit. A word is stored in low-order then hi
00072 ;     order.
00073 ;
00074 ; o   The high-order byte ("high byte") of a word consists of bits
00075 ;     word.
00076 ;
00077 ; o   The low-order byte ("low byte") of a word consists of bits B
00078 ;     word.
00079 ;
00080 ; NUMBERS
00081 ;
00082 ; o   The dollar sign ($) prefixes hexadecimal numbers.
00083 ;     Example: $101 is the decimal number 257.
00084 ;
00085 ; o   The percent sign (%) prefixes binary numbers.
00086 ;     Example: %101 is the decimal number 5.
00087 ;
00088 ; o   The asterisk (*) is a wildcard character for a single hexade
00089 ;     binary digit.
00090 ;     Example: $0*00 is a placeholder for the numbers $0000, $0100
00091 ;
00092 ; o   The lowercase R (r) is a wildcard character for a single ran
```

```
00093 ;     hexadecimal or binary digit. The random digit r is chosen by
00094 ;     number generator.
00095 ;     Example: %00r0 is a placeholder for the numbers %0000 or %00
00096 ;
00097 ; OPERATORS
00098 ;
00099 ; o   The exclamation mark (!) is the binary OR operator.
00100 ;     Example: $01!$02 is $03.
00101 ;
00102 ; o   The less-than sign (<) indicates bits B7..0 of a word.
00103 ;     Example: <$1234 is $34.
00104 ;
00105 ; o   The greater-than sign (>) indicates bits B15..8 of a word.
00106 ;     Example: >$1234 is $12.
00107 ;
00108 ; o   A pair of brackets ([]) groups mathematical expressions.
00109 ;     Example: [3-1]*4 is 8.
00110 ;
00111 ; ASSEMBLY LANGUAGE
00112 ;
00113 ; o   The uppercase A (A) indicates the accumulator register of th
00114 ;
00115 ; o   The uppercase X (X) indicates the X register of the 6502 CPU
00116 ;
00117 ; o   The uppercase Y (Y) indicates the Y register of the 6502 CPU
00118 ;
00119 ; o   The prefix uppercase L and dot (L.) indicates a local variab
00120 ;     location used temporarily in a subroutine.
00121 ;
00122 ; PSEUDO-FUNCTIONS
00123 ;
00124 ; o   The function ABS(<num>) returns the absolute value of <num>.
00125 ;     Example: ABS(3) returns 3.
00126 ;     Example: ABS(-3) returns 3.
00127 ;
00128 ; o   The function RND(<num1>..<num2>) returns a random integer in
00129 ;     <num1>..<num2>.
00130 ;     Example: RND(3..5) returns a random number out of 3, 4, or 5
00131 ;
00132 ; o   The function MAX(<num1>,<num2>) returns the larger number of
00133 ;     <num2>.
00134 ;     Example: MAX(2,4) returns 4.
00135 ;
00136 ; VECTORS
00137 ;
00138 ; o   The lowercase X (x) indicates the x-axis of the 3D coordinat
00139 ;
00140 ; o   The lowercase Y (y) indicates the y-axis of the 3D coordinat
00141 ;
00142 ; o   The lowercase Z (z) indicates the z-axis of the 3D coordinat
00143 ;
00144 ; o   Components of a position vector (called "coordinates") have
00145 ;     unit <KM> ("kilometers").
00146 ;
00147 ; o   Components of a velocity vector have the arbitrary unit <KM/
00148 ;     ("kilometers per hour").
00149 ;
00150 ; o   A positive component of a position vector (coordinate) in he
00151 ;     notation is written in the form +$<hexNum> <KM>. <hexNum> is
```

```
00152 ;     integer value.
00153 ;     Example: The starbase is +$1000 (or 4096) <KM> ahead of our
00154 ;
00155 ; o   A negative component of a position vector (coordinate) in he
00156 ;     notation is written in the form -($<hexNum>) <KM>. <hexNum>
00157 ;     integer value. To calculate the actual bit pattern of this c
00158 ;     value compute the two's-complement of <hexNum>. See also "ON
00159 ;     VECTORS".
00160 ;     Example: The starbase is -($1000) (or -4096) <KM> behind our
00161 ;
00162 ; o   An absolute component of a position vector (coordinate) in h
00163 ;     notation is written in the form $<hexNum> <KM>. <hexNum> is
00164 ;     integer value.
00165 ;     Example: The Zylon fighter fires when it is closer than $100
00166 ;              <KM>.
00167 ;
00168 ; DISPLAY LIST
00169 ;
00170 ; o   The following notation is used for Display List instructions
00171 ;
00172 ;     BLK<n>           = Display <n> blank video lines (<n> in 1..
00173 ;     GR1              = Display one GRAPHICS 1 row of 20 text cha
00174 ;     GR2              = Display one GRAPHICS 2 row of 20 text cha
00175 ;     GR7              = Display one GRAPHICS 7 row of 160 pixels
00176 ;     DLI              = Trigger a Display List Interrupt
00177 ;     ... @ <addr>     = Point to screen memory at address <addr>
00178 ;     JMP @ <addr>     = Jump to next Display List instruction at
00179 ;     WAITJMP @ <addr> = Wait for vertical blank phase, then jump
00180 ;                        Display List instruction at address <addr
00181 ;
00182 ; MISCELLANEOUS
00183 ;
00184 ; o   Probabilities are written in the form <percentage>% (<number
00185 ;     of the possible values>:<number of possible values>).
00186 ;     Example: The probability to throw the number 3 with a die is
00187 ;
00188 ; o   A "game loop iteration" (or "game loop") is a single executi
00189 ;     loop, the main program of the game.
00190 ;
00191 ; o   A "TICK" is the time span it takes to update the TV screen (
00192 ;     NTSC TV system, 1/50 s on a PAL TV system).
00193 ;
00194 ; o   A pair of braces ({}) encloses color names.
00195 ;     Example: {BLACK}
00196 ;
00197 ; o   A pair of parentheses enclosing a question mark ((?)) indica
00198 ;     is not well understood.
00199 ;
00200 ; o   A pair of parentheses enclosing an exclamation mark ((!)) in
00201 ;     potential bug.
00202
00203 ;*****************************************************************
00204 ;*
00205 ;*                              O V E R V I E W
00206 ;*
00207 ;*****************************************************************
00208
00209 ; ON POSITION VECTORS
00210 ;
```

```
00211 ; The game uses a 3D coordinate system with the position of our st
00212 ; center of the coordinate system and the following coordinate axe
00213 ;
00214 ; o   The x-axis points to the right.
00215 ; o   The y-axis points up.
00216 ; o   The z-axis points in flight direction.
00217 ;
00218 ; By the way, this is called a "left-handed" coordinate system.
00219 ;
00220 ; The locations of all space objects (Zylon ships, meteors, photon
00221 ; starbase, transfer vessel, Hyperwarp Target Marker, stars, and e
00222 ; fragments) are described by a "position vector".
00223 ;
00224 ; A "position vector" is composed of an x, y, and z component. The
00225 ; position vector components are called the x, y, and z "coordinat
00226 ; the arbitrary unit <KM>.
00227 ;
00228 ; Each coordinate is a signed 17-bit integer number, which fits in
00229 ;
00230 ;     Sign     Mantissa
00231 ;          B16 B15...B8 B7....B0
00232 ;            | |      | |      |
00233 ;     0000000* ******** ********
00234 ;
00235 ; o   B16 contains the sign bit. Used values are:
00236 ;       1 -> Positive sign
00237 ;       0 -> Negative sign
00238 ; o   B15..0 contain the coordinate value (or "mantissa") as a two
00239 ;     integer number.
00240 ;
00241 ; The range of a position vector component is -65536..+65535 <KM>.
00242 ;
00243 ; Examples:
00244 ;
00245 ;     00000001 11111111 11111111 = +65535 <KM>
00246 ;     00000001 00010000 00000000 =  +4096 <KM>
00247 ;     00000001 00001111 11111111 =  +4095 <KM>
00248 ;     00000001 00000001 00000000 =   +256 <KM>
00249 ;     00000001 00000000 11111111 =   +255 <KM>
00250 ;     00000001 00000000 00010000 =    +16 <KM>
00251 ;     00000001 00000000 00001111 =    +15 <KM>
00252 ;     00000001 00000000 00000001 =     +1 <KM>
00253 ;     00000001 00000000 00000000 =     +0 <KM>
00254 ;
00255 ;     00000000 11111111 11111111 =     -1 <KM>
00256 ;     00000000 11111111 11111110 =     -2 <KM>
00257 ;     00000000 11111111 11110001 =    -15 <KM>
00258 ;     00000000 11111111 11110000 =    -16 <KM>
00259 ;     00000000 11111111 00000001 =   -255 <KM>
00260 ;     00000000 11111111 00000000 =   -256 <KM>
00261 ;     00000000 11110000 00000001 =  -4095 <KM>
00262 ;     00000000 11110000 00000000 =  -4096 <KM>
00263 ;     00000000 00000000 00000000 = -65536 <KM>
00264 ;
00265 ; The position vector for each space object is stored in 9 tables:
00266 ;
00267 ; o   XPOSSIGN ($09DE..$0A0E), XPOSHI ($0A71..$0AA1), and XPOSLO (
00268 ; o   YPOSSIGN ($0A0F..$0A3F), YPOSHI ($0AA2..$0AD2), and YPOSLO (
00269 ; o   ZPOSSIGN ($09AD..$09DD), ZPOSHI ($0A40..$0A70), and ZPOSLO (
```

```
00270 ;
00271 ; There are up to 49 space objects used in the game simultaneously
00272 ; table is 49 bytes long.
00273 ;
00274 ; o   Position vectors 0..4 belong to space objects represented by
00275 ;     (Zylon ships, meteors, photon torpedoes, starbase, transfer
00276 ;     Hyperwarp Target Marker).
00277 ; o   Position vectors 5..48 belong to space objects represented b
00278 ;     pixels. Position vectors 5..16 (stars, explosion fragments)
00279 ;     stars, position vectors 17..48 are used for explosion fragme
00280 ;     trails.
00281 ;
00282 ; INFO: The x and y coordinates of space objects are converted and
00283 ; the THETA and PHI readouts of the Control Panel Display in "grad
00284 ; z-coordinate is converted and displayed by the RANGE readout in
00285 ; The conversion takes place in subroutine SHOWDIGITS ($B8CD) wher
00286 ; byte of a coordinate (with values $00..$FF) is transformed with
00287 ; MAPTOBCD99 ($0EE9) into a BCD value of 00..99 in "gradons" or "c
00288 ;
00289 ;
00290 ; ON VELOCITY VECTORS
00291 ;
00292 ; The velocities of all space objects are described by a "velocity
00293 ; velocity vector is relative to our starship.
00294 ;
00295 ; A "velocity vector" is composed of an x, y, and z component. The
00296 ; velocity vector components are called the x, y, and z "velocitie
00297 ; the arbitrary unit <KM/H>.
00298 ;
00299 ; Each velocity vector component is an 8-bit integer number, which
00300 ; byte:
00301 ;
00302 ;     B7 Sign
00303 ;     |
00304 ;     |B6...B0 Mantissa
00305 ;     ||     |
00306 ;     ********
00307 ;
00308 ; o   B7 contains the sign bit. Used values are:
00309 ;     0 -> Positive sign, movement along the positive coordinate a
00310 ;          (x-velocity: right, y-velocity: up, z-velocity: in flig
00311 ;     1 -> Negative sign, movement along the negative coordinate a
00312 ;          (x-velocity: left, y-velocity: down, z-velocity: in rev
00313 ;          direction)
00314 ; o   B6..B0 contain the velocity value (or "mantissa"). It is an
00315 ;     number.
00316 ;
00317 ; The range of a velocity vector component is -127..+127 <KM/H>.
00318 ;
00319 ; Examples:
00320 ;
00321 ;     01111111 = +127 <KM/H>
00322 ;     00010000 =  +16 <KM/H>
00323 ;     00001111 =  +15 <KM/H>
00324 ;     00000001 =   +1 <KM/H>
00325 ;     00000000 =   +0 <KM/H>
00326 ;
00327 ;     10000000 =   -0 <KM/H>
00328 ;     10000001 =   -1 <KM/H>
```

```
00329 ;     10001111 =  +15 <KM/H>
00330 ;     10010000 =  +16 <KM/H>
00331 ;     11111111 = -127 <KM/H>
00332 ;
00333 ; The velocity vector for each space object stored in 3 tables:
00334 ;
00335 ; o   XVEL ($0B97..$0BC7)
00336 ; o   YVEL ($0BC8..$0BF8)
00337 ; o   ZVEL ($0B66..$0B96)
00338 ;
00339 ; There are up to 49 space objects used in the game simultaneously
00340 ; table is 49 bytes long.
00341 ;
00342 ; o   Velocity vectors 0..4 belong to space objects represented by
00343 ;     (Zylon ships, meteors, photon torpedoes, starbase, transfer
00344 ;     Hyperwarp Target Marker).
00345 ; o   Velocity vectors 5..48 belong to space objects represented b
00346 ;     pixels. Velocity vectors 5..16 are used for stars, velocity
00347 ;     are used for explosion fragments and star trails.
00348 ;
00349 ; INFO: The velocity of our starship is converted and displayed by
00350 ; readout of the Control Panel Display in "metrons per second" uni
00351 ; conversion takes place in subroutine SHOWDIGITS ($B8CD) where ou
00352 ; velocity VELOCITYL ($70) (with values $00..$FF) is transformed w
00353 ; table MAPTOBCD99 ($0EE9) into a BCD value of 00..99 in "metrons
00354
00355 ;*****************************************************************
00356 ;*
00357 ;*                              M E M O R Y   M A P
00358 ;*
00359 ;*****************************************************************
00360 ;
00361 ; The following variables are not changed by a SYSTEM RESET:
00362 ;
00363 ; $62      MISSIONLEVEL
00364 ;
00365 ;          Mission level. Used values are:
00366 ;            $00 -> NOVICE mission
00367 ;            $01 -> PILOT mission
00368 ;            $02 -> WARRIOR mission
00369 ;            $03 -> COMMANDER mission
00370 ;
00371 ; $63      FKEYCODE
00372 ;
00373 ;          Function key code. Used values are:
00374 ;            $00 -> No function key pressed
00375 ;            $01 -> START function key pressed
00376 ;            $02 -> SELECT function key pressed
00377 ;
00378 ; $64      ISDEMOMODE
00379 ;
00380 ;          Indicates whether the game is in game or in demo mode.
00381 ;          are:
00382 ;            $00 -> Game mode
00383 ;            $FF -> Demo mode
00384 ;
00385 ; $65      NEWTITLEPHR
00386 ;
00387 ;          New title phrase offset for the text in the title line.
```

```
00388 ;          phrase is not immediately displayed in the title line b
00389 ;          the display time of the currently displayed title phras
00390 ;          Thus, setting a value to NEWTITLEPHR ($65) "enqueues" t
00391 ;          new title phrase. Used values are:
00392 ;            $00..$7B -> Title phrase offset into PHRASETAB ($BBAA
00393 ;            $FF      -> Hide title line
00394 ;
00395 ;          See also TITLEPHR ($D1).
00396 ;
00397 ; $66      IDLECNTHI
00398 ;
00399 ;          Idle counter (high byte). Forms a 16-bit counter togeth
00400 ;          IDLECNTLO ($77), which is incremented during the execut
00401 ;          Vertical Blank Interrupt handler VBIHNDLR ($A6D1). IDLE
00402 ;          reset to 0 when the joystick trigger or a keyboard key
00403 ;          pressed, or to 1..3 when a function key has been presse
00404 ;          IDLECNTHI ($66) reaches a value of 128 (after about 10
00405 ;          the game enters demo mode.
00406 ;
00407 ; The following variables are set to 0 after a SYSTEM RESET:
00408 ;
00409 ; $67      ISVBISYNC
00410 ;
00411 ;          Indicates whether the Vertical Blank Interrupt handler
00412 ;          ($A6D1) is executed. Used to synchronize the execution
00413 ;          loop iteration in GAMELOOP ($A1F3) with the vertical bl
00414 ;          Used values are:
00415 ;            $00 -> Halt execution at start of game loop and wait
00416 ;            $FF -> Continue execution of game loop
00417 ;
00418 ; $68..$69 MEMPTR
00419 ;
00420 ;          A 16-bit memory pointer.
00421 ;
00422 ;          Also used as a local variable.
00423 ;
00424 ; $6A..$6B DIVIDEND
00425 ;
00426 ;          A 16-bit dividend value passed in GAMELOOP ($A1F3) to s
00427 ;          PROJECTION ($AA21) to calculate a division.
00428 ;
00429 ;          Also used as a local variable.
00430 ;
00431 ; $6C      Used as a local variable.
00432 ;
00433 ; $6D      JOYSTICKDELTA
00434 ;
00435 ;          Used to pass joystick directions from GAMELOOP ($A1F3)
00436 ;          ROTATE ($B69B). Used values are:
00437 ;            $01 -> Joystick pressed right or up
00438 ;            $00 -> Joystick centered
00439 ;            $FF -> Joystick pressed left or down
00440 ;
00441 ;          Also used as a local variable.
00442 ;
00443 ; $6E      Used as a local variable.
00444 ;
00445 ; $70      VELOCITYLO
00446 ;
```

```
00447 ;          Our starship's current velocity (low byte) in <KM/H>. F
00448 ;          value together with VELOCITYHI ($C1). In subroutine UPD
00449 ;          VELOCITYLO ($70) is mapped to a BCD-value in 00..99 and
00450 ;          the VELOCITY readout of the Control Panel Display. See
00451 ;          NEWVELOCITY ($71).
00452 ;
00453 ; $71      NEWVELOCITY
00454 ;
00455 ;          Our starship's new velocity (low byte) in <KM/H>. It is
00456 ;          pressing one of the speed keys '0'..'9'. A pressed spee
00457 ;          mapped to the new velocity value with VELOCITYTAB ($BAB
00458 ;
00459 ; $72      COUNT8
00460 ;
00461 ;          Wrap-around counter. Counts from 0..7, then starts over
00462 ;          incremented every game loop iteration. It is used to ch
00463 ;          brightness of stars and explosion fragments more random
00464 ;          ($A1F3) and to slow down the movement of the hyperwarp
00465 ;          Galactic Chart in subroutine SELECTWARP ($B162).
00466 ;
00467 ; $73      EXPLLIFE
00468 ;
00469 ;          Explosion lifetime. It is decremented every game loop i
00470 ;          values are:
00471 ;              $00 -> Explosion is over
00472 ;            < $18 -> Number of explosion fragment space objects i
00473 ;            < $70 -> HITBADNESS ($8A) is reset
00474 ;              $80 -> Initial value at start of explosion
00475 ;
00476 ; $74      CLOCKTIM
00477 ;
00478 ;          Star date clock delay timer. Counts down from 40 to 0.
00479 ;          decremented every game loop iteration. When the timer f
00480 ;          the last digit of the star date of the Galactic Chart P
00481 ;          is increased and the timer is reset to a value of 40.
00482 ;
00483 ; $75      DOCKSTATE
00484 ;
00485 ;          State of docking operation. Used values are:
00486 ;            $00 -> NOT DOCKED
00487 ;            $01 -> TRANSFER COMPLETE
00488 ;            $81 -> RETURN TRANSFER VESSEL
00489 ;            $FF -> ORBIT ESTABLISHED
00490 ;
00491 ; $76      COUNT256
00492 ;
00493 ;          Wrap-around counter. Counts from 0..255, then starts ov
00494 ;          incremented every game loop iteration. It is used to ma
00495 ;          starbase pulsate in brightness in GAMELOOP ($A1F3) and
00496 ;          the creation of a meteor in subroutine MANEUVER ($AA79)
00497 ;
00498 ; $77      IDLECNTLO
00499 ;
00500 ;          Idle counter (low byte). Forms a 16-bit counter togethe
00501 ;          IDLECNTHI ($66), which is incremented during the execut
00502 ;          Vertical Blank Interrupt handler VBIHNDLR ($A6D1).
00503 ;
00504 ;          NOTE: This variable is never properly initialized excep
00505 ;          cartridge startup (cold start).
```

```
00506 ;
00507 ; $78      ZYLONUNITTIM
00508 ;
00509 ;          Zylon unit movement timer. This delay timer triggers mo
00510 ;          Zylon units on the Galactic Chart. At the start of the
00511 ;          timer is initialized to a value of 100. It is decrement
00512 ;          game loop iterations. When the timer falls below 0 the
00513 ;          move on the Galactic Chart and the timer value is reset
00514 ;          starbase is surrounded the timer is reset to 99 to buy
00515 ;          time to destroy one of the surrounding Zylon units.
00516 ;
00517 ; $79      MAXSPCOBJIND
00518 ;
00519 ;          Maximum index of used space objects in the current game
00520 ;          iteration. Frequently used values are:
00521 ;            $10 -> During regular cruise (5 PLAYER space objects
00522 ;                   space objects (stars), counted $00..$10)
00523 ;            $30 -> During explosion or hyperwarp (5 PLAYER space
00524 ;                   PLAYFIELD space objects (stars) + 32 PLAYFIELD
00525 ;                   (explosion fragments or stars of star trails),
00526 ;                   $00..$30)
00527 ;
00528 ; $7A      OLDMAXSPCOBJIND
00529 ;
00530 ;          Maximum index of used space objects in the previous gam
00531 ;          iteration. Frequently used values are:
00532 ;            $10 -> During regular cruise (5 PLAYER space objects
00533 ;                   space objects (stars), counted $00..$10)
00534 ;            $30 -> During explosion or hyperwarp (5 PLAYER space
00535 ;                   PLAYFIELD space objects (stars) + 32 PLAYFIELD
00536 ;                   (explosion fragments or stars of star trails),
00537 ;                   $00..$30)
00538 ;
00539 ; $7B      ISSTARBASESECT
00540 ;
00541 ;          Indicates whether a starbase is in this sector. Used va
00542 ;            $00 -> Sector contains no starbase
00543 ;            $FF -> Sector contains starbase
00544 ;
00545 ; $7C      ISTRACKCOMPON
00546 ;
00547 ;          Indicates whether the Tracking Computer is on or off. U
00548 ;          are:
00549 ;            $00 -> Tracking Computer is off
00550 ;            $FF -> Tracking Computer is on
00551 ;
00552 ; $7D      DRAINSHIELDS
00553 ;
00554 ;          Energy drain rate of the Shields per game loop iteratio
00555 ;          subunits. See also subroutine UPDPANEL ($B804). Used va
00556 ;            $00 -> Shields are off
00557 ;            $08 -> Shields are on
00558 ;
00559 ; $7E      DRAINATTCOMP
00560 ;
00561 ;          Energy drain rate of the Attack Computer per game loop
00562 ;          energy subunits. See also subroutine UPDPANEL ($B804).
00563 ;          are:
00564 ;            $00 -> Attack Computer off
```

```
00565 ;           $02 -> Attack Computer on
00566 ;
00567 ; $7F      ENERGYCNT
00568 ;
00569 ;          Running counter of consumed energy subunits (256 energy
00570 ;          energy unit displayed by the 4-digit ENERGY readout of
00571 ;          Panel Display). Forms an invisible fractional or "decim
00572 ;          the 4-digit ENERGY readout of the Control Panel Display
00573 ;          subroutine UPDPANEL ($B804).
00574 ;
00575 ; $80      DRAINENGINES
00576 ;
00577 ;          Energy drain rate of our starship's Engines per game lo
00578 ;          in energy subunits (256 energy subunits = 1 energy unit
00579 ;          the 4-digit ENERGY readout of the Control Panel Display
00580 ;          picked from table DRAINRATETAB ($BAD3). See also subrou
00581 ;          ($B804).
00582 ;
00583 ; $81      SHIELDSCOLOR
00584 ;
00585 ;          Shields color. Used values are:
00586 ;            $00 -> {BLACK} (Shields are off)
00587 ;            $A0 -> {DARK GREEN} (Shields are on)
00588 ;
00589 ; $82      PL3HIT
00590 ;
00591 ;          Collision register of PLAYER3 (usually our starship's p
00592 ;          0) with other PLAYERs. Used values are:
00593 ;              $00 -> No collision
00594 ;            > $00 -> PLAYER3 has collided with another PLAYER spa
00595 ;                     subroutine COLLISION ($AF3D) for details whi
00596 ;                     been hit by PLAYER3.
00597 ;
00598 ; $83      PL4HIT
00599 ;
00600 ;          Collision register of PLAYER4 (usually our starship's p
00601 ;          1) with other PLAYERs. Used values are:
00602 ;              $00 -> No collision
00603 ;            > $00 -> PLAYER4 has collided with another PLAYER spa
00604 ;                     subroutine COLLISION ($AF3D) for details whi
00605 ;                     been hit by PLAYER4.
00606 ;
00607 ; $84      OLDTRIG0
00608 ;
00609 ;          Joystick trigger state. Used values are:
00610 ;            $00 -> Joystick trigger was pressed
00611 ;            $01 -> Joystick trigger was not pressed
00612 ;            $AA -> Joystick trigger was "virtually" pressed (will
00613 ;                   another of our starship's photon torpedoes, se
00614 ;                   TRIGGER ($AE29).
00615 ;
00616 ; $86      ISTRACKING
00617 ;
00618 ;          Indicates whether one of our starship's photon torpedoe
00619 ;          tracking (homing in on) the target space object. Used v
00620 ;              $00 -> No target space object tracked. Our starship
00621 ;                     torpedoes will fly just straight ahead.
00622 ;            > $00 -> Tracking a target space object. Our starship
00623 ;                     torpedoes will home in on the tracked space
```

```
00624 ;
00625 ; $87      BARRELNR
00626 ;
00627 ;          Barrel from which our starship's next photon torpedo wi
00628 ;          launched. Used values are:
00629 ;            $00 -> Left barrel
00630 ;            $01 -> Right barrel
00631 ;
00632 ; $88      LOCKONLIFE
00633 ;
00634 ;          Lifetime of target lock-on. A target remains in lock-on
00635 ;          LOCKONLIFE ($88) counts down from 12 to 0. It is decrem
00636 ;          game loop iteration.
00637 ;
00638 ; $89      PLTRACKED
00639 ;
00640 ;          Index of currently tracked PLAYER. It is copied in subr
00641 ;          ($AE29) from TRACKDIGIT ($095C). Used values are:
00642 ;            $00 -> Track Zylon ship 0
00643 ;            $01 -> Track Zylon ship 1
00644 ;            $02 -> Track starbase during docking operations
00645 ;            $03 -> Track Hyperwarp Target Marker during hyperwarp
00646 ;
00647 ; $8A      HITBADNESS
00648 ;
00649 ;          Severity of a Zylon photon torpedo hit. Used values are
00650 ;            $00 -> NO HIT
00651 ;            $7F -> SHIELDS HIT
00652 ;            $FF -> STARSHIP DESTROYED
00653 ;
00654 ; $8B      REDALERTLIFE
00655 ;
00656 ;          Lifetime of red alert. It decreases from 255 to 0. It i
00657 ;          every game loop iteration.
00658 ;
00659 ; $8C      WARPDEPRROW
00660 ;
00661 ;          Departure hyperwarp marker row number on the Galactic C
00662 ;          given in Player/Missile pixels relative to the top Gala
00663 ;          border. It is initialized to a value of $47 (vertical c
00664 ;          Galactic Chart). Divide this value by 16 to get the dep
00665 ;          row number. Used values are: $00..$7F.
00666 ;
00667 ; $8D      WARPDEPRCOLUMN
00668 ;
00669 ;          Departure hyperwarp marker column number on the Galacti
00670 ;          given in Player/Missile pixels relative to the left Gal
00671 ;          border and initialized to a value of $43 (horizontal ce
00672 ;          Galactic Chart). Divide this value by 8 to get the depa
00673 ;          column number. Used values are: $00..$7F.
00674 ;
00675 ; $8E      WARPARRVROW
00676 ;
00677 ;          Arrival hyperwarp marker row number on the Galactic Cha
00678 ;          Player/Missile pixels relative to top Galactic Chart bo
00679 ;          initialized to a value of $47 (vertical center of Galac
00680 ;          Divide this value by 16 to get the arrival sector row n
00681 ;          values are: $00..$7F.
00682 ;
```

```
00683 ; $8F      WARPARRVCOLUMN
00684 ;
00685 ;          Arrival hyperwarp marker column number on the Galactic
00686 ;          Player/Missile pixels relative to left Galactic Chart b
00687 ;          initialized to a value of $43 (horizontal center of Gal
00688 ;          Divide this value by 8 to get the arrival sector column
00689 ;          values are: $00..$7F.
00690 ;
00691 ; $90      CURRSECTOR
00692 ;
00693 ;          Galactic Chart sector of the current location of our st
00694 ;          start of the game it is initialized to a value of $48.
00695 ;          are: $00..$7F with, for example,
00696 ;            $00 -> NORTHWEST corner sector
00697 ;            $0F -> NORTHEAST corner sector
00698 ;            $70 -> SOUTHWEST corner sector
00699 ;            $7F -> SOUTHWEST corner sector
00700 ;
00701 ;          See also ARRVSECTOR ($92).
00702 ;
00703 ; $91      WARPENERGY
00704 ;
00705 ;          Energy required to hyperwarp between the departure and
00706 ;          hyperwarp marker locations on the Galactic Chart divide
00707 ;          Values are picked from table WARPENERGYTAB ($BADD). Mul
00708 ;          value by 10 to get the actual value in energy units dis
00709 ;          Galactic Chart Panel Display.
00710 ;
00711 ; $92      ARRVSECTOR
00712 ;
00713 ;          Galactic Chart arrival sector of our starship after hyp
00714 ;          values are: $00..$7F with, for example,
00715 ;            $00 -> NORTHWEST corner sector
00716 ;            $0F -> NORTHEAST corner sector
00717 ;            $70 -> SOUTHWEST corner sector
00718 ;            $7F -> SOUTHWEST corner sector
00719 ;
00720 ;          See also CURRSECTOR ($90).
00721 ;
00722 ; $93      HUNTSECTOR
00723 ;
00724 ;          Galactic Chart sector of the starbase toward which the
00725 ;          are currently moving. Used values are: $00..$7F with, f
00726 ;            $00 -> NORTHWEST corner sector
00727 ;            $0F -> NORTHEAST corner sector
00728 ;            $70 -> SOUTHWEST corner sector
00729 ;            $7F -> SOUTHWEST corner sector
00730 ;
00731 ; $94      HUNTSECTCOLUMN
00732 ;
00733 ;          Galactic Chart sector column number of the starbase tow
00734 ;          Zylon units are currently moving. Used values are: 0..1
00735 ;
00736 ; $95      HUNTSECTROW
00737 ;
00738 ;          Galactic Chart sector row number of the starbase toward
00739 ;          Zylon units are currently moving. Used values are: 0..7
00740 ;
00741 ; $96..$9E NEWZYLONDIST
```

```
00742 ;
00743 ;         Table of distances between a Zylon unit and the hunted
00744 ;         the Zylon unit is tentatively moved in one of the 9 pos
00745 ;         directions NORTH, NORTHWEST, WEST, SOUTHWEST, SOUTH, SO
00746 ;         NORTHEAST, CENTER. Used to decide into which sector the
00747 ;         should move.
00748 ;
00749 ; $9E     OLDZYLONDIST
00750 ;
00751 ;         Current distance between the Zylon unit and the hunted
00752 ;
00753 ; $9F     HUNTTIM
00754 ;
00755 ;         Delay timer for Zylon units to decide on which starbase
00756 ;         counts down from 7. It is decremented every game loop i
00757 ;         the timer falls below 0 the Zylon units re-decide towar
00758 ;         starbase to move.
00759 ;
00760 ; $A0     BLIPCOLUMN
00761 ;
00762 ;         Top-left screen pixel column number of blip shape displ
00763 ;         Attack Computer Display. Used in subroutine UPDATTCOMP
00764 ;         values are: 120..142.
00765 ;
00766 ; $A1     BLIPROW
00767 ;
00768 ;         Top-left screen pixel row number of blip shape displaye
00769 ;         Attack Computer Display. Used in subroutine UPDATTCOMP
00770 ;         values are: 71..81.
00771 ;
00772 ; $A2     BLIPCYCLECNT
00773 ;
00774 ;         Blip cycle counter. It controls drawing the blip shape
00775 ;         Computer Display. Its value is incremented every game l
00776 ;         Used in subroutine UPDATTCOMP ($A7BF). Used values are:
00777 ;           $00..$04 -> Draw 0..4th row of blip shape
00778 ;           $05..$09 -> Do not draw blip shape (delay)
00779 ;           $0A      -> Recalculate blip shape position, erase At
00780 ;                       Display
00781 ;
00782 ; $A3     ISINLOCKON
00783 ;
00784 ;         Indicates whether the tracked space object is currently
00785 ;         lock-on (horizontally and vertically centered as well a
00786 ;         the Attack Computer Display. If so, all lock-on markers
00787 ;         the Attack Computer Display and our starship's launched
00788 ;         torpedoes will home in on the tracked space object. Use
00789 ;           $00 -> Not in lock-on
00790 ;           $A0 -> In lock-on
00791 ;
00792 ; $A4     DIRLEN
00793 ;
00794 ;         Used to pass the direction and length of a single line
00795 ;         the PLAYFIELD. Used in subroutines DRAWLINES ($A76F), D
00796 ;         ($A782), and UPDATTCOMP ($A7BF). Used values are:
00797 ;           Bit B7 = 0 -> Draw right
00798 ;           Bit B7 = 1 -> Draw down
00799 ;           Bits B6..0 -> Length of line in pixels.
00800 ;
```

```
00801 ;          See also PENROW ($A5) and PENCOLUMN ($A6).
00802 ;
00803 ; $A5      PENROW
00804 ;
00805 ;          Used to pass the start screen pixel row number of the l
00806 ;          drawn in the PLAYFIELD. Used in subroutines DRAWLINES (
00807 ;          DRAWLINE ($A782), and UPDATTCOMP ($A7BF).
00808 ;
00809 ; $A6      PENCOLUMN
00810 ;
00811 ;          Used to pass the start screen pixel column number of th
00812 ;          drawn in the PLAYFIELD. Used in subroutines DRAWLINES (
00813 ;          DRAWLINE ($A782), and UPDATTCOMP ($A7BF).
00814 ;
00815 ; $A7      CTRLDZYLON
00816 ;
00817 ;          Index of Zylon ship currently controlled by the game. U
00818 ;          subroutine MANEUVER ($AA79). The value is toggled every
00819 ;          loop iteration. Used values are:
00820 ;            $00 -> Control Zylon ship 0.
00821 ;            $01 -> Control Zylon ship 1.
00822 ;
00823 ; $A8      ZYLONFLPAT0
00824 ;
00825 ;          Flight pattern of Zylon ship 0. Used in subroutine MANE
00826 ;          Used values are:
00827 ;            $00 -> Attack flight pattern "0"
00828 ;            $01 -> Flight pattern "1"
00829 ;            $04 -> Flight pattern "4"
00830 ;
00831 ; $A9      ZYLONFLPAT1
00832 ;
00833 ;          Flight pattern of Zylon ship 1. Compare ZYLONFLPAT0 ($A
00834 ;
00835 ; $AA      MILESTTIM0
00836 ;
00837 ;          Delay timer of the milestone velocity indices of Zylon
00838 ;          in subroutine MANEUVER ($AA79).
00839 ;
00840 ;          When Zylon ship 0 is active, this value is decremented
00841 ;          loop iteration. If it falls below 0 then the milestone
00842 ;          indices of Zylon ship 0 are recalculated. When Zylon sh
00843 ;          controlled by the computer for the first time, the time
00844 ;          initial value of 1, later to an initial value of 120.
00845 ;
00846 ; $AB      MILESTTIM1
00847 ;
00848 ;          Delay timer of the milestone velocity index vector of Z
00849 ;          Compare MILESTTIM0 ($AA).
00850 ;
00851 ; $AC      MILESTVELINDZ0
00852 ;
00853 ;          Milestone z-velocity index of Zylon ship 0. Used in sub
00854 ;          MANEUVER ($AA79). The current z-velocity index of Zylon
00855 ;          ZYLONVELINDZ0 ($B2) is compared with this index and gra
00856 ;          adjusted to it. Used values are: 0..15.
00857 ;
00858 ; $AD      MILESTVELINDZ1
00859 ;
```

```
00860 ;          Milestone z-velocity index of Zylon ship 1. Compare MIL
00861 ;          ($AC).
00862 ;
00863 ; $AE      MILESTVELINDX0
00864 ;
00865 ;          Milestone x-velocity index of Zylon ship 0. Used in sub
00866 ;          MANEUVER ($AA79). The current x-velocity index of Zylon
00867 ;          ZYLONVELINDX0 ($B4) is compared with this index and gra
00868 ;          adjusted to it. Used values are: 0..15.
00869 ;
00870 ; $AF      MILESTVELINDX1
00871 ;
00872 ;          Milestone x-velocity index of Zylon ship 1. Compare MIL
00873 ;          ($AE).
00874 ;
00875 ; $B0      MILESTVELINDY0
00876 ;
00877 ;          Milestone y-velocity index of Zylon ship 0. Used in sub
00878 ;          MANEUVER ($AA79). The current y-velocity index of Zylon
00879 ;          ZYLONVELINDY0 ($B6) is compared with this index and gra
00880 ;          adjusted to it. Used values are: 0..15.
00881 ;
00882 ; $B1      MILESTVELINDY1
00883 ;
00884 ;          Milestone y-velocity index of Zylon ship 1. Compare MIL
00885 ;          ($B0).
00886 ;
00887 ; $B2      ZYLONVELINDZ0
00888 ;
00889 ;          Current z-velocity index of Zylon ship 0. Used in subro
00890 ;          ($AA79). It indexes velocity values in ZYLONVELTAB ($BF
00891 ;          values are: 0..15.
00892 ;
00893 ; $B3      ZYLONVELINDZ1
00894 ;
00895 ;          Current z-velocity index of Zylon ship 1. Compare ZYLON
00896 ;          ($B2).
00897 ;
00898 ; $B4      ZYLONVELINDX0
00899 ;
00900 ;          Current x-velocity index of Zylon ship 0. Compare ZYLON
00901 ;          ($B2).
00902 ;
00903 ; $B5      ZYLONVELINDX1
00904 ;
00905 ;          Current x-velocity index of Zylon ship 1. Compare ZYLON
00906 ;          ($B2).
00907 ;
00908 ; $B6      ZYLONVELINDY0
00909 ;
00910 ;          Current y-velocity index of Zylon ship 0. Compare ZYLON
00911 ;          ($B2).
00912 ;
00913 ; $B7      ZYLONVELINDY1
00914 ;
00915 ;          Current y-velocity index of Zylon ship 1. Compare ZYLON
00916 ;          ($B2).
00917 ;
00918 ; $B8      ISBACKATTACK0
```

```
00919 ;
00920 ;          Indicates whether Zylon ship 0 will attack our starship
00921 ;          back. Used in subroutine MANEUVER ($AA79). Used values
00922 ;            $00 -> Zylon ship 0 attacks from the front of our sta
00923 ;            $01 -> Zylon ship 0 attacks from the front and back o
00924 ;
00925 ; $B9      ISBACKATTACK1
00926 ;
00927 ;          Indicates whether Zylon ship 1 will attack our starship
00928 ;          back. Compare ISBACKATTACK0 ($B8).
00929 ;
00930 ; $BA      ZYLONTIMX0
00931 ;
00932 ;          Delay timer of the x-velocity index of Zylon ship 0. Us
00933 ;          subroutine MANEUVER ($AA79). It is decremented every ga
00934 ;          iteration. When the timer value falls below 0 the curre
00935 ;          index ZYLONVELINDX0 ($B4) is adjusted depending on the
00936 ;          joystick position. The new timer value is set depending
00937 ;          resulting new x-velocity index. Used values are: 0, 2,
00938 ;
00939 ; $BB      ZYLONTIMX1
00940 ;
00941 ;          Delay timer of x-velocity index of Zylon ship 1. Compar
00942 ;          ($BA).
00943 ;
00944 ; $BC      ZYLONTIMY0
00945 ;
00946 ;          Delay timer of y-velocity index of Zylon ship 0. Compar
00947 ;          ($BA).
00948 ;
00949 ; $BD      ZYLONTIMY1
00950 ;
00951 ;          Delay timer of y-velocity index of Zylon ship 1. Compar
00952 ;          ($BA).
00953 ;
00954 ; $BE      TORPEDODELAY
00955 ;
00956 ;          After a Zylon photon torpedo has hit our starship this
00957 ;          initialized to a value of 2. It is decremented every ga
00958 ;          iteration and so delays the launch of the next Zylon ph
00959 ;          for 2 game loop iterations.
00960 ;
00961 ; $BF      ZYLONATTACKER
00962 ;
00963 ;          Index of the Zylon ship that launched the Zylon photon
00964 ;          used in GAMELOOP ($A1F3) to override the current tracki
00965 ;          settings in order to track this Zylon ship first. Used
00966 ;            $00 -> Zylon photon torpedo was launched by Zylon shi
00967 ;            $01 -> Zylon photon torpedo was launched by Zylon shi
00968 ;
00969 ; $C0      WARPSTATE
00970 ;
00971 ;          Hyperwarp state. Used values are:
00972 ;            $00 -> Hyperwarp not engaged
00973 ;            $7F -> Hyperwarp engaged
00974 ;            $FF -> In hyperspace
00975 ;
00976 ; $C1      VELOCITYHI
00977 ;
```

```
00978 ;          Our starship's velocity (high byte) in <KM/H>. Used val
00979 ;            $00 -> Not in hyperspace (regular cruise or accelerat
00980 ;                   hyperspace velocity)
00981 ;            $01 -> Hyperspace velocity
00982 ;
00983 ;          See also VELOCITYLO ($70).
00984 ;
00985 ; $C2      TRAILDELAY
00986 ;
00987 ;          Delay timer to create the next star trail. Its value is
00988 ;          from 3 to 0 every game loop iteration during the hyperw
00989 ;          PHASE in subroutine INITTRAIL ($A9B4).
00990 ;
00991 ; $C3      TRAILIND
00992 ;
00993 ;          Position vector index of the star trail's first star. U
00994 ;          subroutine INITTRAIL ($A9B4) to initialize a star trail
00995 ;          then displayed during the hyperwarp STAR TRAIL PHASE. U
00996 ;          are: 17..48 in wrap-around fashion.
00997 ;
00998 ; $C4      WARPTEMPCOLUMN
00999 ;
01000 ;          Temporary arrival column number of our starship on the
01001 ;          at the beginning of hyperspace. It is given in Player/M
01002 ;          relative to the left Galactic Chart border. Divide this
01003 ;          get the sector column number. Used values are: $00..$7F
01004 ;          WARPARRVCOLUMN ($8F).
01005 ;
01006 ; $C5      WARPTEMPROW
01007 ;
01008 ;          Temporary arrival row number of our starship on the Gal
01009 ;          the beginning of hyperspace. It is given in Player/Miss
01010 ;          relative to top Galactic Chart border. Divide this valu
01011 ;          the sector row number.  Used values are: $00..$7F. See
01012 ;          WARPARRVROW ($8E).
01013 ;
01014 ; $C6      VEERMASK
01015 ;
01016 ;          Limits the veer-off velocity of the Hyperwarp Target Ma
01017 ;          the hyperwarp ACCELERATION PHASE in subroutine HYPERWAR
01018 ;          Values are picked from table VEERMASKTAB ($BED7).
01019 ;
01020 ;          Also used as a local variable.
01021 ;
01022 ; $C7      VICINITYMASK
01023 ;
01024 ;          Mask to confine space objects' position vector componen
01025 ;          (coordinates) in a sector into a certain interval aroun
01026 ;          after its arrival from hyperspace. Values are picked fr
01027 ;          VICINITYMASKTAB ($BFB3).
01028 ;
01029 ; $C8      JOYSTICKX
01030 ;
01031 ;          Horizontal joystick direction. Values are picked from t
01032 ;          STICKINCTAB ($BAF5). Used values are:
01033 ;            $01 -> Right
01034 ;            $00 -> Centered
01035 ;            $FF -> Left
01036 ;
```

```
01037 ; $C9      JOYSTICKY
01038 ;
01039 ;          Vertical joystick direction. Values are picked from tab
01040 ;          ($BAF5). Used values are:
01041 ;            $01 -> Up
01042 ;            $00 -> Centered
01043 ;            $FF -> Down
01044 ;
01045 ; $CA      KEYCODE
01046 ;
01047 ;          Hardware keyboard code of the pressed key on the keyboa
01048 ;          Control key bits B7..6 are always set.
01049 ;
01050 ; $CB..$CC SCORE
01051 ;
01052 ;          Internal 16-bit score of the game in low byte-high byte
01053 ;
01054 ; $CD      SCOREDRANKIND
01055 ;
01056 ;          Scored Rank Index. It is translated with table RANKTAB
01057 ;          title phrase offset pointing to the rank string. Used v
01058 ;            $00 -> GALACTIC COOK
01059 ;            $01 -> GARBAGE SCOW CAPTAIN
01060 ;            $02 -> GARBAGE SCOW CAPTAIN
01061 ;            $03 -> ROOKIE
01062 ;            $04 -> ROOKIE
01063 ;            $05 -> NOVICE
01064 ;            $06 -> NOVICE
01065 ;            $07 -> ENSIGN
01066 ;            $08 -> ENSIGN
01067 ;            $09 -> PILOT
01068 ;            $0A -> PILOT
01069 ;            $0B -> ACE
01070 ;            $0C -> LIEUTENANT
01071 ;            $0D -> WARRIOR
01072 ;            $0E -> CAPTAIN
01073 ;            $0F -> COMMANDER
01074 ;            $10 -> COMMANDER
01075 ;            $11 -> STAR COMMANDER
01076 ;            $12 -> STAR COMMANDER
01077 ;
01078 ; $CE      SCOREDCLASSIND
01079 ;
01080 ;          Scored Class Index. It is translated into a class numbe
01081 ;          CLASSTAB ($BEFC). Used values are:
01082 ;            $00 -> Class 5
01083 ;            $01 -> Class 5
01084 ;            $02 -> Class 5
01085 ;            $03 -> Class 4
01086 ;            $04 -> Class 4
01087 ;            $05 -> Class 4
01088 ;            $06 -> Class 4
01089 ;            $07 -> Class 3
01090 ;            $08 -> Class 3
01091 ;            $09 -> Class 3
01092 ;            $0A -> Class 2
01093 ;            $0B -> Class 2
01094 ;            $0C -> Class 2
01095 ;            $0D -> Class 1
```

```
01096 ;           $0E -> Class 1
01097 ;           $0F -> Class 1
01098 ;
01099 ; $CF      TITLELIFE
01100 ;
01101 ;          Lifetime of title line. It is decremented every game lo
01102 ;          Used initial values are:
01103 ;            $3C -> When displaying regular title phrases
01104 ;            $FE -> When displaying "STARBASE SURROUNDED", "STARBA
01105 ;                   and "RED ALERT" messages
01106 ;            $FF -> Hide title line
01107 ;
01108 ; $D0      SHIPVIEW
01109 ;
01110 ;          Current view of our starship. Values are picked from ta
01111 ;          VIEWMODETAB ($BE22). Used values are:
01112 ;            $00 -> Front view
01113 ;            $01 -> Aft view
01114 ;            $40 -> Long-Range Scan view
01115 ;            $80 -> Galactic Chart view
01116 ;
01117 ; $D1      TITLEPHR
01118 ;
01119 ;          Title phrase offset for text phrase in title line. Used
01120 ;            $00..$7B -> Title phrase offset into PHRASETAB ($BBAA
01121 ;            $FF      -> Hide title line
01122 ;
01123 ;          See also NEWTITLEPHR ($65).
01124 ;
01125 ; $D2      BEEPFRQIND
01126 ;
01127 ;          Beeper sound pattern: Running index into frequency tabl
01128 ;          ($BF5C). See also BEEPFRQSTART ($D7). See also subrouti
01129 ;          ($B3A6) and SOUND ($B2AB).
01130 ;
01131 ; $D3      BEEPREPEAT
01132 ;
01133 ;          Beeper sound pattern: Number of times the beeper sound
01134 ;          repeated - 1. See also subroutines BEEP ($B3A6) and SOU
01135 ;
01136 ; $D4      BEEPTONELIFE
01137 ;
01138 ;          Beeper sound pattern: Lifetime of tone in TICKs - 1. Se
01139 ;          subroutines BEEP ($B3A6) and SOUND ($B2AB).
01140 ;
01141 ; $D5      BEEPPAUSELIFE
01142 ;
01143 ;          Beeper sound pattern: Lifetime of pause in TICKs - 1. U
01144 ;          are:
01145 ;            < $FF -> Number of TICKs - 1 to play
01146 ;              $FF -> Skip playing pause
01147 ;
01148 ;          See also subroutines BEEP ($B3A6) and SOUND ($B2AB).
01149 ;
01150 ; $D6      BEEPPRIORITY
01151 ;
01152 ;          Beeper sound pattern: Pattern priority. Each beeper sou
01153 ;          a priority. When a pattern of higher priority is about
01154 ;          the pattern that is currently playing is stopped. Used
```

```
01155 ;             $00 -> No pattern playing at the moment
01156 ;           > $00 -> Pattern priority
01157 ;
01158 ;         See also subroutines BEEP ($B3A6) and SOUND ($B2AB).
01159 ;
01160 ; $D7     BEEPFRQSTART
01161 ;
01162 ;         Beeper sound pattern: Index to first byte of the patter
01163 ;         table BEEPFRQTAB ($BF5C). See also BEEPFRQIND ($D2). Se
01164 ;         subroutines BEEP ($B3A6) and SOUND ($B2AB).
01165 ;
01166 ; $D8     BEEPLIFE
01167 ;
01168 ;         Beeper sound pattern: Lifetime of the current tone or p
01169 ;         It is decremented every TICK. See also subroutines BEEP
01170 ;         SOUND ($B2AB).
01171 ;
01172 ; $D9     BEEPTOGGLE
01173 ;
01174 ;         Beeper sound pattern: Indicates that either a tone or a
01175 ;         currently played. Used values are:
01176 ;           $00 -> Tone
01177 ;           $01 -> Pause
01178 ;
01179 ;         See also subroutines BEEP ($B3A6) and SOUND ($B2AB).
01180 ;
01181 ; $DA     NOISETORPTIM
01182 ;
01183 ;         Noise sound pattern: Delay timer for PHOTON TORPEDO LAU
01184 ;         sound pattern. It is decremented every TICK. See also s
01185 ;         NOISE ($AEA8) and SOUND ($B2AB).
01186 ;
01187 ; $DB     NOISEEXPLTIM
01188 ;
01189 ;         Noise sound pattern: Delay timer for SHIELD EXPLOSION a
01190 ;         EXPLOSION noise sound pattern. It is decremented every
01191 ;         subroutines NOISE ($AEA8) and SOUND ($B2AB).
01192 ;
01193 ; $DC     NOISEAUDC2
01194 ;
01195 ;         Noise sound pattern: Audio channel 1/2 control shadow r
01196 ;         also subroutines NOISE ($AEA8) and SOUND ($B2AB).
01197 ;
01198 ; $DD     NOISEAUDC3
01199 ;
01200 ;         Noise sound pattern: Audio channel 3 control shadow reg
01201 ;         also subroutines NOISE ($AEA8) and SOUND ($B2AB).
01202 ;
01203 ; $DE     NOISEAUDF1
01204 ;
01205 ;         Noise sound pattern: Audio channel 1 frequency shadow r
01206 ;         also subroutines NOISE ($AEA8) and SOUND ($B2AB).
01207 ;
01208 ; $DF     NOISEAUDF2
01209 ;
01210 ;         Noise sound pattern: Audio channel 2 frequency shadow r
01211 ;         also subroutines NOISE ($AEA8) and SOUND ($B2AB).
01212 ;
01213 ; $E0     NOISEFRQINC
```

```
01214 ;
01215 ;         Noise sound pattern: Audio channel 1/2 frequency increm
01216 ;         subroutines NOISE ($AEA8) and SOUND ($B2AB).
01217 ;
01218 ; $E1      NOISELIFE
01219 ;
01220 ;         Noise sound pattern: Noise sound pattern lifetime. It i
01221 ;         every TICK. See also subroutines NOISE ($AEA8) and SOUN
01222 ;
01223 ; $E2      NOISEZYLONTIM
01224 ;
01225 ;         Delay timer to trigger the ZYLON EXPLOSION noise sound
01226 ;         set in subroutine COLLISION ($AF3D) when an impact of o
01227 ;         starship's photon torpedoes into a target is imminent.
01228 ;         decremented every TICK during the execution of the Vert
01229 ;         Interrupt handler VBIHNDLR ($A6D1). When the timer valu
01230 ;         the ZYLON EXPLOSION noise sound pattern is played in su
01231 ;         ($B2AB).
01232 ;
01233 ; $E3      NOISEHITLIFE
01234 ;
01235 ;         Lifetime of STARSHIP EXPLOSION noise when our starship
01236 ;         by a Zylon photon torpedo. It is set in routine GAMELOO
01237 ;         value of 64 TICKs. It is decremented every TICK during
01238 ;         of the Vertical Blank Interrupt handler VBIHNDLR ($A6D1
01239 ;
01240 ; $E4      PL0SHAPOFF
01241 ;
01242 ;         PLAYER0 offset into shape table PLSHAP2TAB ($B9B1)
01243 ;
01244 ; $E5      PL1SHAPOFF
01245 ;
01246 ;         PLAYER1 offset into shape table PLSHAP2TAB ($B9B1)
01247 ;
01248 ; $E6      PL2SHAPOFF
01249 ;
01250 ;         PLAYER2 offset into shape table PLSHAP1TAB ($B8E4)
01251 ;
01252 ; $E7      PL3SHAPOFF
01253 ;
01254 ;         PLAYER3 offset into shape table PLSHAP1TAB ($B8E4)
01255 ;
01256 ; $E8      PL4SHAPOFF
01257 ;
01258 ;         PLAYER4 offset into shape table PLSHAP1TAB ($B8E4)
01259 ;
01260 ; $E9      PL0LIFE
01261 ;
01262 ;         Lifetime of the space object represented by PLAYER0 (us
01263 ;         ship 0). Any value other than $FF is decremented with e
01264 ;         iteration. Used values are:
01265 ;           $00      -> Space object not alive (= not in use)
01266 ;           $01..$FE -> Values during lifetime countdown
01267 ;           $FF      -> Infinite lifetime (not counted down)
01268 ;
01269 ; $EA      PL1LIFE
01270 ;
01271 ;         Lifetime of a space object represented by PLAYER1 (usua
01272 ;         1). Compare PL0LIFE ($E9).
```

```
01273 ;
01274 ; $EB      PL2LIFE
01275 ;
01276 ;         Lifetime of a space object represented by PLAYER2 (usua
01277 ;         photon torpedo). Compare PL0LIFE ($E9).
01278 ;
01279 ;         If this PLAYER represents a photon torpedo, its lifetim
01280 ;         decremented from an initial value of $FF.
01281 ;
01282 ; $EC      PL3LIFE
01283 ;
01284 ;         Lifetime of a space object represented by PLAYER3 (usua
01285 ;         starship's photon torpedo 0). Compare PL2LIFE ($EB).
01286 ;
01287 ;         If this PLAYER represents a photon torpedo, its lifetim
01288 ;         decremented from an initial value of $FF.
01289 ;
01290 ; $ED      PL4LIFE
01291 ;
01292 ;         Lifetime of a space object represented by PLAYER4 (usua
01293 ;         starship's photon torpedo 1). Compare PL2LIFE ($EB).
01294 ;
01295 ;         If this PLAYER represents a photon torpedo, its lifetim
01296 ;         decremented from an initial value of $FF.
01297 ;
01298 ; $EE      PL0COLOR
01299 ;
01300 ;         Color of PLAYER0
01301 ;
01302 ; $EF      PL1COLOR
01303 ;
01304 ;         Color of PLAYER1
01305 ;
01306 ; $F0      PL2COLOR
01307 ;
01308 ;         Color of PLAYER2
01309 ;
01310 ; $F1      PL3COLOR
01311 ;
01312 ;         Color of PLAYER3
01313 ;
01314 ; $F2      PF0COLOR
01315 ;
01316 ;         Color of PLAYFIELD0
01317 ;
01318 ; $F3      PF1COLOR
01319 ;
01320 ;         Color of PLAYFIELD1
01321 ;
01322 ; $F4      PF2COLOR
01323 ;
01324 ;         Color of PLAYFIELD2
01325 ;
01326 ; $F5      PF3COLOR
01327 ;
01328 ;         Color of PLAYFIELD3
01329 ;
01330 ; $F6      BGRCOLOR
01331 ;
```

```
01332 ;          Color of BACKGROUND
01333 ;
01334 ; $F7      PF0COLORDLI
01335 ;
01336 ;          Color of PLAYFIELD0 after DLI
01337 ;
01338 ; $F8      PF1COLORDLI
01339 ;
01340 ;          Color of PLAYFIELD1 after DLI
01341 ;
01342 ; $F9      PF2COLORDLI
01343 ;
01344 ;          Color of PLAYFIELD2 after DLI
01345 ;
01346 ; $FA      PF3COLORDLI
01347 ;
01348 ;          Color of PLAYFIELD3 after DLI
01349 ;
01350 ; $FB      BGRCOLORDLI
01351 ;
01352 ;          Color of BACKGROUND after DLI
01353 ;
01354 ; $0280..$02E9 DSPLST
01355 ;
01356 ;              Display List
01357 ;
01358 ; $0300..$03FF PL4DATA
01359 ;
01360 ;              PLAYER4 data area
01361 ;
01362 ; $0400..$04FF PL0DATA
01363 ;
01364 ;              PLAYER0 data area
01365 ;
01366 ; $0500..$05FF PL1DATA
01367 ;
01368 ;              PLAYER1 data area
01369 ;
01370 ; $0600..$06FF PL2DATA
01371 ;
01372 ;              PLAYER2 data area
01373 ;
01374 ; $0700..$07FF PL3DATA
01375 ;
01376 ;              PLAYER3 data area
01377 ;
01378 ; $0800..$0863 PFMEMROWLO
01379 ;
01380 ;              Lookup table of start addresses (low byte) for each
01381 ;              PLAYFIELD memory, which is located at PFMEM ($1000)
01382 ;              contains 100 bytes for 100 rows (of which only 99 a
01383 ;              the Display List, the PLAYFIELD is 160 x 99 pixels)
01384 ;              addresses grow in increments of 40 (40 bytes = 160
01385 ;              GRAPHICS7 mode = 1 PLAYFIELD row of pixels). See al
01386 ;              ($0864).
01387 ;
01388 ; $0864..$08C7 PFMEMROWHI
01389 ;
01390 ;              Lookup table of start addresses (high byte) of each
```

```
01391 ;              PLAYFIELD memory. See also PFMEMROWLO ($0800).
01392 ;
01393 ; $08C9..$0948 GCMEMMAP
01394 ;
01395 ;              Galactic Chart memory map (16 columns x 8 rows = 12
01396 ;
01397 ; $0949..$0970 PANELTXT
01398 ;
01399 ;              Memory of Control Panel Display (bottom text window
01400 ;              view, Aft view, and Long-Range Scan view (20 charac
01401 ;              = 40 bytes).
01402 ;
01403 ; $094A        VELOCD1
01404 ;
01405 ;              First digit (of 2) of the VELOCITY readout in Contr
01406 ;              Display memory.
01407 ;
01408 ; $0950        KILLCNTD1
01409 ;
01410 ;              First digit (of 2) of the KILL COUNTER readout in C
01411 ;              Display memory.
01412 ;
01413 ; $0955        ENERGYD1
01414 ;
01415 ;              First digit (of 4) of the ENERGY readout in Control
01416 ;              memory.
01417 ;
01418 ; $095A        TRACKC1
01419 ;
01420 ;              Character of the TRACKING readout 'T' or 'C' in Con
01421 ;              Display memory.
01422 ;
01423 ; $095C        TRACKDIGIT
01424 ;
01425 ;              Digit of the TRACKING readout in Control Panel Disp
01426 ;              is used to store the index of the currently tracked
01427 ;              Used values are:
01428 ;                $00 -> Track Zylon ship 0
01429 ;                $01 -> Track Zylon ship 1
01430 ;                $02 -> Track starbase
01431 ;                $03 -> Track Hyperwarp Target Marker
01432 ;
01433 ; $0960        THETAC1
01434 ;
01435 ;              First character of the THETA readout in Control Pan
01436 ;              memory.
01437 ;
01438 ; $0966        PHIC1
01439 ;
01440 ;              First character of the PHI readout in Control Panel
01441 ;              memory.
01442 ;
01443 ; $096C        RANGEC1
01444 ;
01445 ;              First character of the RANGE readout in Control Pan
01446 ;              memory.
01447 ;
01448 ; $0971..$09AC GCTXT
01449 ;
```

```
01450 ;             Memory of Galactic Chart Panel Display (bottom text
01451 ;             Galactic Chart view (20 characters x 3 rows = 60 by
01452 ;
01453 ; $097D       GCWARPD1
01454 ;
01455 ;             First digit (of 4) of the HYPERWARP ENERGY readout
01456 ;             Chart Panel Display memory.
01457 ;
01458 ; $098D       GCTRGCNT
01459 ;
01460 ;             First target counter digit (of 2) in Galactic Chart
01461 ;             memory.
01462 ;
01463 ; $0992       GCSTATPHO
01464 ;
01465 ;             Photon Torpedo status letter in Galactic Chart Pane
01466 ;             memory. Used values are:
01467 ;               %00****** -> OK
01468 ;               %10****** -> Destroyed
01469 ;               %11****** -> Damaged
01470 ;
01471 ; $0993       GCSTATENG
01472 ;
01473 ;             Engines status letter in Galactic Chart Panel Displ
01474 ;             Used values are:
01475 ;               %00****** -> OK
01476 ;               %10****** -> Destroyed
01477 ;               %11****** -> Damaged
01478 ;
01479 ; $0994       GCSTATSHL
01480 ;
01481 ;             Shields status letter in Galactic Chart Panel Displ
01482 ;             Used values are:
01483 ;               %00****** -> OK
01484 ;               %10****** -> Destroyed
01485 ;               %11****** -> Damaged
01486 ;
01487 ; $0995       GCSTATCOM
01488 ;
01489 ;             Attack Computer status letter in Galactic Chart Pan
01490 ;             memory. Used values are:
01491 ;               %00****** -> OK
01492 ;               %10****** -> Destroyed
01493 ;               %11****** -> Damaged
01494 ;
01495 ; $0996       GCSTATLRS
01496 ;
01497 ;             Long-Range Scan status letter in Galactic Chart Pan
01498 ;             memory. Used values are:
01499 ;               %00****** -> OK
01500 ;               %10****** -> Destroyed
01501 ;               %11****** -> Damaged
01502 ;
01503 ; $0997       GCSTATRAD
01504 ;
01505 ;             Subspace Radio status letter in Galactic Chart Pane
01506 ;             memory. Used values are:
01507 ;               %00****** -> OK
01508 ;               %10****** -> Destroyed
```

```
01509 ;                 %11****** -> Damaged
01510 ;
01511 ; $09A3         GCSTARDAT
01512 ;
01513 ;               First (of 5) digits of the star date clock in the G
01514 ;               Panel Display memory.
01515 ;
01516 ; $09AD..$09DD ZPOSSIGN
01517 ;
01518 ;               Table containing the sign bit (B16) of position vec
01519 ;               z-components (z-coordinate) (49 bytes). Bytes 0..4
01520 ;               position vectors of PLAYER space objects (Zylon shi
01521 ;               torpedoes, etc.). Bytes 5..48 belong to position ve
01522 ;               PLAYFIELD space objects (stars, explosion fragments
01523 ;               are:
01524 ;                 $00 -> Negative sign (behind our starship)
01525 ;                 $01 -> Positive sign (in front of our starship)
01526 ;
01527 ;               See also "ON POSITION VECTORS".
01528 ;
01529 ; $09AD         PL0ZPOSSIGN
01530 ;
01531 ;               Sign bit (B16) of position vector z-component (z-co
01532 ;               PLAYER0. Compare ZPOSSIGN ($09AD). See also "ON POS
01533 ;               VECTORS".
01534 ;
01535 ; $09AE         PL1ZPOSSIGN
01536 ;
01537 ;               Sign bit (B16) of position vector z-component (z-co
01538 ;               PLAYER1. Compare ZPOSSIGN ($09AD). See also "ON POS
01539 ;               VECTORS".
01540 ;
01541 ; $09AF         PL2ZPOSSIGN
01542 ;
01543 ;               Sign bit (B16) of position vector z-component (z-co
01544 ;               PLAYER2. Compare ZPOSSIGN ($09AD). See also "ON POS
01545 ;               VECTORS".
01546 ;
01547 ; $09B0         PL3ZPOSSIGN
01548 ;
01549 ;               Sign bit (B16) of position vector z-component (z-co
01550 ;               PLAYER3. Compare ZPOSSIGN ($09AD). See also "ON POS
01551 ;               VECTORS".
01552 ;
01553 ; $09B1         PL4ZPOSSIGN
01554 ;
01555 ;               Sign bit (B16) of position vector z-component (z-co
01556 ;               PLAYER4. Compare ZPOSSIGN ($09AD). See also "ON POS
01557 ;               VECTORS".
01558 ;
01559 ; $09DE..$0A0E XPOSSIGN
01560 ;
01561 ;               Table containing the sign bit (B16) of position vec
01562 ;               x-components (x-coordinate) (49 bytes). Bytes 0..4
01563 ;               position vectors of PLAYER space objects (Zylon shi
01564 ;               torpedoes, etc.). Bytes 5..48 belong to position ve
01565 ;               PLAYFIELD space objects (stars, explosion fragments
01566 ;               are:
01567 ;                 $00 -> Negative sign (left)
```

```
01568 ;                 $01 -> Positive sign (right)
01569 ;
01570 ;               See also "ON POSITION VECTORS".
01571 ;
01572 ; $09DE         PL0XPOSSIGN
01573 ;
01574 ;               Sign bit (B16) of position vector x-component (x-co
01575 ;               PLAYER0. Compare XPOSSIGN ($09DE). See also "ON POS
01576 ;               VECTORS".
01577 ;
01578 ; $09DF         PL1XPOSSIGN
01579 ;
01580 ;               Sign bit (B16) of position vector x-component (x-co
01581 ;               PLAYER1. Compare XPOSSIGN ($09DE). See also "ON POS
01582 ;               VECTORS".
01583 ;
01584 ; $09E0         PL2XPOSSIGN
01585 ;
01586 ;               Sign bit (B16) of position vector x-component (x-co
01587 ;               PLAYER2. Compare XPOSSIGN ($09DE). See also "ON POS
01588 ;               VECTORS".
01589 ;
01590 ; $09E1         PL3XPOSSIGN
01591 ;
01592 ;               Sign bit (B16) of position vector x-component (x-co
01593 ;               PLAYER3. Compare XPOSSIGN ($09DE). See also "ON POS
01594 ;               VECTORS".
01595 ;
01596 ; $09E2         PL4XPOSSIGN
01597 ;
01598 ;               Sign bit (B16) of position vector x-component (x-co
01599 ;               PLAYER4. Compare XPOSSIGN ($09DE). See also "ON POS
01600 ;               VECTORS".
01601 ;
01602 ; $0A0F..$0A3F YPOSSIGN
01603 ;
01604 ;               Table containing the sign bit (B16) of position vec
01605 ;               y-components (y-coordinate) (49 bytes). Bytes 0..4
01606 ;               position vectors of PLAYER space objects (Zylon shi
01607 ;               torpedoes, etc.). Bytes 5..48 belong to position ve
01608 ;               PLAYFIELD space objects (stars, explosion fragments
01609 ;               are:
01610 ;                 $00 -> Negative sign (down)
01611 ;                 $01 -> Positive sign (up)
01612 ;
01613 ;               See also "ON POSITION VECTORS".
01614 ;
01615 ; $0A0F         PL0YPOSSIGN
01616 ;
01617 ;               Sign bit (B16) of position vector y-component (y-co
01618 ;               PLAYER0. Compare YPOSSIGN ($0A0F). See also "ON POS
01619 ;               VECTORS".
01620 ;
01621 ; $0A10         PL1YPOSSIGN
01622 ;
01623 ;               Sign bit (B16) of position vector y-component (y-co
01624 ;               PLAYER1. Compare YPOSSIGN ($0A0F). See also "ON POS
01625 ;               VECTORS".
01626 ;
```

```
01627 ; $0A11        PL2YPOSSIGN
01628 ;
01629 ;              Sign bit (B16) of position vector y-component (y-co
01630 ;              PLAYER2. Compare YPOSSIGN ($0A0F). See also "ON POS
01631 ;              VECTORS".
01632 ;
01633 ; $0A12        PL3YPOSSIGN
01634 ;
01635 ;              Sign bit (B16) of position vector y-component (y-co
01636 ;              PLAYER3. Compare YPOSSIGN ($0A0F). See also "ON POS
01637 ;              VECTORS".
01638 ;
01639 ; $0A13        PL4YPOSSIGN
01640 ;
01641 ;              Sign bit (B16) of position vector y-component (y-co
01642 ;              PLAYER4. Compare YPOSSIGN ($0A0F). See also "ON POS
01643 ;              VECTORS".
01644 ;
01645 ; $0A40..$0A70 ZPOSHI
01646 ;
01647 ;              Table containing the high byte (B15..8) of position
01648 ;              y-components (y-coordinate) (49 bytes). Bytes 0..4
01649 ;              position vectors of PLAYER space objects (Zylon shi
01650 ;              torpedoes, etc.). Bytes 5..48 belong to position ve
01651 ;              PLAYFIELD space objects (stars, explosion fragments
01652 ;              "ON POSITION VECTORS".
01653 ;
01654 ; $0A40        PL0ZPOSHI
01655 ;
01656 ;              High byte (B15..8) of position vector z-component (
01657 ;              of PLAYER0. Compare ZPOSHI ($0A40). See also "ON PO
01658 ;              VECTORS".
01659 ;
01660 ; $0A41        PL1ZPOSHI
01661 ;
01662 ;              High byte (B15..8) of position vector z-component (
01663 ;              of PLAYER1. Compare ZPOSHI ($0A40). See also "ON PO
01664 ;              VECTORS".
01665 ;
01666 ; $0A42        PL2ZPOSHI
01667 ;
01668 ;              High byte (B15..8) of position vector z-component (
01669 ;              of PLAYER2. Compare ZPOSHI ($0A40). See also "ON PO
01670 ;              VECTORS".
01671 ;
01672 ; $0A43        PL3ZPOSHI
01673 ;
01674 ;              High byte (B15..8) of position vector z-component (
01675 ;              of PLAYER3. Compare ZPOSHI ($0A40). See also "ON PO
01676 ;              VECTORS".
01677 ;
01678 ; $0A44        PL4ZPOSHI
01679 ;
01680 ;              High byte (B15..8) of position vector z-component (
01681 ;              of PLAYER4. Compare ZPOSHI ($0A40). See also "ON PO
01682 ;              VECTORS".
01683 ;
01684 ; $0A71..$0AA1 XPOSHI
01685 ;
```

```
01686 ;             Table containing the high byte (B15..8) of position
01687 ;             x-components (x-coordinate) (49 bytes). Bytes 0..4
01688 ;             position vectors of PLAYER space objects (Zylon shi
01689 ;             torpedoes, etc.). Bytes 5..48 belong to position ve
01690 ;             PLAYFIELD space objects (stars, explosion fragments
01691 ;             "ON POSITION VECTORS".
01692 ;
01693 ; $0A71       PL0XPOSHI
01694 ;
01695 ;             High byte (B15..8) of position vector x-component (
01696 ;             of PLAYER0. Compare XPOSHI ($0A71). See also "ON PO
01697 ;             VECTORS".
01698 ;
01699 ; $0A72       PL1XPOSHI
01700 ;
01701 ;             High byte (B15..8) of position vector x-component (
01702 ;             of PLAYER1. Compare XPOSHI ($0A71). See also "ON PO
01703 ;             VECTORS".
01704 ;
01705 ; $0A73       PL2XPOSHI
01706 ;
01707 ;             High byte (B15..8) of position vector x-component (
01708 ;             of PLAYER2. Compare XPOSHI ($0A71). See also "ON PO
01709 ;             VECTORS".
01710 ;
01711 ; $0A74       PL3XPOSHI
01712 ;
01713 ;             High byte (B15..8) of position vector x-component (
01714 ;             of PLAYER3. Compare XPOSHI ($0A71). See also "ON PO
01715 ;             VECTORS".
01716 ;
01717 ; $0A75       PL4XPOSHI
01718 ;
01719 ;             High byte (B15..8) of position vector x-component (
01720 ;             of PLAYER4. Compare XPOSHI ($0A71). See also "ON PO
01721 ;             VECTORS".
01722 ;
01723 ; $0AA2..$0AD2 YPOSHI
01724 ;
01725 ;             Table containing the high byte (B15..8) of position
01726 ;             y-components (y-coordinate) (49 bytes). Bytes 0..4
01727 ;             position vectors of PLAYER space objects (Zylon shi
01728 ;             torpedoes, etc.). Bytes 5..48 belong to position ve
01729 ;             PLAYFIELD space objects (stars, explosion fragments
01730 ;             "ON POSITION VECTORS".
01731 ;
01732 ; $0AA2       PL0YPOSHI
01733 ;
01734 ;             High byte (B15..8) of position vector y-component (
01735 ;             of PLAYER0. Compare YPOSHI ($0AA2). See also "ON PO
01736 ;             VECTORS".
01737 ;
01738 ; $0AA3       PL1YPOSHI
01739 ;
01740 ;             High byte (B15..8) of position vector y-component (
01741 ;             of PLAYER1. Compare YPOSHI ($0AA2). See also "ON PO
01742 ;             VECTORS".
01743 ;
01744 ; $0AA4       PL2YPOSHI
```

```
01745 ;
01746 ;               High byte (B15..8) of position vector y-component (
01747 ;               of PLAYER2. Compare YPOSHI ($0AA2). See also "ON PO
01748 ;               VECTORS".
01749 ;
01750 ; $0AA5         PL3YPOSHI
01751 ;
01752 ;               High byte (B15..8) of position vector y-component (
01753 ;               of PLAYER3. Compare YPOSHI ($0AA2). See also "ON PO
01754 ;               VECTORS".
01755 ;
01756 ; $0AA6         PL4YPOSHI
01757 ;
01758 ;               High byte (B15..8) of position vector y-component (
01759 ;               of PLAYER4. Compare YPOSHI ($0AA2). See also "ON PO
01760 ;               VECTORS".
01761 ;
01762 ; $0AD3..$0B03 ZPOSLO
01763 ;
01764 ;               Table containing the low byte (B7..0) of position v
01765 ;               z-components (z-coordinate) (49 bytes). Bytes 0..4
01766 ;               position vectors of PLAYER space objects (Zylon shi
01767 ;               torpedoes, etc.). Bytes 5..48 belong to position ve
01768 ;               PLAYFIELD space objects (stars, explosion fragments
01769 ;               "ON POSITION VECTORS".
01770 ;
01771 ; $0AD3         PL0ZPOSLO
01772 ;
01773 ;               Low byte (B7..0) of position vector z-component (z-
01774 ;               PLAYER0. Compare ZPOSLO ($0AD3). See also "ON POSIT
01775 ;
01776 ; $0AD4         PL1ZPOSLO
01777 ;
01778 ;               Low byte (B7..0) of position vector z-component (z-
01779 ;               PLAYER1. Compare ZPOSLO ($0AD3). See also "ON POSIT
01780 ;
01781 ; $0AD5         PL2ZPOSLO
01782 ;
01783 ;               Low byte (B7..0) of position vector z-component (z-
01784 ;               PLAYER2. Compare ZPOSLO ($0AD3). See also "ON POSIT
01785 ;
01786 ; $0AD6         PL3ZPOSLO
01787 ;
01788 ;               Low byte (B7..0) of position vector z-component (z-
01789 ;               PLAYER3. Compare ZPOSLO ($0AD3). See also "ON POSIT
01790 ;
01791 ; $0AD7         PL4ZPOSLO
01792 ;
01793 ;               Low byte (B7..0) of position vector z-component (z-
01794 ;               PLAYER4. Compare ZPOSLO ($0AD3). See also "ON POSIT
01795 ;
01796 ; $0B04..$0B34 XPOSLO
01797 ;
01798 ;               Table containing the low byte (B7..0) of position v
01799 ;               x-components (x-coordinate) (49 bytes). Bytes 0..4
01800 ;               position vectors of PLAYER space objects (Zylon shi
01801 ;               torpedoes, etc.). Bytes 5..48 belong to position ve
01802 ;               PLAYFIELD space objects (stars, explosion fragments
01803 ;               "ON POSITION VECTORS".
```

```
01804 ;
01805 ; $0B04        PL0XPOSLO
01806 ;
01807 ;             Low byte (B7..0) of position vector x-component (x-
01808 ;             PLAYER0. Compare XPOSLO ($0B04). See also "ON POSIT
01809 ;
01810 ; $0B05        PL1XPOSLO
01811 ;
01812 ;             Low byte (B7..0) of position vector x-component (x-
01813 ;             PLAYER1. Compare XPOSLO ($0B04). See also "ON POSIT
01814 ;
01815 ; $0B06        PL2XPOSLO
01816 ;
01817 ;             Low byte (B7..0) of position vector x-component (x-
01818 ;             PLAYER2. Compare XPOSLO ($0B04). See also "ON POSIT
01819 ;
01820 ; $0B07        PL3XPOSLO
01821 ;
01822 ;             Low byte (B7..0) of position vector x-component (x-
01823 ;             PLAYER3. Compare XPOSLO ($0B04). See also "ON POSIT
01824 ;
01825 ; $0B08        PL4XPOSLO
01826 ;
01827 ;             Low byte (B7..0) of position vector x-component (x-
01828 ;             PLAYER4. Compare XPOSLO ($0B04). See also "ON POSIT
01829 ;
01830 ; $0B35..$0B65 YPOSLO
01831 ;
01832 ;             Table containing the low byte (B7..0) of position v
01833 ;             y-components (y-coordinate) (49 bytes). Bytes 0..4
01834 ;             position vectors of PLAYER space objects (Zylon shi
01835 ;             torpedoes, etc.). Bytes 5..48 belong to position ve
01836 ;             PLAYFIELD space objects (stars, explosion fragments
01837 ;             "ON POSITION VECTORS".
01838 ;
01839 ; $0B35        PL0YPOSLO
01840 ;
01841 ;             Low byte (B7..0) of position vector y-component (y-
01842 ;             PLAYER0. Compare YPOSLO ($0B35). See also "ON POSIT
01843 ;
01844 ; $0B36        PL1YPOSLO
01845 ;
01846 ;             Low byte (B7..0) of position vector y-component (y-
01847 ;             PLAYER1. Compare YPOSLO ($0B35). See also "ON POSIT
01848 ;
01849 ; $0B37        PL2YPOSLO
01850 ;
01851 ;             Low byte (B7..0) of position vector y-component (y-
01852 ;             PLAYER2. Compare YPOSLO ($0B35). See also "ON POSIT
01853 ;
01854 ; $0B38        PL3YPOSLO
01855 ;
01856 ;             Low byte (B7..0) of position vector y-component (y-
01857 ;             PLAYER3. Compare YPOSLO ($0B35). See also "ON POSIT
01858 ;
01859 ; $0B39        PL4YPOSLO
01860 ;
01861 ;             Low byte (B7..0) of position vector y-component (y-
01862 ;             PLAYER4. Compare YPOSLO ($0B35). See also "ON POSIT
```

```
01863 ;
01864 ; $0B66..$0B96 ZVEL
01865 ;
01866 ;              Table containing velocity vector z-components (z-ve
01867 ;              bytes). Bytes 0..4 belong to velocity vectors of PL
01868 ;              objects (Zylon ships, photon torpedoes, etc.). Byte
01869 ;              to velocity vectors of PLAYFIELD space objects (sta
01870 ;              fragments). Each z-velocity is stored in the binary
01871 ;              %sxxxxxxx where
01872 ;                %s = 0   -> Positive sign (moving in flight direc
01873 ;                %s = 1   -> Negative sign (moving in reverse flig
01874 ;                %xxxxxxx -> Unsigned 7-bit velocity value in <KM/
01875 ;
01876 ;              See also "ON VELOCITY VECTORS".
01877 ;
01878 ; $0B66        PL0ZVEL
01879 ;
01880 ;              Velocity vector z-component (z-velocity) of PLAYER0
01881 ;              ($0B66). See also "ON VELOCITY VECTORS".
01882 ;
01883 ; $0B67        PL1ZVEL
01884 ;
01885 ;              Velocity vector z-component (z-velocity) of PLAYER1
01886 ;              ($0B66). See also "ON VELOCITY VECTORS".
01887 ;
01888 ; $0B68        PL2ZVEL
01889 ;
01890 ;              Velocity vector z-component (z-velocity) of PLAYER2
01891 ;              ($0B66). See also "ON VELOCITY VECTORS".
01892 ;
01893 ; $0B69        PL3ZVEL
01894 ;
01895 ;              Velocity vector z-component (z-velocity) of PLAYER3
01896 ;              ($0B66). See also "ON VELOCITY VECTORS".
01897 ;
01898 ; $0B6A        PL4ZVEL
01899 ;
01900 ;              Velocity vector z-component (z-velocity) of PLAYER4
01901 ;              ($0B66). See also "ON VELOCITY VECTORS".
01902 ;
01903 ; $0B97..$0BC7 XVEL
01904 ;
01905 ;              Table containing velocity vector x-components (x-ve
01906 ;              bytes). Bytes 0..4 belong to velocity vectors of PL
01907 ;              objects (Zylon ships, photon torpedoes, etc.). Byte
01908 ;              to velocity vectors of PLAYFIELD space objects (sta
01909 ;              fragments). Each x-velocity is stored in the binary
01910 ;              %sxxxxxxx where
01911 ;                %s = 0   -> Positive sign (moving to the right)
01912 ;                %s = 1   -> Negative sign (moving to the left)
01913 ;                %xxxxxxx -> Unsigned 7-bit velocity value in <KM/
01914 ;
01915 ;              See also "ON VELOCITY VECTORS".
01916 ;
01917 ; $0B97        PL0XVEL
01918 ;
01919 ;              Velocity vector x-component (x-velocity) of PLAYER0
01920 ;              ($0B97). See also "ON VELOCITY VECTORS".
01921 ;
```

```
01922 ; $0B98        PL1XVEL
01923 ;
01924 ;             Velocity vector x-component (x-velocity) of PLAYER1
01925 ;             ($0B97). See also "ON VELOCITY VECTORS".
01926 ;
01927 ; $0B99        PL2XVEL
01928 ;
01929 ;             Velocity vector x-component (x-velocity) of PLAYER2
01930 ;             ($0B97). See also "ON VELOCITY VECTORS".
01931 ;
01932 ; $0B9A        PL3XVEL
01933 ;
01934 ;             Velocity vector x-component (x-velocity) of PLAYER3
01935 ;             ($0B97). See also "ON VELOCITY VECTORS".
01936 ;
01937 ; $0B9B        PL4XVEL
01938 ;
01939 ;             Velocity vector x-component (x-velocity) of PLAYER4
01940 ;             ($0B97). See also "ON VELOCITY VECTORS".
01941 ;
01942 ; $0BC8..$0BF8 YVEL
01943 ;
01944 ;             Table containing velocity vector y-components (y-ve
01945 ;             bytes). Bytes 0..4 belong to velocity vectors of PL
01946 ;             objects (Zylon ships, photon torpedoes, etc.). Byte
01947 ;             to velocity vectors of PLAYFIELD space objects (sta
01948 ;             fragments). Each y-velocity is stored in the binary
01949 ;             %sxxxxxxx where
01950 ;               %s = 0   -> Positive sign (moving up)
01951 ;               %s = 1   -> Negative sign (moving down)
01952 ;               %xxxxxxx -> Unsigned 7-bit velocity value in <KM/
01953 ;
01954 ;             See also "ON VELOCITY VECTORS".
01955 ;
01956 ; $0BC8        PL0YVEL
01957 ;
01958 ;             Velocity vector y-component (y-velocity) of PLAYER0
01959 ;             ($0BC8). See also "ON VELOCITY VECTORS".
01960 ;
01961 ; $0BC9        PL1YVEL
01962 ;
01963 ;             Velocity vector y-component (y-velocity) of PLAYER1
01964 ;             ($0BC8). See also "ON VELOCITY VECTORS".
01965 ;
01966 ; $0BCA        PL2YVEL
01967 ;
01968 ;             Velocity vector y-component (y-velocity) of PLAYER2
01969 ;             ($0BC8). See also "ON VELOCITY VECTORS".
01970 ;
01971 ; $0BCB        PL3YVEL
01972 ;
01973 ;             Velocity vector y-component (y-velocity) of PLAYER3
01974 ;             ($0BC8). See also "ON VELOCITY VECTORS".
01975 ;
01976 ; $0BCC        PL4YVEL
01977 ;
01978 ;             Velocity vector y-component (y-velocity) of PLAYER4
01979 ;             ($0BC8). See also "ON VELOCITY VECTORS".
01980 ;
```

```
01981 ; $0BF9..$0C29 PIXELROWNEW
01982 ;
01983 ;              Table containing the new pixel row number of space
01984 ;              bytes). Bytes 0..4 belong to PLAYER space objects a
01985 ;              Player/Missile (PM) pixel row numbers. They are cou
01986 ;              vertical PM position 0, which is offscreen. Bytes 5
01987 ;              PLAYFIELD space objects (stars, explosion fragments
01988 ;              PLAYFIELD pixel row numbers. They are counted from
01989 ;              of the PLAYFIELD and have values of 0..99. See also
01990 ;              ($0C5B).
01991 ;
01992 ; $0BF9        PL0ROWNEW
01993 ;
01994 ;              New pixel row number of PLAYER0 in Player/Missile p
01995 ;              also PIXELROWNEW ($0BF9).
01996 ;
01997 ; $0BFA        PL1ROWNEW
01998 ;
01999 ;              New pixel row number of PLAYER1 in Player/Missile p
02000 ;              also PIXELROWNEW ($0BF9).
02001 ;
02002 ; $0BFB        PL2ROWNEW
02003 ;
02004 ;              New pixel row number of PLAYER2 in Player/Missile p
02005 ;              also PIXELROWNEW ($0BF9).
02006 ;
02007 ; $0BFC        PL3ROWNEW
02008 ;
02009 ;              New pixel row number of PLAYER3 in Player/Missile p
02010 ;              also PIXELROWNEW ($0BF9).
02011 ;
02012 ; $0BFD        PL4ROWNEW
02013 ;
02014 ;              New pixel row number of PLAYER4 in Player/Missile p
02015 ;              also PIXELROWNEW ($0BF9).
02016 ;
02017 ; $0C2A..$0C5A PIXELCOLUMN
02018 ;
02019 ;              Table containing the pixel column number of space o
02020 ;              bytes). Bytes 0..4 belong to PLAYER space objects a
02021 ;              Player/Missile (PM) pixel column numbers. They are
02022 ;              horizontal PM position 0, which is offscreen. Bytes
02023 ;              to PLAYFIELD space objects (stars, explosion fragme
02024 ;              contain PLAYFIELD pixel column numbers. They are co
02025 ;              left border of the PLAYFIELD and have values of 0..
02026 ;
02027 ; $0C2A        PL0COLUMN
02028 ;
02029 ;              Pixel column number of PLAYER0 in Player/Missile pi
02030 ;              PIXELCOLUMN ($0C2A).
02031 ;
02032 ; $0C2B        PL1COLUMN
02033 ;
02034 ;              Pixel column number of PLAYER1 in Player/Missile pi
02035 ;              PIXELCOLUMN ($0C2A).
02036 ;
02037 ; $0C2C        PL2COLUMN
02038 ;
02039 ;              Pixel column number of PLAYER2 in Player/Missile pi
```

```
02040 ;              PIXELCOLUMN ($0C2A).
02041 ;
02042 ; $0C2D        PL3COLUMN
02043 ;
02044 ;              Pixel column number of PLAYER3 in Player/Missile pi
02045 ;              PIXELCOLUMN ($0C2A).
02046 ;
02047 ; $0C2E        PL4COLUMN
02048 ;
02049 ;              Pixel column number of PLAYER4 in Player/Missile pi
02050 ;              PIXELCOLUMN ($0C2A).
02051 ;
02052 ; $0C5B..$0C8B PIXELROW
02053 ;
02054 ;              Table containing the pixel row number of space obje
02055 ;              bytes). Bytes 0..4 belong to PLAYER space objects a
02056 ;              Player/Missile (PM) pixel row numbers. They are cou
02057 ;              vertical PM position 0, which is offscreen. Bytes 5
02058 ;              PLAYFIELD space objects (stars, explosion fragments
02059 ;              PLAYFIELD pixel row numbers. They are counted from
02060 ;              of the PLAYFIELD and have values of 0..99. See also
02061 ;              ($0BF9).
02062 ;
02063 ; $0C5B        PL0ROW
02064 ;
02065 ;              Pixel row number of PLAYER0 in Player/Missile pixel
02066 ;              PIXELROW ($0C5B).
02067 ;
02068 ; $0C5C        PL1ROW
02069 ;
02070 ;              Pixel row number of PLAYER1 in Player/Missile pixel
02071 ;              PIXELROW ($0C5B).
02072 ;
02073 ; $0C5D        PL2ROW
02074 ;
02075 ;              Pixel row number of PLAYER2 in Player/Missile pixel
02076 ;              PIXELROW ($0C5B).
02077 ;
02078 ; $0C5E        PL3ROW
02079 ;
02080 ;              Pixel row number of PLAYER3 in Player/Missile pixel
02081 ;              PIXELROW ($0C5B).
02082 ;
02083 ; $0C5F        PL4ROW
02084 ;
02085 ;              Pixel row number of PLAYER4 in Player/Missile pixel
02086 ;              PIXELROW ($0C5B).
02087 ;
02088 ; $0C8C..$0CBC PIXELBYTEOFF
02089 ;
02090 ;              Table containing a byte offset into PLAYFIELD memor
02091 ;              PLAYFIELD space object (stars, explosion fragments)
02092 ;              the number of bytes from the start of the PLAYFIELD
02093 ;              byte containing the space object pixel in the same
02094 ;              In other words, the pixel column modulo 4 (1 byte =
02095 ;              pixels).
02096 ;
02097 ;              NOTE: Only bytes 5..48 are used for PLAYFIELD space
02098 ;              this way. Bytes 0..4 are used differently. See PL0S
```

```
02099 ;              ($0C8C)..PL4SHAPTYPE ($0C90).
02100 ;
02101 ; $0C8C        PL0SHAPTYPE
02102 ;
02103 ;              Shape type of PLAYER0. Used to index the PLAYER's s
02104 ;              cells in tables PLSHAPOFFTAB ($BE2F) and PLSHAPHEIG
02105 ;              Used values are:
02106 ;                $00 -> PHOTON TORPEDO
02107 ;                $10 -> ZYLON FIGHTER
02108 ;                $20 -> STARBASE RIGHT
02109 ;                $30 -> STARBASE CENTER
02110 ;                $40 -> STARBASE LEFT
02111 ;                $50 -> TRANSFER VESSEL
02112 ;                $60 -> METEOR
02113 ;                $70 -> ZYLON CRUISER
02114 ;                $80 -> ZYLON BASESTAR
02115 ;                $90 -> HYPERWARP TARGET MARKER
02116 ;
02117 ; $0C8D        PL1SHAPTYPE
02118 ;
02119 ;              Shape type of PLAYER1. Compare PL0SHAPTYPE ($0C8C).
02120 ;
02121 ; $0C8E        PL2SHAPTYPE
02122 ;
02123 ;              Shape type of PLAYER2. Compare PL0SHAPTYPE ($0C8C).
02124 ;
02125 ; $0C8F        PL3SHAPTYPE
02126 ;
02127 ;              Shape type of PLAYER3. Compare PL0SHAPTYPE ($0C8C).
02128 ;
02129 ; $0C90        PL4SHAPTYPE
02130 ;
02131 ;              Shape type of PLAYER4. Compare PL0SHAPTYPE ($0C8C).
02132 ;
02133 ; $0CBD..$0CED PIXELSAVE
02134 ;
02135 ;              Table containing the byte of PLAYFIELD memory befor
02136 ;              PLAYFIELD space object pixel into it (star, explosi
02137 ;              for each PLAYFIELD space object (49 bytes).
02138 ;
02139 ;              NOTE: Only bytes 5..48 are used for PLAYFIELD space
02140 ;              this way. Bytes 0..4 are used differently. See PL0H
02141 ;              ($0CBD)..PL4HEIGHT ($0CC1).
02142 ;
02143 ; $0CBD        PL0HEIGHT
02144 ;
02145 ;              Shape height of PLAYER0
02146 ;
02147 ; $0CBE        PL1HEIGHT
02148 ;
02149 ;              Shape height of PLAYER1
02150 ;
02151 ; $0CBF        PL2HEIGHT
02152 ;
02153 ;              Shape height of PLAYER2
02154 ;
02155 ; $0CC0        PL3HEIGHT
02156 ;
02157 ;              Shape height of PLAYER3
```

```
02158 ;
02159 ; $0CC1        PL4HEIGHT
02160 ;
02161 ;              Shape height of PLAYER4
02162 ;
02163 ; $0CEE..$0D1E PIXELBYTE
02164 ;
02165 ;              Table containing a 1-byte bit pattern for 4 pixels
02166 ;              of the space object's pixel, for each PLAYFIELD spa
02167 ;              bytes).
02168 ;
02169 ;              NOTE: Only bytes 5..48 are used for PLAYFIELD space
02170 ;              this way. Bytes 0..4 are used differently. See PL0H
02171 ;              ($0CEE)..PL4HEIGHTNEW ($0CF2).
02172 ;
02173 ; $0CEE        PL0HEIGHTNEW
02174 ;
02175 ;              New shape height of PLAYER0
02176 ;
02177 ; $0CEF        PL1HEIGHTNEW
02178 ;
02179 ;              New shape height of PLAYER1
02180 ;
02181 ; $0CF0        PL2HEIGHTNEW
02182 ;
02183 ;              New shape height of PLAYER2
02184 ;
02185 ; $0CF1        PL3HEIGHTNEW
02186 ;
02187 ;              New shape height of PLAYER3
02188 ;
02189 ; $0CF2        PL4HEIGHTNEW
02190 ;
02191 ;              New shape height of PLAYER4
02192 ;
02193 ; $0D1F..$0D32 TITLETXT
02194 ;
02195 ;              Title text line, contains ATASCII-coded characters
02196 ;
02197 ; $0D35..$0DE8 GCPFMEM
02198 ;
02199 ;              Galactic Chart PLAYFIELD memory (20 characters x 9
02200 ;              bytes)
02201 ;
02202 ; $0DE9..$0EE8 MAPTO80
02203 ;
02204 ;              Lookup table to convert values in $00..$FF to value
02205 ;              (255 bytes). Used to map position vector components
02206 ;              to pixel row or column numbers relative to the PLAY
02207 ;
02208 ; $0EE9..$0FE8 MAPTOBCD99
02209 ;
02210 ;              Lookup table to convert values in $00..$FF to BCD-v
02211 ;              00..99 (255 bytes). Used in subroutines UPDPANEL ($
02212 ;              SHOWDIGITS ($B8CD) to convert values to a 2-digit d
02213 ;              value of the Control Panel Display.
02214 ;
02215 ; $1000..$1F77 PFMEM
02216 ;
```

```
                02217 ;              PLAYFIELD graphics memory (40 bytes x 100 rows = 40
                02218 ;              byte stores 4 pixels, 40 bytes = 160 pixels in GRAP
                02219 ;              PLAYFIELD row of pixels).
                02220 ;
                02221 ;              NOTE: The Display List displays only PLAYFIELD rows
                02222
                02223 ;*****************************************************************
                02224 ;*
                02225 ;*                          S Y S T E M   S Y M B O L S
                02226 ;*
                02227 ;*****************************************************************
                02228
=0200           02229 VDSLST          = $0200                          ; Display List I
=0216           02230 VIMIRQ          = $0216                          ; Interrupt requ
=0222           02231 VVBLKI          = $0222                          ; Vertical blank
=D000           02232 HPOSP0          = $D000                          ; Horizontal pos
=D001           02233 HPOSP1          = $D001                          ; Horizontal pos
=D002           02234 HPOSP2          = $D002                          ; Horizontal pos
=D003           02235 HPOSP3          = $D003                          ; Horizontal pos
=D004           02236 HPOSM0          = $D004                          ; Horizontal pos
=D005           02237 HPOSM1          = $D005                          ; Horizontal pos
=D006           02238 HPOSM2          = $D006                          ; Horizontal pos
=D007           02239 HPOSM3          = $D007                          ; Horizontal pos
=D008           02240 M0PL            = $D008                          ; MISSILE0 to PL
=D009           02241 M1PL            = $D009                          ; MISSILE1 to PL
=D00A           02242 M2PL            = $D00A                          ; MISSILE2 to PL
=D00B           02243 M3PL            = $D00B                          ; MISSILE3 to PL
=D00F           02244 P3PL            = $D00F                          ; PLAYER3 to PLA
=D010           02245 TRIG0           = $D010                          ; Joystick 0 tri
=D012           02246 COLPM0          = $D012                          ; Color and brig
=D016           02247 COLPF0          = $D016                          ; Color and brig
=D01B           02248 PRIOR           = $D01B                          ; Priority selec
=D01D           02249 GRACTL          = $D01D                          ; Graphics contr
=D01E           02250 HITCLR          = $D01E                          ; Clear collisio
=D01F           02251 CONSOL          = $D01F                          ; Function keys
=D200           02252 AUDF1           = $D200                          ; Audio channel
=D202           02253 AUDF2           = $D202                          ; Audio channel
=D203           02254 AUDC2           = $D203                          ; Audio channel
=D204           02255 AUDF3           = $D204                          ; Audio channel
=D205           02256 AUDC3           = $D205                          ; Audio channel
=D206           02257 AUDF4           = $D206                          ; Audio channel
=D207           02258 AUDC4           = $D207                          ; Audio channel
=D208           02259 AUDCTL          = $D208                          ; Audio control
=D209           02260 KBCODE          = $D209                          ; Keyboard code
=D209           02261 STIMER          = $D209                          ; Start POKEY ti
=D20A           02262 RANDOM          = $D20A                          ; Random number
=D20E           02263 IRQEN           = $D20E                          ; Interrupt requ
=D20F           02264 SKCTL           = $D20F                          ; Serial port co
=D300           02265 PORTA           = $D300                          ; Port A
=D302           02266 PACTL           = $D302                          ; Port A control
=D400           02267 DMACTL          = $D400                          ; Direct Memory
=D402           02268 DLIST           = $D402                          ; Display List p
=D407           02269 PMBASE          = $D407                          ; Player/Missile
=D409           02270 CHBASE          = $D409                          ; Character set
=D40A           02271 WSYNC           = $D40A                          ; Wait for horiz
=D40B           02272 VCOUNT          = $D40B                          ; Vertical line
=D40E           02273 NMIEN           = $D40E                          ; Non-maskable i
=E000           02274 ROMCHARSET      = $E000                          ; ROM character
                02275
```

```
          02276 ;*****************************************************************
          02277 ;*
          02278 ;*                          G A M E   S Y M B O L S
          02279 ;*
          02280 ;*****************************************************************
          02281
=0062     02282 MISSIONLEVEL    = $62
=0063     02283 FKEYCODE        = $63
=0064     02284 ISDEMOMODE      = $64
=0065     02285 NEWTITLEPHR     = $65
=0066     02286 IDLECNTHI       = $66
=0067     02287 ISVBISYNC       = $67
=0068     02288 MEMPTR          = $68
          02289
=006A     02290 DIVIDEND        = $6A
=006D     02291 JOYSTICKDELTA   = $6D
          02292
          02293
=0070     02294 VELOCITYLO      = $70
=0071     02295 NEWVELOCITY     = $71
=0072     02296 COUNT8          = $72
=0073     02297 EXPLLIFE        = $73
=0074     02298 CLOCKTIM        = $74
=0075     02299 DOCKSTATE       = $75
=0076     02300 COUNT256        = $76
=0077     02301 IDLECNTLO       = $77
=0078     02302 ZYLONUNITTIM    = $78
=0079     02303 MAXSPCOBJIND    = $79
=007A     02304 OLDMAXSPCOBJIND = $7A
=007B     02305 ISSTARBASESECT  = $7B
=007C     02306 ISTRACKCOMPON   = $7C
=007D     02307 DRAINSHIELDS    = $7D
=007E     02308 DRAINATTCOMP    = $7E
=007F     02309 ENERGYCNT       = $7F
=0080     02310 DRAINENGINES    = $80
=0081     02311 SHIELDSCOLOR    = $81
=0082     02312 PL3HIT          = $82
=0083     02313 PL4HIT          = $83
=0084     02314 OLDTRIG0        = $84
          02315
=0086     02316 ISTRACKING      = $86
=0087     02317 BARRELNR        = $87
=0088     02318 LOCKONLIFE      = $88
=0089     02319 PLTRACKED       = $89
=008A     02320 HITBADNESS      = $8A
=008B     02321 REDALERTLIFE    = $8B
=008C     02322 WARPDEPRROW     = $8C
=008D     02323 WARPDEPRCOLUMN  = $8D
=008E     02324 WARPARRVROW     = $8E
=008F     02325 WARPARRVCOLUMN  = $8F
=0090     02326 CURRSECTOR      = $90
=0091     02327 WARPENERGY      = $91
=0092     02328 ARRVSECTOR      = $92
=0093     02329 HUNTSECTOR      = $93
=0094     02330 HUNTSECTCOLUMN  = $94
=0095     02331 HUNTSECTROW     = $95
=0096     02332 NEWZYLONDIST    = $96
=009E     02333 OLDZYLONDIST    = $9E
=009F     02334 HUNTTIM         = $9F
```

```
=00A0     02335 BLIPCOLUMN      = $A0
=00A1     02336 BLIPROW         = $A1
=00A2     02337 BLIPCYCLECNT    = $A2
=00A3     02338 ISINLOCKON      = $A3
=00A4     02339 DIRLEN          = $A4
=00A5     02340 PENROW          = $A5
=00A6     02341 PENCOLUMN       = $A6
=00A7     02342 CTRLDZYLON      = $A7
=00A8     02343 ZYLONFLPAT0     = $A8
=00A9     02344 ZYLONFLPAT1     = $A9
=00AA     02345 MILESTTIM0      = $AA
=00AB     02346 MILESTTIM1      = $AB
=00AC     02347 MILESTVELINDZ0  = $AC
=00AD     02348 MILESTVELINDZ1  = $AD
=00AE     02349 MILESTVELINDX0  = $AE
=00AF     02350 MILESTVELINDX1  = $AF
=00B0     02351 MILESTVELINDY0  = $B0
=00B1     02352 MILESTVELINDY1  = $B1
=00B2     02353 ZYLONVELINDZ0   = $B2
=00B3     02354 ZYLONVELINDZ1   = $B3
=00B4     02355 ZYLONVELINDX0   = $B4
=00B5     02356 ZYLONVELINDX1   = $B5
=00B6     02357 ZYLONVELINDY0   = $B6
=00B7     02358 ZYLONVELINDY1   = $B7
=00B8     02359 ISBACKATTACK0   = $B8
=00B9     02360 ISBACKATTACK1   = $B9
=00BA     02361 ZYLONTIMX0      = $BA
=00BB     02362 ZYLONTIMX1      = $BB
=00BC     02363 ZYLONTIMY0      = $BC
=00BD     02364 ZYLONTIMY1      = $BD
=00BE     02365 TORPEDODELAY    = $BE
=00BF     02366 ZYLONATTACKER   = $BF
=00C0     02367 WARPSTATE       = $C0
=00C1     02368 VELOCITYHI      = $C1
=00C2     02369 TRAILDELAY      = $C2
=00C3     02370 TRAILIND        = $C3
=00C4     02371 WARPTEMPCOLUMN  = $C4
=00C5     02372 WARPTEMPROW     = $C5
=00C6     02373 VEERMASK        = $C6
=00C7     02374 VICINITYMASK    = $C7
=00C8     02375 JOYSTICKX       = $C8
=00C9     02376 JOYSTICKY       = $C9
=00CA     02377 KEYCODE         = $CA
=00CB     02378 SCORE           = $CB
=00CD     02379 SCOREDRANKIND   = $CD
=00CE     02380 SCOREDCLASSIND  = $CE
=00CF     02381 TITLELIFE       = $CF
=00D0     02382 SHIPVIEW        = $D0
=00D1     02383 TITLEPHR        = $D1
=00D2     02384 BEEPFRQIND      = $D2
=00D3     02385 BEEPREPEAT      = $D3
=00D4     02386 BEEPTONELIFE    = $D4
=00D5     02387 BEEPPAUSELIFE   = $D5
=00D6     02388 BEEPPRIORITY    = $D6
=00D7     02389 BEEPFRQSTART    = $D7
=00D8     02390 BEEPLIFE        = $D8
=00D9     02391 BEEPTOGGLE      = $D9
=00DA     02392 NOISETORPTIM    = $DA
=00DB     02393 NOISEEXPLTIM    = $DB
```

```
=00DC     02394 NOISEAUDC2      = $DC
=00DD     02395 NOISEAUDC3      = $DD
=00DE     02396 NOISEAUDF1      = $DE
=00DF     02397 NOISEAUDF2      = $DF
=00E0     02398 NOISEFRQINC     = $E0
=00E1     02399 NOISELIFE       = $E1
=00E2     02400 NOISEZYLONTIM   = $E2
=00E3     02401 NOISEHITLIFE    = $E3
=00E4     02402 PL0SHAPOFF      = $E4
=00E5     02403 PL1SHAPOFF      = $E5
=00E6     02404 PL2SHAPOFF      = $E6
=00E7     02405 PL3SHAPOFF      = $E7
=00E8     02406 PL4SHAPOFF      = $E8
=00E9     02407 PL0LIFE         = $E9
=00EA     02408 PL1LIFE         = $EA
=00EB     02409 PL2LIFE         = $EB
=00EC     02410 PL3LIFE         = $EC
=00ED     02411 PL4LIFE         = $ED
=00EE     02412 PL0COLOR        = $EE
=00EF     02413 PL1COLOR        = $EF
=00F0     02414 PL2COLOR        = $F0
=00F1     02415 PL3COLOR        = $F1
=00F2     02416 PF0COLOR        = $F2
=00F3     02417 PF1COLOR        = $F3
=00F4     02418 PF2COLOR        = $F4
=00F5     02419 PF3COLOR        = $F5
=00F6     02420 BGRCOLOR        = $F6
=00F7     02421 PF0COLORDLI     = $F7
=00F8     02422 PF1COLORDLI     = $F8
=00F9     02423 PF2COLORDLI     = $F9
=00FA     02424 PF3COLORDLI     = $FA
=00FB     02425 BGRCOLORDLI     = $FB
=0280     02426 DSPLST          = $0280
=0300     02427 PL4DATA         = $0300
=0400     02428 PL0DATA         = $0400
=0500     02429 PL1DATA         = $0500
=0600     02430 PL2DATA         = $0600
=0700     02431 PL3DATA         = $0700
=0800     02432 PFMEMROWLO      = $0800
=0864     02433 PFMEMROWHI      = $0864
=08C9     02434 GCMEMMAP        = $08C9
=0949     02435 PANELTXT        = $0949
=094B     02436 VELOCD1         = $094B
=0950     02437 KILLCNTD1       = $0950
=0955     02438 ENERGYD1        = $0955
=095A     02439 TRACKC1         = $095A
=095C     02440 TRACKDIGIT      = $095C
=0960     02441 THETAC1         = $0960
=0966     02442 PHIC1           = $0966
=096C     02443 RANGEC1         = $096C
=0971     02444 GCTXT           = $0971
=097D     02445 GCWARPD1        = $097D
=098D     02446 GCTRGCNT        = $098D
=0992     02447 GCSTATPHO       = $0992
=0993     02448 GCSTATENG       = $0993
=0994     02449 GCSTATSHL       = $0994
=0995     02450 GCSTATCOM       = $0995
=0996     02451 GCSTATLRS       = $0996
=0997     02452 GCSTATRAD       = $0997
```

```
=09A3     02453 GCSTARDAT       = $09A3
=09AD     02454 ZPOSSIGN        = $09AD
=09AF     02455 PL2ZPOSSIGN     = $09AF
=09B0     02456 PL3ZPOSSIGN     = $09B0
=09B1     02457 PL4ZPOSSIGN     = $09B1
=09DE     02458 XPOSSIGN        = $09DE
=09E0     02459 PL2XPOSSIGN     = $09E0
=09E1     02460 PL3XPOSSIGN     = $09E1
=09E2     02461 PL4XPOSSIGN     = $09E2
=0A0F     02462 YPOSSIGN        = $0A0F
=0A11     02463 PL2YPOSSIGN     = $0A11
=0A12     02464 PL3YPOSSIGN     = $0A12
=0A13     02465 PL4YPOSSIGN     = $0A13
=0A40     02466 ZPOSHI          = $0A40
=0A40     02467 PL0ZPOSHI       = $0A40
=0A42     02468 PL2ZPOSHI       = $0A42
=0A43     02469 PL3ZPOSHI       = $0A43
=0A44     02470 PL4ZPOSHI       = $0A44
=0A71     02471 XPOSHI          = $0A71
=0A73     02472 PL2XPOSHI       = $0A73
=0A74     02473 PL3XPOSHI       = $0A74
=0A75     02474 PL4XPOSHI       = $0A75
=0AA2     02475 YPOSHI          = $0AA2
=0AA4     02476 PL2YPOSHI       = $0AA4
=0AA5     02477 PL3YPOSHI       = $0AA5
=0AA6     02478 PL4YPOSHI       = $0AA6
=0AD3     02479 ZPOSLO          = $0AD3
=0AD5     02480 PL2ZPOSLO       = $0AD5
=0AD6     02481 PL3ZPOSLO       = $0AD6
=0AD7     02482 PL4ZPOSLO       = $0AD7
=0B04     02483 XPOSLO          = $0B04
=0B06     02484 PL2XPOSLO       = $0B06
=0B07     02485 PL3XPOSLO       = $0B07
=0B08     02486 PL4XPOSLO       = $0B08
=0B35     02487 YPOSLO          = $0B35
=0B37     02488 PL2YPOSLO       = $0B37
=0B38     02489 PL3YPOSLO       = $0B38
=0B39     02490 PL4YPOSLO       = $0B39
=0B66     02491 ZVEL            = $0B66
=0B66     02492 PL0ZVEL         = $0B66
=0B67     02493 PL1ZVEL         = $0B67
=0B68     02494 PL2ZVEL         = $0B68
=0B69     02495 PL3ZVEL         = $0B69
=0B6A     02496 PL4ZVEL         = $0B6A
=0B97     02497 XVEL            = $0B97
=0B97     02498 PL0XVEL         = $0B97
=0B98     02499 PL1XVEL         = $0B98
=0B99     02500 PL2XVEL         = $0B99
=0B9A     02501 PL3XVEL         = $0B9A
=0B9B     02502 PL4XVEL         = $0B9B
=0BC8     02503 YVEL            = $0BC8
=0BC8     02504 PL0YVEL         = $0BC8
=0BC9     02505 PL1YVEL         = $0BC9
=0BCA     02506 PL2YVEL         = $0BCA
=0BCB     02507 PL3YVEL         = $0BCB
=0BCC     02508 PL4YVEL         = $0BCC
=0BF9     02509 PIXELROWNEW     = $0BF9
=0BF9     02510 PL0ROWNEW       = $0BF9
=0BFA     02511 PL1ROWNEW       = $0BFA
```

```
=0BFB     02512 PL2ROWNEW       = $0BFB
=0BFC     02513 PL3ROWNEW       = $0BFC
=0BFD     02514 PL4ROWNEW       = $0BFD
=0C2A     02515 PIXELCOLUMN     = $0C2A
=0C2A     02516 PL0COLUMN       = $0C2A
=0C2B     02517 PL1COLUMN       = $0C2B
=0C2C     02518 PL2COLUMN       = $0C2C
=0C2D     02519 PL3COLUMN       = $0C2D
=0C2E     02520 PL4COLUMN       = $0C2E
=0C5B     02521 PIXELROW        = $0C5B
=0C5B     02522 PL0ROW          = $0C5B
=0C5C     02523 PL1ROW          = $0C5C
=0C5D     02524 PL2ROW          = $0C5D
=0C5E     02525 PL3ROW          = $0C5E
=0C5F     02526 PL4ROW          = $0C5F
=0C8C     02527 PIXELBYTEOFF    = $0C8C
=0C8C     02528 PL0SHAPTYPE     = $0C8C
=0C8D     02529 PL1SHAPTYPE     = $0C8D
=0C8E     02530 PL2SHAPTYPE     = $0C8E
=0C8F     02531 PL3SHAPTYPE     = $0C8F
=0C90     02532 PL4SHAPTYPE     = $0C90
=0CBD     02533 PIXELSAVE       = $0CBD
=0CBD     02534 PL0HEIGHT       = $0CBD
=0CBE     02535 PL1HEIGHT       = $0CBE
=0CBF     02536 PL2HEIGHT       = $0CBF
=0CC0     02537 PL3HEIGHT       = $0CC0
=0CC1     02538 PL4HEIGHT       = $0CC1
=0CEE     02539 PIXELBYTE       = $0CEE
=0CEE     02540 PL0HEIGHTNEW    = $0CEE
=0CEF     02541 PL1HEIGHTNEW    = $0CEF
=0CF0     02542 PL2HEIGHTNEW    = $0CF0
=0CF1     02543 PL3HEIGHTNEW    = $0CF1
=0CF2     02544 PL4HEIGHTNEW    = $0CF2
=0D1F     02545 TITLETXT        = $0D1F
=0D35     02546 GCPFMEM         = $0D35
=0DE9     02547 MAPTO80         = $0DE9
=0EE9     02548 MAPTOBCD99      = $0EE9
=1000     02549 PFMEM           = $1000
          02550
          02551                 *= $A000
          02552
          02553 ;*****************************************************************
          02554 ;*
          02555 ;*               G A M E   D A T A   ( P A R T   1   O F   2 )
          02556 ;*
          02557 ;*****************************************************************
          02558
          02559 ;*** Number of space objects ************************************
          02560
=0005     02561 NUMSPCOBJ.PL    = 5                              ; Number of PLAY
=000C     02562 NUMSPCOBJ.STARS = 12                             ; Number of PLAY
=0011     02563 NUMSPCOBJ.NORM  = NUMSPCOBJ.PL+NUMSPCOBJ.STARS  ; Normal number
=0031     02564 NUMSPCOBJ.ALL   = 49                             ; Maximum number
          02565
          02566 ;*** PLAYER shape data offsets *********************************
          02567
=0000     02568 SHAP.TORPEDO    = $00                            ; Photon torpedo
=0010     02569 SHAP.ZFIGHTER   = $10                            ; Zylon fighter
=0020     02570 SHAP.STARBASEL  = $20                            ; Starbase (left
```

```
=0030     02571 SHAP.STARBASEC  = $30                          ; Starbase (cent
=0040     02572 SHAP.STARBASER  = $40                          ; Starbase (righ
=0050     02573 SHAP.TRANSVSSL  = $50                          ; Transfer vesse
=0060     02574 SHAP.METEOR     = $60                          ; Meteor
=0070     02575 SHAP.ZCRUISER   = $70                          ; Zylon cruiser
=0080     02576 SHAP.ZBASESTAR  = $80                          ; Zylon basestar
=0090     02577 SHAP.HYPERWARP  = $90                          ; Hyperwarp Targ
          02578
          02579 ;*** ROM character set constants ********************************
=0000     02580 ROM.SPC         = $00                          ; ROM character
=000E     02581 ROM.DOT         = $0E                          ; ROM character
=0010     02582 ROM.0           = $10                          ; ROM character
=0011     02583 ROM.1           = $11                          ; ROM character
=0012     02584 ROM.2           = $12                          ; ROM character
=0013     02585 ROM.3           = $13                          ; ROM character
=0014     02586 ROM.4           = $14                          ; ROM character
=0015     02587 ROM.5           = $15                          ; ROM character
=0019     02588 ROM.9           = $19                          ; ROM character
=001A     02589 ROM.COLON       = $1A                          ; ROM character
=0021     02590 ROM.A           = $21                          ; ROM character
=0023     02591 ROM.C           = $23                          ; ROM character
=0024     02592 ROM.D           = $24                          ; ROM character
=0025     02593 ROM.E           = $25                          ; ROM character
=0027     02594 ROM.G           = $27                          ; ROM character
=002C     02595 ROM.L           = $2C                          ; ROM character
=002E     02596 ROM.N           = $2E                          ; ROM character
=0030     02597 ROM.P           = $30                          ; ROM character
=0032     02598 ROM.R           = $32                          ; ROM character
=0033     02599 ROM.S           = $33                          ; ROM character
=0034     02600 ROM.T           = $34                          ; ROM character
=0037     02601 ROM.W           = $37                          ; ROM character
=0039     02602 ROM.Y           = $39                          ; ROM character
          02603
          02604 ;*** Custom character set constants *****************************
=0040     02605 CCS.COL1        = $40                          ; COLOR1 bits fo
=0080     02606 CCS.COL2        = $80                          ; COLOR2 bits fo
=00C0     02607 CCS.COL3        = $C0                          ; COLOR3 bits fo
          02608
=0000     02609 CCS.0           = 0                            ; Custom charact
=0001     02610 CCS.1           = 1                            ; Custom charact
=0002     02611 CCS.2           = 2                            ; Custom charact
=0003     02612 CCS.3           = 3                            ; Custom charact
=0004     02613 CCS.4           = 4                            ; Custom charact
=0005     02614 CCS.5           = 5                            ; Custom charact
=0006     02615 CCS.6           = 6                            ; Custom charact
=0007     02616 CCS.7           = 7                            ; Custom charact
=0008     02617 CCS.8           = 8                            ; Custom charact
=0009     02618 CCS.9           = 9                            ; Custom charact
=000A     02619 CCS.SPC         = 10                           ; Custom charact
=000B     02620 CCS.COLON       = 11                           ; Custom charact
=000C     02621 CCS.BORDERSW    = 12                           ; Custom charact
=000D     02622 CCS.E           = 13                           ; Custom charact
=000E     02623 CCS.INF         = 14                           ; Custom charact
=000F     02624 CCS.MINUS       = 15                           ; Custom charact
=0010     02625 CCS.PLUS        = 16                           ; Custom charact
=0011     02626 CCS.PHI         = 17                           ; Custom charact
=0012     02627 CCS.V           = 18                           ; Custom charact
=0013     02628 CCS.R           = 19                           ; Custom charact
=0014     02629 CCS.THETA       = 20                           ; Custom charact
```

```
    =0015     02630 CCS.K           = 21                              ; Custom charact
    =0016     02631 CCS.T           = 22                              ; Custom charact
    =0017     02632 CCS.C           = 23                              ; Custom charact
    =0018     02633 CCS.BORDERS     = 24                              ; Custom charact
    =0019     02634 CCS.BORDERW     = 25                              ; Custom charact
    =001A     02635 CCS.CORNERSW    = 26                              ; Custom charact
    =001B     02636 CCS.STARBASE    = 27                              ; Custom charact
    =001C     02637 CCS.4ZYLONS     = 28                              ; Custom charact
    =001D     02638 CCS.3ZYLONS     = 29                              ; Custom charact
    =001E     02639 CCS.2ZYLONS     = 30                              ; Custom charact
              02640
              02641 ;*** Custom character set ***************************************
              02642 ;
              02643 ; 0        1        2        3        4        5        6        7
              02644 ; ........ ........ ........ ........ ........ ........ ........ .
              02645 ; .#######  ..##....  .####...  .####...  .##.....  .####...  .####...  .
              02646 ; .#...###  ...#....  ....#...  ....#...  .##.....  .#......  .#..#...  .
              02647 ; .#...###  ...#....  ....#...  ....#...  .##.....  .#......  .#......  .
              02648 ; .#...###  ...#....  .####...  .#####..  .##.##..  .####...  .#......  .
              02649 ; .#...###  ..###...  .#......  ....##..  .#####..  ....#...  .######.  .
              02650 ; .#...###  ..###...  .#......  ....##..  ....##..  ....#...  .#....#.  .
              02651 ; .#######  ..###...  .####...  .#####..  ....##..  .####...  .######.  .
              02652 ;
              02653 ; 8        9        10       11       12       13       14       1
              02654 ; ........ ........ ........ ..###... #.......  ........ ........ .
              02655 ; ..###...  .#####..  ........ ..###... #.......  ..####..  .##..##.  .
              02656 ; ..#.#...  .#...#..  ........ ..###... #.......  ..#.....  #..##..#  .
              02657 ; ..#.#...  .#...#..  ........ ........ #.......  ..#.....  #..##..#  .
              02658 ; .#####..  .#####..  ........ ........ #.......  .####...  #..##..#  .
              02659 ; .##.##..  ....##..  ........ ..###... #.......  .##.....  .##..##.  .
              02660 ; .##.##..  ....##..  ........ ..###... #.......  .##.....  ........ .
              02661 ; .#####..  ....##..  ........ ..###... ########  .#####..  ........ .
              02662 ;
              02663 ; 16       17       18       19       20       21       22       2
              02664 ; ........ ........ .##..##. ........ ........ ........ #######. #
              02665 ; ...##...  ...##...  .##..##.  .#####..  ...###..  .#...##.  #..#..#.  #
              02666 ; ...##...  .######.  .##..##.  .#...#..  ..#####.  .#...##.  ...#....  #
              02667 ; ...##...  ##.##.##  .##..##.  .#...#..  .##...##  .#...#..  ...##...  #
              02668 ; .######.  #..##..#  .##..##.  .#####..  .#.###.#  .#####..  ...##...  #
              02669 ; ...##...  ##.##.##  ..#.##..  .##.#...  .##...##  .##..#..  ...##...  #
              02670 ; ...##...  .######.  ..###...  .##.##..  ..#####.  .##..##.  ...##...  #
              02671 ; ...##...  ...##...  ..##....  .##.##..  ...###..  .##..##.  ...##...  #
              02672 ;
              02673 ; 24       25       26       27       28       29       30
              02674 ; ........ #.......  ........ #.......  #.......  #.......  #.......
              02675 ; ........ #.......  ........ #.#.#.#.  #..##...  #...###.  #.##....
              02676 ; ........ #.......  ........ #..###..  #.......  #.......  #..##...
              02677 ; ........ #.......  ........ #.#####.  #.##.##.  #.###...  #.#####.
              02678 ; ........ #.......  ........ #..###..  #.......  #.......  #..##...
              02679 ; ........ #.......  ........ #.#.#.#.  #...##..  #..###..  #.##....
              02680 ; ........ #.......  ........ #.......  #.......  #.......  #.......
              02681 ; ######## #.......  #.......  ########  ########  ########  ########
              02682
A000 007F4747 02683 CHARSET         .BYTE $00,$7F,$47,$47,$47,$47,$47,$7F ; Custom cha
A004 4747477F
A008 00301010 02684                 .BYTE $00,$30,$10,$10,$10,$38,$38,$38 ; Custom cha
A00C 10383838
A010 00780808 02685                 .BYTE $00,$78,$08,$08,$78,$40,$40,$78 ; Custom cha
A014 78404078
```

```
A018 00780808 02686                .BYTE $00,$78,$08,$08,$7C,$0C,$0C,$7C ; Custom cha
A01C 7C0C0C7C
A020 00606060 02687                .BYTE $00,$60,$60,$60,$6C,$7C,$0C,$0C ; Custom cha
A024 6C7C0C0C
A028 00784040 02688                .BYTE $00,$78,$40,$40,$78,$08,$08,$78 ; Custom cha
A02C 78080878
A030 00784840 02689                .BYTE $00,$78,$48,$40,$40,$7E,$42,$7E ; Custom cha
A034 407E427E
A038 007C4404 02690                .BYTE $00,$7C,$44,$04,$1C,$10,$10,$10 ; Custom cha
A03C 1C101010
A040 00382828 02691                .BYTE $00,$38,$28,$28,$7C,$6C,$6C,$7C ; Custom cha
A044 7C6C6C7C
A048 007C4444 02692                .BYTE $00,$7C,$44,$44,$7C,$0C,$0C,$0C ; Custom cha
A04C 7C0C0C0C
A050 00000000 02693                .BYTE $00,$00,$00,$00,$00,$00,$00,$00 ; Custom cha
A054 00000000
A058 38383800 02694                .BYTE $38,$38,$38,$00,$00,$38,$38,$38 ; Custom cha
A05C 00383838
A060 80808080 02695                .BYTE $80,$80,$80,$80,$80,$80,$80,$FF ; Custom cha
A064 808080FF
A068 003C2020 02696                .BYTE $00,$3C,$20,$20,$78,$60,$60,$7C ; Custom cha
A06C 7860607C
A070 00669999 02697                .BYTE $00,$66,$99,$99,$99,$66,$00,$00 ; Custom cha
A074 99660000
A078 0000007E 02698                .BYTE $00,$00,$00,$7E,$00,$00,$00,$00 ; Custom cha
A07C 00000000
A080 00181818 02699                .BYTE $00,$18,$18,$18,$7E,$18,$18,$18 ; Custom cha
A084 7E181818
A088 00187EDB 02700                .BYTE $00,$18,$7E,$DB,$99,$DB,$7E,$18 ; Custom cha
A08C 99DB7E18
A090 66666666 02701                .BYTE $66,$66,$66,$66,$66,$2C,$38,$30 ; Custom cha
A094 662C3830
A098 007C4444 02702                .BYTE $00,$7C,$44,$44,$7C,$68,$6C,$6C ; Custom cha
A09C 7C686C6C
A0A0 001C3E63 02703                .BYTE $00,$1C,$3E,$63,$5D,$63,$3E,$1C ; Custom cha
A0A4 5D633E1C
A0A8 00464644 02704                .BYTE $00,$46,$46,$44,$7C,$64,$66,$66 ; Custom cha
A0AC 7C646666
A0B0 FE921018 02705                .BYTE $FE,$92,$10,$18,$18,$18,$18,$18 ; Custom cha
A0B4 18181818
A0B8 FC8C8C80 02706                .BYTE $FC,$8C,$8C,$80,$80,$80,$84,$FC ; Custom cha
A0BC 808084FC
A0C0 00000000 02707                .BYTE $00,$00,$00,$00,$00,$00,$00,$FF ; Custom cha
A0C4 000000FF
A0C8 80808080 02708                .BYTE $80,$80,$80,$80,$80,$80,$80,$80 ; Custom cha
A0CC 80808080
A0D0 00000000 02709                .BYTE $00,$00,$00,$00,$00,$00,$00,$80 ; Custom cha
A0D4 00000080
A0D8 80AA9CBE 02710                .BYTE $80,$AA,$9C,$BE,$9C,$AA,$80,$FF ; Custom cha
A0DC 9CAA80FF
A0E0 809880B6 02711                .BYTE $80,$98,$80,$B6,$80,$8C,$80,$FF ; Custom cha
A0E4 808C80FF
A0E8 808E80B8 02712                .BYTE $80,$8E,$80,$B8,$80,$9C,$80,$FF ; Custom cha
A0EC 809C80FF
A0F0 80B098BE 02713                .BYTE $80,$B0,$98,$BE,$98,$B0,$80,$FF ; Custom cha
A0F4 98B080FF
              02714
              02715 ;*** Header text of Long-Range Scan view (shares spaces with follo
A0F8 00006C6F 02716 LRSHEADER      .BYTE $00,$00,$6C,$6F,$6E,$67,$00,$72 ; "  LONG RA
```

```
A0FC 6E670072
A100 616E6765 02717                .BYTE $61,$6E,$67,$65,$00,$73,$63,$61
A104 00736361
A108 6E       02718                .BYTE $6E
              02719
              02720 ;*** Header text of Aft view (shares spaces with following header)
A109 00000000 02721 AFTHEADER      .BYTE $00,$00,$00,$00,$00,$00,$61,$66 ; "      AFT
A10D 00006166
A111 74007669 02722                .BYTE $74,$00,$76,$69,$65,$77,$00,$00
A115 65770000
A119 00       02723                .BYTE $00
              02724
              02725 ;*** Header text of Galactic Chart view ************************
A11A 00000067 02726 GCHEADER       .BYTE $00,$00,$00,$67,$61,$6C,$61,$63 ; "   GALACT
A11E 616C6163
A122 74696300 02727                .BYTE $74,$69,$63,$00,$63,$68,$61,$72
A126 63686172
A12A 74000000 02728                .BYTE $74,$00,$00,$00
              02729
              02730 ;*** Display List of Galactic Chart view ************************
A12E 60       02731 DLSTGC         .BYTE $60                          ; BLK7
A12F 461AA1   02732                .BYTE $46,<GCHEADER,>GCHEADER      ; GR1 @ GCHE
A132 F0       02733                .BYTE $F0                          ; BLK8 + DLI
A133 47350D   02734                .BYTE $47,<GCPFMEM,>GCPFMEM        ; GR2 @ GCPF
A136 07       02735                .BYTE $07                          ; GR2
A137 07       02736                .BYTE $07                          ; GR2
A138 07       02737                .BYTE $07                          ; GR2
A139 07       02738                .BYTE $07                          ; GR2
A13A 07       02739                .BYTE $07                          ; GR2
A13B 07       02740                .BYTE $07                          ; GR2
A13C 07       02741                .BYTE $07                          ; GR2
A13D 07       02742                .BYTE $07                          ; GR2
A13E 80       02743                .BYTE $80                          ; BLK1 + DLI
A13F 461F0D   02744                .BYTE $46,<TITLETXT,>TITLETXT      ; GR1 @ TITL
A142 467109   02745                .BYTE $46,<GCTXT,>GCTXT            ; GR1 @ GCTX
A145 06       02746                .BYTE $06                          ; GR1
A146 06       02747                .BYTE $06                          ; GR1
A147 418002   02748                .BYTE $41,<DSPLST,>DSPLST          ; JMP @ DSPL
              02749
              02750 ;*****************************************************************
              02751 ;*
              02752 ;*                          G A M E   C O D E
              02753 ;*
              02754 ;*****************************************************************
              02755
              02756 ;*****************************************************************
              02757 ;*
              02758 ;*                              INITCOLD
              02759 ;*
              02760 ;*                     Initialize game (Cold start)
              02761 ;*
              02762 ;*****************************************************************
              02763
              02764 ; DESCRIPTION
              02765 ;
              02766 ; Initializes the game, then continues into the game loop at GAMEL
              02767 ;
              02768 ; There are four entry points to initialization:
              02769 ;
```

```
02770 ; (1)  INITCOLD ($A14A) is entered at initial cartridge startup (c
02771 ;      This initializes POKEY, resets the idle counter, sets the m
02772 ;      to NOVICE mission, and clears the function key code. POKEY
02773 ;      receive keyboard input. Code execution continues into INITS
02774 ;      below.
02775 ;
02776 ; (2)  INITSELECT ($A15A) is entered from GAMELOOP ($A1F3) after t
02777 ;      function key has been pressed. This loads the title phrase
02778 ;      copyright notice. Code execution continues into INITDEMO ($
02779 ;
02780 ; (3)  INITDEMO ($A15C) is entered when the game switches into dem
02781 ;      loads the demo mode flag. Code execution continues into INI
02782 ;      below.
02783 ;
02784 ; (4)  INITSTART ($A15E) is entered from GAMELOOP ($A1F3) after th
02785 ;      function key has been pressed. This enqueues the new title
02786 ;      enables or disables demo mode, depending on the preloaded v
02787 ;
02788 ; Initialization continues with the following steps:
02789 ;
02790 ; (1)  Clear the custom chip registers and zero page game variable
02791 ;      ISVBISYNC ($0067) on.
02792 ;
02793 ;      NOTE: Because of loop jamming there is a loop index oversho
02794 ;      clears memory at $0067..$0166 instead of the game's zero pa
02795 ;      $0067..$00FB. However, this does no harm because memory at
02796 ;      is - at this point in time - a yet unused part of the 6502
02797 ;      (memory addresses $0100..$01FF).
02798 ;
02799 ;      NOTE: At address $A175 a hack is necessary in the source co
02800 ;      STA ISVBISYNC,X instruction with a 16-bit address operand,
02801 ;      an 8-bit (zero page) address operand. The latter would be c
02802 ;      virtually all 6502 assemblers, as ISVBISYNC ($0067) is loca
02803 ;      zero page (memory addresses $0000..$00FF). The reason to fo
02804 ;      address operand is the following: The instruction STA ISVBI
02805 ;      in a loop which iterates the CPU's X register from 0 to 255
02806 ;      memory. By using this instruction with a 16-bit address ope
02807 ;      ("indexed, absolute" mode), memory at $0067..$0166 is clear
02808 ;      code been using the same operation with an 8-bit address op
02809 ;      ("indexed, zero page" mode), memory at $0067..$00FF would h
02810 ;      cleared first, then the indexed address would have wrapped
02811 ;      and cleared memory at $0000..$0066, thus effectively overwr
02812 ;      initialized memory locations.
02813 ;
02814 ; (2)  Initialize the 6502 CPU (reset the stack pointer, disable d
02815 ;
02816 ; (3)  Clear game memory from $0200..$1FFF in subroutine CLRMEM ($
02817 ;
02818 ; (4)  Set the address vectors of the IRQ, VBI, and DLI handlers.
02819 ;
02820 ; (5)  Enable input from Joystick 0.
02821 ;
02822 ; (6)  Enable Player/Missile graphics, providing a fifth PLAYER, a
02823 ;      PLAYER-PLAYFIELD priority.
02824 ;
02825 ;      BUG (at $A1A6): The set PLAYER-PLAYFIELD priority arranges
02826 ;      (PL0..4) in front of the PLAYFIELD (PF0..4) in this specifi
02827 ;      front to back:
02828 ;
```

```
                02829 ;          PL0 > PL1 > PL2 > PL3 > PL4 > PF0, PF1, PF2 > PF4 (BGR)
                02830 ;
                02831 ;      This makes sense as space objects represented by PLAYERs (f
                02832 ;      Zylon ships, photon torpedoes, and meteors) move in front o
                02833 ;      which are part of the PLAYFIELD. However, PLAYERs also move
                02834 ;      the cross hairs, which are also part of the PLAYFIELD. Sugg
                02835 ;      None, technically not possible.
                02836 ;
                02837 ; (7)  Do more initialization in subroutine INITIALIZE ($B3BA).
                02838 ;
                02839 ; (8)  Set display to Front view.
                02840 ;
                02841 ; (9)  Show or hide the Control Panel Display (bottom text window)
                02842 ;      MODDLST ($ADF1), depending on the demo mode flag.
                02843 ;
                02844 ; (10) Initialize our starship's velocity equivalent to speed key
                02845 ;
                02846 ; (11) Enable the Display List.
                02847 ;
                02848 ; (12) Initialize the number of space objects to 16 (5 PLAYER spac
                02849 ;      PLAYFIELD space objects (stars), counted 0..16).
                02850 ;
                02851 ; (13) Set the title phrase to the selected mission level in subro
                02852 ;      ($B223).
                02853 ;
                02854 ; (14) Enable the IRQ, DLI, and VBI interrupts.
                02855 ;
                02856 ; Code execution continues into the game loop at GAMELOOP ($A1F3).
                02857
A14A A900       02858 INITCOLD        LDA #0                ;
A14C 8D0FD2     02859                 STA SKCTL             ; POKEY: Initialization
A14F 8566       02860                 STA IDLECNTHI         ; Reset idle counter
A151 8562       02861                 STA MISSIONLEVEL      ; Mission level := NOVICE
A153 8563       02862                 STA FKEYCODE          ; Clear function key code
A155 A903       02863                 LDA #$03              ; POKEY: Enable keyboard s
A157 8D0FD2     02864                 STA SKCTL             ;
                02865
                02866 ;*** Entry point when SELECT function key was pressed ************
A15A A02F       02867 INITSELECT      LDY #$2F              ; Prep title phrase "COPYR
                02868
                02869 ;*** Entry point when game switches into demo mode ***************
A15C A9FF       02870 INITDEMO        LDA #$FF              ; Prep demo mode flag
                02871
                02872 ;*** Entry point when START function key was pressed *************
A15E 8465       02873 INITSTART       STY NEWTITLEPHR       ; Enqueue new title phrase
A160 8564       02874                 STA ISDEMOMODE        ; Store demo mode flag
                02875
                02876 ;*** More initialization ****************************************
A162 A900       02877                 LDA #0                ; Clear custom chip regist
A164 AA         02878                 TAX                   ;
A165 9D00D0     02879 LOOP001         STA HPOSP0,X          ; Clear $D000..$D0FF (GTIA
A168 9D00D4     02880                 STA DMACTL,X          ; Clear $D400..$D4FF (ANTI
A16B E00F       02881                 CPX #$0F              ;
A16D B003       02882                 BCS SKIP001           ;
A16F 9D00D2     02883                 STA AUDF1,X           ; Clear $D200..$D20E (POKE
                02884
A172 9D00D3     02885 SKIP001         STA PORTA,X           ; Clear $D300..$D3FF (PIA
                02886                                       ; Clear $0067..$0166 (zero
A175 9D         02887                 .BYTE $9D             ; HACK: Force ISVBISYNC,X
```

```
A176 6700     02888                .WORD ISVBISYNC         ; (loop jamming)
A178 E8       02889                INX                     ;
A179 D0EA     02890                BNE LOOP001             ;
              02891
A17B CA       02892                DEX                     ; Reset 6502 CPU stack poi
A17C 9A       02893                TXS                     ;
              02894
A17D D8       02895                CLD                     ; Clear 6502 CPU decimal m
              02896
A17E A902     02897                LDA #$02                ; Clear $0200..$1FFF (game
A180 200FAE   02898                JSR CLRMEM              ;
              02899
A183 A951     02900                LDA #<IRQHNDLR          ; Set IRQ handler (VIMIRQ)
A185 8D1602   02901                STA VIMIRQ              ;
A188 A9A7     02902                LDA #>IRQHNDLR          ;
A18A 8D1702   02903                STA VIMIRQ+1            ;
              02904
A18D A9D1     02905                LDA #<VBIHNDLR          ; Set VBI and DLI handler
A18F 8D2202   02906                STA VVBLKI              ;
A192 A918     02907                LDA #<DLSTHNDLR         ;
A194 8D0002   02908                STA VDSLST              ;
A197 A9A6     02909                LDA #>VBIHNDLR          ;
A199 8D2302   02910                STA VVBLKI+1            ;
A19C A9A7     02911                LDA #>DLSTHNDLR         ;
A19E 8D0102   02912                STA VDSLST+1            ;
              02913
A1A1 A904     02914                LDA #$04                ; PIA: Enable PORTA (Joyst
A1A3 8D02D3   02915                STA PACTL               ;
A1A6 A911     02916                LDA #$11                ; GTIA: Enable PLAYER4, pr
A1A8 8D1BD0   02917                STA PRIOR               ; (PLAYERs in front of sta
A1AB A903     02918                LDA #$03                ; GTIA: Enable DMA for PLA
A1AD 8D1DD0   02919                STA GRACTL              ;
              02920
A1B0 20BAB3   02921                JSR INITIALIZE          ; Init Display List, table
              02922
A1B3 A20A     02923                LDX #$0A                ; Set Front view
A1B5 2045B0   02924                JSR SETVIEW             ;
              02925
A1B8 A564     02926                LDA ISDEMOMODE          ; If in/not in demo mode h
A1BA 2980     02927                AND #$80                ; ...Control Panel Display
A1BC A8       02928                TAY                     ;
A1BD A25F     02929                LDX #$5F                ;
A1BF A908     02930                LDA #$08                ;
A1C1 20F1AD   02931                JSR MODDLST             ;
              02932
A1C4 A920     02933                LDA #32                 ; Init our starship's velo
A1C6 8571     02934                STA NEWVELOCITY         ;
              02935
A1C8 A980     02936                LDA #<DSPLST            ; ANTIC: Set Display List
A1CA 8D02D4   02937                STA DLIST               ;
A1CD A902     02938                LDA #>DSPLST            ;
A1CF 8D03D4   02939                STA DLIST+1             ;
              02940
A1D2 A93E     02941                LDA #$3E                ; ANTIC: Enable Display Li
A1D4 8D00D4   02942                STA DMACTL              ; resolution, PM DMA, norm
              02943
A1D7 A900     02944                LDA #0                  ; ANTIC: Set PM memory bas
A1D9 8D07D4   02945                STA PMBASE              ;
              02946
```

```
A1DC A910     02947                LDA #NUMSPCOBJ.NORM-1   ; Set normal number of spa
A1DE 8579     02948                STA MAXSPCOBJIND        ; (5 PLAYER spc objs + 12
              02949
A1E0 A662     02950                LDX MISSIONLEVEL        ; Set title phrase
A1E2 BC0CBF   02951                LDY MISSIONPHRTAB,X     ; NOVICE, PILOT, WARRIOR,
A1E5 2023B2   02952                JSR SETTITLE            ;
              02953
A1E8 A940     02954                LDA #$40                ; POKEY: Enable keyboard i
A1EA 8D0ED2   02955                STA IRQEN               ;
              02956
A1ED 58       02957                CLI                     ; Enable all IRQs
              02958
A1EE A9C0     02959                LDA #$C0                ; ANTIC: Enable DLI and VB
A1F0 8D0ED4   02960                STA NMIEN               ;
              02961
              02962 ;*****************************************************************
              02963 ;*
              02964 ;*                                GAMELOOP
              02965 ;*
              02966 ;*****************************************************************
              02967
              02968 ; DESCRIPTION
              02969 ;
              02970 ; The game loop is the main part of the game. It is basically an i
              02971 ; that collects input, computes the game state, and updates the di
              02972 ; executes the following steps:
              02973 ;
              02974 ; (1)  Synchronize the start of the game loop with the vertical bl
              02975 ;      the TV beam, which flagged by the Vertical Blank Interrupt
              02976 ;      VBIHNDLR ($A6D1). This prevents screen flicker while the PL
              02977 ;      redrawn at the beginning of the game loop, because during t
              02978 ;      blank phase the TV beam is turned off and nothing is render
              02979 ;      display.
              02980 ;
              02981 ; (2)  Erase all PLAYFIELD space objects (stars, explosion fragmen
              02982 ;      PLAYFIELD that were drawn in the previous game loop iterati
              02983 ;
              02984 ; (3)  Draw the updated PLAYFIELD space objects (stars, explosion
              02985 ;      into the PLAYFIELD (skip this if in hyperspace).
              02986 ;
              02987 ; (4)  If the idle counter has reached its trigger value then clea
              02988 ;      of the PLAYFIELD, an 8 x 2 pixel rectangle with a top-left
              02989 ;      pixel column number 76 and pixel row number 49 (?).
              02990 ;
              02991 ; (5)  Clear all PLAYER shapes.
              02992 ;
              02993 ; (6)  Update the vertical position of all PLAYERs and update all
              02994 ;
              02995 ; (7)  Update the horizontal position of all PLAYERs.
              02996 ;
              02997 ; (8)  Rotate the position vector of all space objects horizontall
              02998 ;      vertically, according to the saved joystick position (skip
              02999 ;      Galactic Chart view) using subroutine ROTATE ($B69B).
              03000 ;
              03001 ; (9)  Move our starship forward in space. Our starship is always
              03002 ;      center of the game's 3D coordinate system, so all space obj
              03003 ;      along the z-axis toward our starship by subtracting a displ
              03004 ;      their z-coordinate. The amount of the displacement depends
              03005 ;      starship's velocity.
```

```
03006 ;
03007 ;      BUG (at $A3C1): This operation is not applied to Photon tor
03008 ;      Suggested fix: Remove LDA PL0SHAPTYPE,X and BEQ SKIP011.
03009 ;
03010 ; (10) Add the proper velocity vector of all space objects to thei
03011 ;      vector (except for stars, which do not have any proper moti
03012 ;
03013 ;      BUG (at $A419): The correct maximum loop index is NUMSPCOBJ
03014 ;      instead of 144. Suggested fix: Replace CMP #144 with CMP #1
03015 ;
03016 ; (11) Correct the position vector components (coordinates) of all
03017 ;      objects if they have over- or underflowed during the calcul
03018 ;      previous steps.
03019 ;
03020 ; (12) Calculate the perspective projection of the position vector
03021 ;      objects and from that their pixel row and column number (ap
03022 ;      and Aft view) using subroutines PROJECTION ($AA21), SCREENC
03023 ;      and SCREENROW ($B71E). If a space object (star, explosion f
03024 ;      offscreen then a new space object is automatically created
03025 ;      SCREENCOLUMN ($B6FB).
03026 ;
03027 ; (13) Handle hyperwarp marker selection in the Galactic Chart vie
03028 ;      subroutine SELECTWARP ($B162).
03029 ;
03030 ; (14) If in Long-Range Scan view, compute the pixel column number
03031 ;      row number of all PLAYFIELD space objects (stars, explosion
03032 ;      the plane established by the z and x axis of the 3D coordin
03033 ;      using subroutines SCREENCOLUMN ($B6FB) and SCREENROW ($B71E
03034 ;      starship's shape is drawn using subroutine DRAWLINES ($A76F
03035 ;      Long-Range Scan is OK then PLAYFIELD space object pixel num
03036 ;      computed and drawn. This is skipped if the Long-Range Scan
03037 ;
03038 ; (15) Update all PLAYER shapes, heights, and colors (see detailed
03039 ;      below).
03040 ;
03041 ; (16) Flash a red alert when leaving hyperspace into a sector con
03042 ;      ships by setting appropriate colors to PLAYFIELD2 and BACKG
03043 ;
03044 ; (17) Update the color of all PLAYFIELD space objects (stars, exp
03045 ;      fragments). The color calculation is similar to that of the
03046 ;      calculation in (15). It also computes a range index and use
03047 ;      color lookup table FOURCOLORPIXEL ($BA90). If a star in the
03048 ;      became too distant (z-coordinate < -$F000 (-4096) <KM>) its
03049 ;      re-initialized in subroutine INITPOSVEC ($B764).
03050 ;
03051 ; (18) If in demo mode skip input handling and jump directly to fu
03052 ;      handling (28).
03053 ;
03054 ; (19) Handle keyboard input in subroutine KEYBOARD ($AFFE).
03055 ;
03056 ; (20) Handle joystick input. Store the current joystick direction
03057 ;      ($C8) and JOYSTICKY ($C9).
03058 ;
03059 ; (21) Check if our starship's photon torpedoes have hit a target
03060 ;      COLLISION ($AF3D). This subroutine triggers a game over if
03061 ;      ships have been destroyed.
03062 ;
03063 ; (22) Handle the joystick trigger in subroutine TRIGGER ($AE29).
03064 ;
```

```
03065 ; (23) Handle the Attack Computer and Tracking Computer. If the At
03066 ;      is neither destroyed nor switched off then execute the foll
03067 ;
03068 ;      o   Update the Attack Computer Display's blip and lock-on m
03069 ;          subroutine UPDATTCOMP ($A7BF) (if in Front view).
03070 ;
03071 ;      o   Update the tracking index of the currently tracked PLAY
03072 ;          object. If a Zylon ship is tracked, then make sure to a
03073 ;          the Zylon ship that launched the last Zylon photon torp
03074 ;          Zylon ship is not alive then track the other Zylon ship
03075 ;
03076 ;      o   If the Tracking Computer is on then switch to the view
03077 ;          tracked PLAYER space object by emulating pressing the '
03078 ;          view) or 'A' (Aft view) key (only if in Front or Aft vi
03079 ;
03080 ; (24) Handle docking at a starbase in subroutine DOCKING ($ACE6).
03081 ;
03082 ; (25) Handle maneuvering both of our starship's photon torpedoes,
03083 ;      Zylon photon torpedo, and the attacking Zylon ships in subr
03084 ;      MANEUVER ($AA79). This subroutine also automatically create
03085 ;      new Zylon ships.
03086 ;
03087 ; (26) Check if our starship was hit by a Zylon photon torpedo (sk
03088 ;      a starbase sector): Its x, y, and z coordinates must be wit
03089 ;      -($0100)..+$00FF (-256..+255) <KM> of our starship.
03090 ;
03091 ; (27) If our starship was hit then execute the following steps:
03092 ;
03093 ;      o   Damage or destroy one of our starship's subsystems in s
03094 ;          DAMAGE ($AEE1).
03095 ;
03096 ;      o   Trigger an explosion in subroutine INITEXPL ($AC6B),
03097 ;
03098 ;      o   Store the severity of the hit.
03099 ;
03100 ;      o   End the lifetime of the Zylon photon torpedo.
03101 ;
03102 ;      o   Subtract 100 energy units for being hit by the Zylon ph
03103 ;          in subroutine DECENERGY ($B86F).
03104 ;
03105 ;      o   Trigger the noise sound pattern SHIELD EXPLOSION in sub
03106 ;          ($AEA8).
03107 ;
03108 ;      If the Shields were down during the hit, our starship is de
03109 ;      Execute the following steps:
03110 ;
03111 ;      o   Switch to Front view.
03112 ;
03113 ;      o   Flash the title phrase "SHIP DESTROYED BY ZYLON FIRE".
03114 ;
03115 ;      o   Add the mission bonus to the internal game score in sub
03116 ;          GAMEOVER ($B10A).
03117 ;
03118 ;      o   Hide the Control Panel Display (bottom text window) in
03119 ;          MODDLST ($ADF1).
03120 ;
03121 ;      o   Clear the PLAYFIELD in subroutine CLRPLAYFIELD ($AE0D).
03122 ;
03123 ;      o   Enable the STARSHIP EXPLOSION noise.
```

```
                03124 ;
                03125 ; (28) Handle the function keys START and SELECT. If SELECT has be
                03126 ;      cycle through the next of the 4 mission levels. If either S
                03127 ;      have been pressed, reset the idle counter, then jump to the
                03128 ;      game initialization subroutines INITSTART ($A15E) or INITSE
                03129 ;      respectively.
                03130 ;
                03131 ; (29) Update the Control Panel Display in subroutine UPDPANEL ($B
                03132 ;
                03133 ; (30) Handle hyperwarp in subroutine HYPERWARP ($A89B).
                03134 ;
                03135 ; (31) Update the text in the title line in subroutine UPDTITLE ($
                03136 ;
                03137 ; (32) Move Zylon units, decrease lifetime of photon torpedoes, el
                03138 ;      time, etc. in subroutine FLUSHGAMELOOP ($B4E4). This subrou
                03139 ;      triggers a game over if our starship's energy is zero.
                03140 ;
                03141 ; (33) Jump back to the start of the game loop for the next game l
                03142
    =006A       03143 L.HEIGHTCNT     = $6A                ; Height counter during co
    =006E       03144 L.ZPOSOFF       = $6E                ; Offset to z-coordinate
    =006B       03145 L.VELOCITYHI    = $6B                ; Velocity vector componen
    =006A       03146 L.VECCOMPIND    = $6A                ; Position vector componen
                03147                                      ;   0 -> z-component
                03148                                      ;   1 -> x-component
                03149                                      ;   2 -> y-component
    =006A       03150 L.RANGEINDEX    = $6A                ; Range index for space ob
                03151                                      ; distance to our starship
                03152                                      ; cell index of the PLAYER
                03153                                      ; height. Used values are:
    =006A       03154 L.FOURCOLORPIX  = $6A                ; 1-byte bit pattern for 4
    =006B       03155 L.COLORMASK     = $6B                ; Color/brightness to modi
                03156
                03157 ;*** (1) Synchronize game loop with execution of VBI ************
A1F3 A567       03158 GAMELOOP        LDA ISVBISYNC        ; Wait for execution of VB
A1F5 F0FC       03159                 BEQ GAMELOOP         ;
                03160
A1F7 A900       03161                 LDA #0               ; VBI is executed, clear V
A1F9 8567       03162                 STA ISVBISYNC        ;
                03163
                03164 ;*** (2) Erase PLAYFIELD space objects (stars, explosion fragments
A1FB A57A       03165                 LDA OLDMAXSPCOBJIND  ; Skip if no space objects
A1FD F020       03166                 BEQ SKIP002          ;
                03167
A1FF A204       03168                 LDX #NUMSPCOBJ.PL-1  ; Loop over all PLAYFIELD
A201 E8         03169 LOOP002         INX                  ;
A202 BC5B0C     03170                 LDY PIXELROW,X       ; Load pixel row number of
                03171
A205 B90008     03172                 LDA PFMEMROWLO,Y     ; Point MEMPTR to start of
A208 8568       03173                 STA MEMPTR           ; ...in PLAYFIELD memory
A20A B96408     03174                 LDA PFMEMROWHI,Y     ;
A20D 8569       03175                 STA MEMPTR+1         ;
                03176
A20F BC8C0C     03177                 LDY PIXELBYTEOFF,X   ; Get within-row-offset to
A212 BDBD0C     03178                 LDA PIXELSAVE,X      ; Load saved byte
A215 9168       03179                 STA (MEMPTR),Y       ; Restore byte of PLAYFIEL
                03180
A217 E47A       03181                 CPX OLDMAXSPCOBJIND  ;
A219 90E6       03182                 BCC LOOP002          ; Next PLAYFIELD space obj
```

```
                03183
A21B A900       03184                   LDA #0                  ; Clear number of space ob
A21D 857A       03185                   STA OLDMAXSPCOBJIND     ;
                03186
                03187 ;*** (3) Draw PLAYFIELD space objects (stars, explosion fragments)
A21F A5C0       03188 SKIP002           LDA WARPSTATE           ; Skip during hyperspace
A221 302D       03189                   BMI SKIP003             ;
                03190
A223 A679       03191                   LDX MAXSPCOBJIND        ; Update number of space o
A225 867A       03192                   STX OLDMAXSPCOBJIND     ;
                03193
A227 BDF90B     03194 LOOP003           LDA PIXELROWNEW,X       ; Loop over all PLAYFIELD
A22A 9D5B0C     03195                   STA PIXELROW,X          ; Update pixel row number
                03196
A22D A8         03197                   TAY                     ;
A22E B90008     03198                   LDA PFMEMROWLO,Y        ; Point MEMPTR to start of
A231 8568       03199                   STA MEMPTR              ; ...in PLAYFIELD memory
A233 B96408     03200                   LDA PFMEMROWHI,Y        ;
A236 8569       03201                   STA MEMPTR+1            ;
                03202
A238 BD2A0C     03203                   LDA PIXELCOLUMN,X       ; Convert pixel column num
A23B 4A         03204                   LSR A                   ; ...of byte with space ob
A23C 4A         03205                   LSR A                   ;
A23D 9D8C0C     03206                   STA PIXELBYTEOFF,X      ; Store within-row-offset
                03207
A240 A8         03208                   TAY                     ;
A241 B168       03209                   LDA (MEMPTR),Y          ; Load pixel's byte from P
A243 9DBD0C     03210                   STA PIXELSAVE,X         ; Save it (for restoring i
A246 1DEE0C     03211                   ORA PIXELBYTE,X         ; Blend with pixel's color
A249 9168       03212                   STA (MEMPTR),Y          ; Store byte in PLAYFIELD
                03213
A24B CA         03214                   DEX                     ;
A24C E004       03215                   CPX #NUMSPCOBJ.PL-1     ;
A24E D0D7       03216                   BNE LOOP003             ; Next PLAYFIELD space obj
                03217
                03218 ;*** (4) Clear PLAYFIELD center if idle counter is up (?) ********
                03219                                           ; PLAYFIELD addresses of..
    =17BB       03220 PFMEM.C76R49      = PFMEM+49*40+76/4      ; ...pixel column number 7
    =17BC       03221 PFMEM.C80R49      = PFMEM+49*40+80/4      ; ...pixel column number 8
    =17E3       03222 PFMEM.C76R50      = PFMEM+50*40+76/4      ; ...pixel column number 7
    =17E4       03223 PFMEM.C80R50      = PFMEM+50*40+80/4      ; ...pixel column number 8
                03224
A250 A566       03225 SKIP003           LDA IDLECNTHI           ; Skip if idle counter not
A252 100E       03226                   BPL SKIP004             ;
                03227
A254 A900       03228                   LDA #0                  ; Clear pixels of 8 x 2 pi
A256 8DE317     03229                   STA PFMEM.C76R50         ; ...@ column number 76, r
A259 8DE417     03230                   STA PFMEM.C80R50        ;
A25C 8DBC17     03231                   STA PFMEM.C80R49        ;
A25F 8DBB17     03232                   STA PFMEM.C76R49        ;
                03233
                03234 ;*** (5) Clear all PLAYER shapes **********************************
A262 A900       03235 SKIP004           LDA #0                  ; Clear shape of PLAYER4
A264 AC5F0C     03236                   LDY PL4ROW              ;
A267 AEC10C     03237                   LDX PL4HEIGHT           ;
A26A 990003     03238 LOOP004           STA PL4DATA,Y           ;
A26D C8         03239                   INY                     ;
A26E CA         03240                   DEX                     ;
A26F 10F9       03241                   BPL LOOP004             ;
```

```
                03242
A271 AC5E0C     03243                   LDY PL3ROW              ; Clear shape of PLAYER3
A274 AEC00C     03244                   LDX PL3HEIGHT           ;
A277 990007     03245 LOOP005           STA PL3DATA,Y           ;
A27A C8         03246                   INY                     ;
A27B CA         03247                   DEX                     ;
A27C 10F9       03248                   BPL LOOP005             ;
                03249
A27E AC5D0C     03250                   LDY PL2ROW              ; Clear shape of PLAYER2
A281 AEBF0C     03251                   LDX PL2HEIGHT           ;
A284 990006     03252 LOOP006           STA PL2DATA,Y           ;
A287 C8         03253                   INY                     ;
A288 CA         03254                   DEX                     ;
A289 10F9       03255                   BPL LOOP006             ;
                03256
A28B AC5C0C     03257                   LDY PL1ROW              ; Clear shape of PLAYER1
A28E AEBE0C     03258                   LDX PL1HEIGHT           ;
A291 990005     03259 LOOP007           STA PL1DATA,Y           ;
A294 C8         03260                   INY                     ;
A295 CA         03261                   DEX                     ;
A296 10F9       03262                   BPL LOOP007             ;
                03263
A298 AC5B0C     03264                   LDY PL0ROW              ; Clear shape of PLAYER0
A29B AEBD0C     03265                   LDX PL0HEIGHT           ;
A29E 990004     03266 LOOP008           STA PL0DATA,Y           ;
A2A1 C8         03267                   INY                     ;
A2A2 CA         03268                   DEX                     ;
A2A3 10F9       03269                   BPL LOOP008             ;
                03270
                03271 ;*** (6) Update PLAYER vertical positions and update PLAYER shapes
A2A5 AD900C     03272                   LDA PL4SHAPTYPE         ; CARRY := PLAYER4 a PHOTO
A2A8 C901       03273                   CMP #1                  ;
A2AA A4E8       03274                   LDY PL4SHAPOFF          ; Load PLAYER4 shape data
                03275
A2AC AEFD0B     03276                   LDX PL4ROWNEW           ; Update vertical position
A2AF 8E5F0C     03277                   STX PL4ROW              ;
                03278
A2B2 ADF20C     03279                   LDA PL4HEIGHTNEW        ; Update PLAYER4 shape hei
A2B5 856A       03280                   STA L.HEIGHTCNT         ;
A2B7 8DC10C     03281                   STA PL4HEIGHT           ;
                03282
A2BA B9E4B8     03283 LOOP009           LDA PLSHAP1TAB,Y        ; Load PLAYER4 shape byte
A2BD B003       03284                   BCS SKIP005             ; Skip if PLAYER4 not PHOT
A2BF 2D0AD2     03285                   AND RANDOM              ; AND random bits to shape
A2C2 9D0003     03286 SKIP005           STA PL4DATA,X           ; Store shape byte in PLAY
A2C5 C8         03287                   INY                     ;
A2C6 E8         03288                   INX                     ;
A2C7 C66A       03289                   DEC L.HEIGHTCNT         ;
A2C9 10EF       03290                   BPL LOOP009             ; Next row of PLAYER4 shap
                03291
A2CB AD8F0C     03292                   LDA PL3SHAPTYPE         ; Repeat above with PLAYER
A2CE C901       03293                   CMP #1                  ;
A2D0 A4E7       03294                   LDY PL3SHAPOFF          ;
A2D2 AEFC0B     03295                   LDX PL3ROWNEW           ;
A2D5 8E5E0C     03296                   STX PL3ROW              ;
A2D8 ADF10C     03297                   LDA PL3HEIGHTNEW        ;
A2DB 856A       03298                   STA L.HEIGHTCNT         ;
A2DD 8DC00C     03299                   STA PL3HEIGHT           ;
A2E0 B9E4B8     03300 LOOP010           LDA PLSHAP1TAB,Y        ;
```

```
A2E3 B003     03301                BCS SKIP006            ;
A2E5 2D0AD2   03302                AND RANDOM             ;
A2E8 9D0007   03303 SKIP006        STA PL3DATA,X          ;
A2EB E8       03304                INX                    ;
A2EC C8       03305                INY                    ;
A2ED C66A     03306                DEC L.HEIGHTCNT        ;
A2EF 10EF     03307                BPL LOOP010            ;
              03308
A2F1 AD8E0C   03309                LDA PL2SHAPTYPE        ; Repeat above with PLAYER
A2F4 C901     03310                CMP #1                 ;
A2F6 A4E6     03311                LDY PL2SHAPOFF         ;
A2F8 AEFB0B   03312                LDX PL2ROWNEW          ;
A2FB 8E5D0C   03313                STX PL2ROW             ;
A2FE ADF00C   03314                LDA PL2HEIGHTNEW       ;
A301 856A     03315                STA L.HEIGHTCNT        ;
A303 8DBF0C   03316                STA PL2HEIGHT          ;
A306 B9E4B8   03317 LOOP011        LDA PLSHAP1TAB,Y       ;
A309 B003     03318                BCS SKIP007            ;
A30B 2D0AD2   03319                AND RANDOM             ;
A30E 9D0006   03320 SKIP007        STA PL2DATA,X          ;
A311 E8       03321                INX                    ;
A312 C8       03322                INY                    ;
A313 C66A     03323                DEC L.HEIGHTCNT        ;
A315 10EF     03324                BPL LOOP011            ;
              03325
A317 A4E5     03326                LDY PL1SHAPOFF         ; Repeat above with PLAYER
A319 AEFA0B   03327                LDX PL1ROWNEW          ;
A31C 8E5C0C   03328                STX PL1ROW             ;
A31F ADEF0C   03329                LDA PL1HEIGHTNEW       ;
A322 856A     03330                STA L.HEIGHTCNT        ;
A324 8DBE0C   03331                STA PL1HEIGHT          ;
A327 B9B1B9   03332 LOOP012        LDA PLSHAP2TAB,Y       ;
A32A 9D0005   03333                STA PL1DATA,X          ;
A32D E8       03334                INX                    ;
A32E C8       03335                INY                    ;
A32F C66A     03336                DEC L.HEIGHTCNT        ;
A331 10F4     03337                BPL LOOP012            ;
              03338
A333 A4E4     03339                LDY PL0SHAPOFF         ; Repeat above with PLAYER
A335 AEF90B   03340                LDX PL0ROWNEW          ;
A338 8E5B0C   03341                STX PL0ROW             ;
A33B ADEE0C   03342                LDA PL0HEIGHTNEW       ;
A33E 856A     03343                STA L.HEIGHTCNT        ;
A340 8DBD0C   03344                STA PL0HEIGHT          ;
A343 B9B1B9   03345 LOOP013        LDA PLSHAP2TAB,Y       ;
A346 9D0004   03346                STA PL0DATA,X          ;
A349 E8       03347                INX                    ;
A34A C8       03348                INY                    ;
A34B C66A     03349                DEC L.HEIGHTCNT        ;
A34D 10F4     03350                BPL LOOP013            ;
              03351
              03352 ;*** (7) Update PLAYER horizontal positions **********************
A34F AD2A0C   03353                LDA PL0COLUMN          ; Update horizontal positi
A352 8D00D0   03354                STA HPOSP0             ;
A355 AD2B0C   03355                LDA PL1COLUMN          ; Update horizontal positi
A358 8D01D0   03356                STA HPOSP1             ;
A35B AD2C0C   03357                LDA PL2COLUMN          ; Update horizontal positi
A35E 8D02D0   03358                STA HPOSP2             ;
A361 AD2D0C   03359                LDA PL3COLUMN          ; Update horizontal positi
```

```
A364 8D03D0   03360                 STA HPOSP3              ;
A367 AD2E0C   03361                 LDA PL4COLUMN           ; Update horizontal positi
A36A 8D07D0   03362                 STA HPOSM3              ;
A36D 18       03363                 CLC                     ;
A36E 6902     03364                 ADC #2                  ;
A370 8D06D0   03365                 STA HPOSM2              ;
A373 6902     03366                 ADC #2                  ;
A375 8D05D0   03367                 STA HPOSM1              ;
A378 6902     03368                 ADC #2                  ;
A37A 8D04D0   03369                 STA HPOSM0              ;
              03370
              03371 ;*** (8) Rotate space objects horizontally and vertically ********
A37D 24D0     03372                 BIT SHIPVIEW            ; Skip if in Galactic Char
A37F 303A     03373                 BMI SKIP009             ;
              03374
              03375 ;*** Rotate horizontally ***************************************
A381 A5C8     03376                 LDA JOYSTICKX           ; Skip if joystick centere
A383 F019     03377                 BEQ SKIP008             ;
              03378
A385 856D     03379                 STA JOYSTICKDELTA       ; Save JOYSTICKX (used in
A387 A479     03380                 LDY MAXSPCOBJIND        ; Loop over all space obje
A389 846E     03381 LOOP014         STY L.ZPOSOFF           ; Save offset to z-coordin
A38B 18       03382                 CLC                     ;
              03383
A38C 98       03384                 TYA                     ;
A38D AA       03385                 TAX                     ; X := offset to z-coordin
A38E 6931     03386                 ADC #NUMSPCOBJ.ALL      ;
A390 A8       03387                 TAY                     ; Y := offset to x-coordin
A391 209BB6   03388                 JSR ROTATE              ; Calc new x-coordinate (h
              03389
A394 98       03390                 TYA                     ;
A395 AA       03391                 TAX                     ; X := offset to x-coordin
A396 A46E     03392                 LDY L.ZPOSOFF           ; Y := offset to z-coordin
A398 209BB6   03393                 JSR ROTATE              ; Calc new z-coordinate (h
A39B 88       03394                 DEY                     ;
A39C 10EB     03395                 BPL LOOP014             ; Next space object
              03396
              03397 ;*** Rotate vertically *****************************************
A39E A5C9     03398 SKIP008         LDA JOYSTICKY           ; Skip if joystick centere
A3A0 F019     03399                 BEQ SKIP009             ;
              03400
A3A2 856D     03401                 STA JOYSTICKDELTA       ; Save JOYSTICKY (used in
A3A4 A479     03402                 LDY MAXSPCOBJIND        ; Loop over all space obje
A3A6 846E     03403 LOOP015         STY L.ZPOSOFF           ; Save offset to z-coordin
A3A8 18       03404                 CLC                     ;
              03405
A3A9 98       03406                 TYA                     ;
A3AA AA       03407                 TAX                     ; X := offset to z-coordin
A3AB 6962     03408                 ADC #NUMSPCOBJ.ALL*2    ;
A3AD A8       03409                 TAY                     ; Y := offset to y-coordin
A3AE 209BB6   03410                 JSR ROTATE              ; Calc new y-coordinate (v
              03411
A3B1 98       03412                 TYA                     ;
A3B2 AA       03413                 TAX                     ; X := offset to y-coordin
A3B3 A46E     03414                 LDY L.ZPOSOFF           ; Y := offset to z-coordin
A3B5 209BB6   03415                 JSR ROTATE              ; Calc new z-coordinate (v
A3B8 88       03416                 DEY                     ;
A3B9 10EB     03417                 BPL LOOP015             ; Next space object
              03418
```

```
                03419 ;*** (9) Move all space objects along z-axis (toward our starship)
A3BB A679       03420 SKIP009         LDX MAXSPCOBJIND        ; Loop over all space obje
A3BD E005       03421 LOOP016         CPX #NUMSPCOBJ.PL       ; Skip if PLAYFIELD space
A3BF B005       03422                 BCS SKIP010             ;
                03423
A3C1 BD8C0C     03424                 LDA PL0SHAPTYPE,X       ; Skip if next PLAYER spac
A3C4 F019       03425                 BEQ SKIP011             ;
                03426
A3C6 38         03427 SKIP010         SEC                     ; New z-coordinate := old
A3C7 BDD30A     03428                 LDA ZPOSLO,X            ; ...our starship's veloci
A3CA E570       03429                 SBC VELOCITYLO          ; (signed 24-bit subtracti
A3CC 9DD30A     03430                 STA ZPOSLO,X            ;
A3CF BD400A     03431                 LDA ZPOSHI,X            ;
A3D2 E5C1       03432                 SBC VELOCITYHI          ;
A3D4 9D400A     03433                 STA ZPOSHI,X            ;
A3D7 BDAD09     03434                 LDA ZPOSSIGN,X          ;
A3DA E900       03435                 SBC #0                  ;
A3DC 9DAD09     03436                 STA ZPOSSIGN,X          ;
                03437
A3DF CA         03438 SKIP011         DEX                     ;
A3E0 10DB       03439                 BPL LOOP016             ; Next space object
                03440
                03441 ;*** (10) Add space object's velocity vector to space object's pos
A3E2 A679       03442                 LDX MAXSPCOBJIND        ; Loop over all space obje
A3E4 E010       03443 LOOP017         CPX #NUMSPCOBJ.NORM-1   ; Skip if space object is
A3E6 D002       03444                 BNE SKIP012             ; ...because stars don't m
A3E8 A204       03445                 LDX #4                  ;
                03446
A3EA 8A         03447 SKIP012         TXA                     ;
A3EB A8         03448 LOOP018         TAY                     ; Loop over all 3 coordina
                03449
A3EC A900       03450                 LDA #0                  ; Expand 8-bit velocity ve
A3EE 856B       03451                 STA L.VELOCITYHI        ; ...16-bit velocity (high
A3F0 B9660B     03452                 LDA ZVEL,Y              ; ...16-bit velocity (low
A3F3 1009       03453                 BPL SKIP013             ; Skip if 16-bit velocity
                03454
A3F5 497F       03455                 EOR #$7F                ; 16-bit velocity < 0 (neg
A3F7 18         03456                 CLC                     ; ...calculate two's-compl
A3F8 6901       03457                 ADC #1                  ;
A3FA B002       03458                 BCS SKIP013             ;
A3FC C66B       03459                 DEC L.VELOCITYHI        ;
                03460
A3FE 18         03461 SKIP013         CLC                     ; New coordinate := old co
A3FF 79D30A     03462                 ADC ZPOSLO,Y            ; (signed 24-bit addition)
A402 99D30A     03463                 STA ZPOSLO,Y            ;
A405 B9400A     03464                 LDA ZPOSHI,Y            ;
A408 656B       03465                 ADC L.VELOCITYHI        ;
A40A 99400A     03466                 STA ZPOSHI,Y            ;
A40D B9AD09     03467                 LDA ZPOSSIGN,Y          ;
A410 656B       03468                 ADC L.VELOCITYHI        ;
A412 99AD09     03469                 STA ZPOSSIGN,Y          ;
                03470
A415 98         03471                 TYA                     ;
A416 18         03472                 CLC                     ;
A417 6931       03473                 ADC #NUMSPCOBJ.ALL      ;
A419 C990       03474                 CMP #144                ; (!)
A41B 90CE       03475                 BCC LOOP018             ; Next coordinate
                03476
A41D CA         03477                 DEX                     ;
```

```
A41E 10C4     03478                 BPL LOOP017             ; Next space object
              03479
              03480 ;*** (11) Correct over/underflow of PLAYER space objects' position
A420 A004     03481                 LDY #NUMSPCOBJ.PL-1     ;
A422 98       03482 LOOP019         TYA                     ; Loop over all PLAYER spa
A423 AA       03483                 TAX                     ;
              03484
A424 A902     03485                 LDA #2                  ; Loop over all 3 coordina
A426 856A     03486                 STA L.VECCOMPIND        ;
              03487
A428 BDAD09   03488 LOOP020         LDA ZPOSSIGN,X          ; Load sign of coordinate
A42B C902     03489                 CMP #2                  ;
A42D 9010     03490                 BCC SKIP015             ; Skip if sign = 0 (negati
              03491
A42F 0A       03492                 ASL A                   ; SUMMARY: Space object ou
A430 A900     03493                 LDA #0                  ; If new coordinate > +655
A432 9DAD09   03494                 STA ZPOSSIGN,X          ; ...until new coordinate
A435 B005     03495                 BCS SKIP014             ; If new coordinate < -655
A437 FEAD09   03496                 INC ZPOSSIGN,X          ; ...until new coordinate
A43A 49FF     03497                 EOR #$FF                ;
A43C 9D400A   03498 SKIP014         STA ZPOSHI,X            ;
              03499
A43F 8A       03500 SKIP015         TXA                     ;
A440 18       03501                 CLC                     ;
A441 6931     03502                 ADC #NUMSPCOBJ.ALL      ;
A443 AA       03503                 TAX                     ;
A444 C66A     03504                 DEC L.VECCOMPIND        ;
A446 10E0     03505                 BPL LOOP020             ; Next coordinate
              03506
A448 88       03507                 DEY                     ;
A449 10D7     03508                 BPL LOOP019             ; Next space object
              03509
              03510 ;*** (12) Calc perspective projection of space objects **********
A44B A5D0     03511                 LDA SHIPVIEW            ; Skip if in Long-Range Sc
A44D C902     03512                 CMP #$02                ;
A44F B05C     03513                 BCS SKIP019             ;
              03514
A451 A679     03515                 LDX MAXSPCOBJIND        ; Loop over all space obje
A453 A9FF     03516 LOOP021         LDA #255                ; Prep magic offscreen pix
A455 BCAD09   03517                 LDY ZPOSSIGN,X          ; Compare sign of z-coordi
A458 C4D0     03518                 CPY SHIPVIEW            ;
A45A F04B     03519                 BEQ SKIP018             ; Equal? Space object is o
              03520
A45C BD0F0A   03521                 LDA YPOSSIGN,X          ; Prepare projection divis
A45F D012     03522                 BNE SKIP016             ; DIVIDEND (16-bit value)
A461 38       03523                 SEC                     ; (used in subroutine PROJ
A462 A900     03524                 LDA #0                  ;
A464 FD350B   03525                 SBC YPOSLO,X            ;
A467 856A     03526                 STA DIVIDEND            ;
A469 A900     03527                 LDA #0                  ;
A46B FDA20A   03528                 SBC YPOSHI,X            ;
A46E 856B     03529                 STA DIVIDEND+1          ;
A470 4C7DA4   03530                 JMP JUMP001             ;
A473 BD350B   03531 SKIP016         LDA YPOSLO,X            ;
A476 856A     03532                 STA DIVIDEND            ;
A478 BDA20A   03533                 LDA YPOSHI,X            ;
A47B 856B     03534                 STA DIVIDEND+1          ;
              03535
A47D 2021AA   03536 JUMP001         JSR PROJECTION          ; Calc pixel row number re
```

```
A480 201EB7   03537                 JSR SCREENROW           ; Calc pixel row number re
              03538
A483 BDDE09   03539                 LDA XPOSSIGN,X          ; Prepare projection divis
A486 D012     03540                 BNE SKIP017             ; DIVIDEND (16-bit value)
A488 38       03541                 SEC                     ; (used in subroutine PROJ
A489 A900     03542                 LDA #0                  ;
A48B FD040B   03543                 SBC XPOSLO,X            ;
A48E 856A     03544                 STA DIVIDEND            ;
A490 A900     03545                 LDA #0                  ;
A492 FD710A   03546                 SBC XPOSHI,X            ;
A495 856B     03547                 STA DIVIDEND+1          ;
A497 4CA4A4   03548                 JMP JUMP002             ;
A49A BD040B   03549 SKIP017         LDA XPOSLO,X            ;
A49D 856A     03550                 STA DIVIDEND            ;
A49F BD710A   03551                 LDA XPOSHI,X            ;
A4A2 856B     03552                 STA DIVIDEND+1          ;
              03553
A4A4 2021AA   03554 JUMP002         JSR PROJECTION          ; Calc pixel column number
A4A7 20FBB6   03555 SKIP018         JSR SCREENCOLUMN        ; Calc pixel column number
A4AA CA       03556                 DEX                     ;
A4AB 10A6     03557                 BPL LOOP021             ; Next space object
              03558
              03559 ;*** (13) Handle hyperwarp marker selection in Galactic Chart view
A4AD 2062B1   03560 SKIP019         JSR SELECTWARP          ; Handle hyperwarp marker
              03561
              03562 ;*** (14) Compute and draw Long-Range Scan view star field on z-x
A4B0 24D0     03563                 BIT SHIPVIEW            ; Skip if not in Long-Rang
A4B2 5031     03564                 BVC SKIP022             ;
              03565
A4B4 A231     03566                 LDX #$31                ; Draw our starship's shap
A4B6 206FA7   03567                 JSR DRAWLINES           ;
              03568
A4B9 2C9609   03569                 BIT GCSTATLRS           ; Skip if Long-Range Scan
A4BC 7027     03570                 BVS SKIP022             ;
              03571
A4BE A679     03572                 LDX MAXSPCOBJIND        ; Loop over all space obje
A4C0 BD400A   03573 LOOP022         LDA ZPOSHI,X            ; Load z-coordinate (high
A4C3 BCAD09   03574                 LDY ZPOSSIGN,X          ; Load sign of z-coordinat
A4C6 D002     03575                 BNE SKIP020             ;
A4C8 49FF     03576                 EOR #$FF                ; A := ABS(z-coordinate (h
A4CA A8       03577 SKIP020         TAY                     ;
A4CB B9E90D   03578                 LDA MAPTO80,Y           ; Calc pixel row number re
A4CE 201EB7   03579                 JSR SCREENROW           ; Calc pixel row number re
              03580
A4D1 BD710A   03581                 LDA XPOSHI,X            ; Load x-coordinate (high
A4D4 BCDE09   03582                 LDY XPOSSIGN,X          ; Load sign of x-coordinat
A4D7 D002     03583                 BNE SKIP021             ;
A4D9 49FF     03584                 EOR #$FF                ; A := ABS(x-coordinate (h
A4DB A8       03585 SKIP021         TAY                     ;
A4DC B9E90D   03586                 LDA MAPTO80,Y           ; Calc pixel column number
A4DF 20FBB6   03587                 JSR SCREENCOLUMN        ; Calc pixel column number
              03588
A4E2 CA       03589                 DEX                     ;
A4E3 10DB     03590                 BPL LOOP022             ; Next space object
              03591
              03592 ;*** (15) Update PLAYER shapes, heights, and colors **************
              03593
              03594 ; DESCRIPTION
              03595 ;
```

```
03596 ; In a loop over all PLAYERs, the following steps are executed:
03597 ;
03598 ; o   Clear the PLAYER shape offset and height.
03599 ;
03600 ; o   If in Galactic Chart view or in Long-Range Scan view, preloa
03601 ;     color and a magic z-coordinate (distance value) for PLAYER3.
03602 ;     (representing hyperwarp markers in Galactic Chart view and b
03603 ;     Long-Range Scan view, like, for example, Zylon ships, meteor
03604 ;     the Hyperwarp Target Marker during hyperwarp!).
03605 ;
03606 ; o   If in Front or Aft view, execute the following steps:
03607 ;
03608 ;      o   Skip dead PLAYERs.
03609 ;
03610 ;      o   Preload the distance value for the remaining live PLAYE
03611 ;
03612 ;      o   If we are in a starbase sector, combine PLAYER0..2 into
03613 ;          starbase shape. Compute the pixel column numbers and pi
03614 ;          numbers of PLAYER0..1 such that they are arranged left
03615 ;          right (PLAYER1) of PLAYER2. In addition, preload a colo
03616 ;          counter actually, that will make the starbase pulsate i
03617 ;
03618 ;     BUG (at $A512): The code at $A512 that skips the combination
03619 ;     PLAYER2..4 jumps for PLAYER3..4 to SKIP025 at $A52A instead
03620 ;     $A52E. Thus it stores a color mask which does not only make
03621 ;     PLAYER0..2 pulsate in brightness but also PLAYER3..4 in a st
03622 ;     for example the transfer vessel, photon torpedoes, etc. - or
03623 ;     Hyperwarp Target Marker when hyperwarping out of such a sect
03624 ;     fix: None, code hard to untwist.
03625 ;
03626 ; o   After storing the color mask, check if the PLAYER shape is s
03627 ;     bottom edge of the PLAYFIELD.
03628 ;
03629 ;     BUG (at $A534): The test checks the vertical position of the
03630 ;     the PLAYER against the bottom edge of the PLAYFIELD above th
03631 ;     Panel Display (= Player/Missile pixel row number 204). This
03632 ;     completely accurate as the Console Panel Display starts at P
03633 ;     number 208. For example, if you carefully navigate a starbas
03634 ;     bottom edge of the PLAYFIELD, at a certain point the center
03635 ;     starbase shape bleeds over the bottom edge of the PLAYFIELD
03636 ;     sometimes even losing its left and right wings!). Suggested
03637 ;     a more elaborate test may consume too many bytes of the cart
03638 ;     memory in order to fix a rarely noticed visual glitch.
03639 ;
03640 ; o   Convert the preloaded distance value of a PLAYER space objec
03641 ;     $2000 (8192) <KM> into a range index of 0..15. PLAYER space
03642 ;     distant than $2000 (8192) <KM> are skipped and not displayed
03643 ;
03644 ;     Later, this range index will pick not only the correct brigh
03645 ;     PLAYER (the closer the space object the brighter its PLAYER)
03646 ;     correct PLAYER shape cell and height (the closer the space o
03647 ;     larger the PLAYER shape and height).
03648 ;
03649 ; o   Update the PLAYER's shape offset and height. On the way to t
03650 ;     offset and height add the PLAYER's shape type to the range i
03651 ;     divide it by 2 to arrive at the shape offset index and heigh
03652 ;     same value). Use this index to pick the correct shape data a
03653 ;     heights from a set of shape cells and their corresponding he
03654 ;     in tables PLSHAPOFFTAB ($BE2F) and PLSHAPHEIGHTTAB ($BE7F),
```

```
                03655 ;
                03656 ;     Remember that magic distance value used in the Galactic Char
                03657 ;     Long-Range Scan view? Its value of $F2 is actually part of a
                03658 ;     z-coordinate which is inverted to $0D00, leading to a range
                03659 ;     which, after the division by 2, picks shape cell 6. Shape ce
                03660 ;     seventh shape cell) of all space objects (except the starbas
                03661 ;     Long-Range Scan blip's dot (see PLSHAPOFFTAB ($BE2F) and PLS
                03662 ;     ($BE7F)).
                03663 ;
                03664 ; o   Update the PLAYER's color/brightness by picking the appropri
                03665 ;     with the range index from lookup tables PLSHAPCOLORTAB ($BFD
                03666 ;     PLSHAPBRITTAB ($BFDB). Apply some special effects to the col
                03667 ;     of certain PLAYERs, such as using random colors for Zylon ba
                03668 ;     using the precomputed pulsating brightness value for a starb
                03669
A4E5 A205       03670 SKIP022         LDX #NUMSPCOBJ.PL       ; Loop over all PLAYER spa
A4E7 CA         03671 LOOP023         DEX                     ;
A4E8 1003       03672                 BPL SKIP023             ; Jump into loop body belo
A4EA 4C79A5     03673                 JMP JUMP003             ; Loop is finished, skip l
                03674
                03675 ;*** Clear PLAYER shape offsets and heights **********************
A4ED A900       03676 SKIP023         LDA #0                  ;
A4EF 95E4       03677                 STA PL0SHAPOFF,X        ; Clear PLAYER shape offse
A4F1 9DEE0C     03678                 STA PL0HEIGHTNEW,X      ; Clear new PLAYER shape h
                03679
                03680 ;*** Preload stuff for hyperwarp markers and Long-Range Scan blips
A4F4 24D0       03681                 BIT SHIPVIEW            ; Skip if not in Galactic
A4F6 100B       03682                 BPL SKIP024             ;
                03683
A4F8 E003       03684                 CPX #3                  ; Next PLAYER space object
A4FA 90EB       03685                 BCC LOOP023             ;
                03686
A4FC AD0AD2     03687 LOOP024         LDA RANDOM              ; Prep random color mask f
A4FF A0F2       03688                 LDY #$F2                ; Prep magic z-coordinate
A501 302B       03689                 BMI SKIP026             ; Unconditional jump
                03690
A503 D5E9       03691 SKIP024         CMP PL0LIFE,X           ; Next PLAYER space object
A505 F0E0       03692                 BEQ LOOP023             ;
                03693
A507 70F3       03694                 BVS LOOP024             ; Skip back if in Long-Ran
                03695
                03696 ;*** Preload stuff for other views ******************************
                03697
A509 BC400A     03698                 LDY PL0ZPOSHI,X         ; Prep z-coordinate (high
                03699
                03700 ;*** Combine PLAYER0..2 to starbase shape ***********************
A50C 247B       03701                 BIT ISSTARBASESECT      ; Skip if no starbase in t
A50E 501E       03702                 BVC SKIP026             ;
                03703
A510 E002       03704                 CPX #2                  ; Skip if PLAYER2..4
A512 B016       03705                 BCS SKIP025             ; (!)
                03706
A514 AD2C0C     03707                 LDA PL2COLUMN           ; Calc new PM pixel column
A517 18         03708                 CLC                     ; Load PLAYER2 (starbase c
A518 7DDBBE     03709                 ADC PLSTARBAOFFTAB,X    ; ...add PLAYER left/right
A51B 9D2A0C     03710                 STA PL0COLUMN,X         ; Store new PM pixel colum
                03711
A51E ADFB0B     03712                 LDA PL2ROWNEW           ; Calc new PM pixel row nu
A521 18         03713                 CLC                     ; Add vertical offset (= 4
```

```
A522 6904     03714                 ADC #4                ;
A524 9DF90B   03715                 STA PL0ROWNEW,X       ; Store new PM pixel row n
              03716
A527 AC420A   03717                 LDY PL2ZPOSHI         ; Prep Y with z-coordinate
              03718
A52A A576     03719 SKIP025         LDA COUNT256          ; Prep color mask with B3.
A52C 290F     03720                 AND #$0F              ; ...(= brightness bits ca
              03721
A52E 856B     03722 SKIP026         STA L.COLORMASK       ; Store color mask
              03723
              03724 ;*** Check if PLAYER is below PLAYFIELD bottom edge **************
A530 98       03725                 TYA                   ; A := z-coordinate (high
              03726
A531 BCF90B   03727                 LDY PL0ROWNEW,X       ; Next PLAYER space object
A534 C0CC     03728                 CPY #204              ; ...is below PLAYFIELD bo
A536 B0AF     03729                 BCS LOOP023           ; ...(PM pixel row number
              03730
              03731 ;*** Convert PLAYER z-coordinate to range index in 0..15 *********
A538 A4D0     03732                 LDY SHIPVIEW          ; Skip if in Front view...
A53A F002     03733                 BEQ SKIP027           ;
A53C 49FF     03734                 EOR #$FF              ; ...else invert z-coordin
              03735
A53E C920     03736 SKIP027         CMP #$20              ; Next PLAYER space object
A540 B0A5     03737                 BCS LOOP023           ; ...(z-coordinate >= $20*
              03738
A542 C910     03739                 CMP #16               ; Load z-coordinate (high
A544 9002     03740                 BCC SKIP028           ;
A546 A90F     03741                 LDA #15               ;
A548 856A     03742 SKIP028         STA L.RANGEINDEX      ; ...trim to range index i
              03743
              03744 ;*** Update PLAYER shape offset and height **********************
A54A 1D8C0C   03745                 ORA PL0SHAPTYPE,X     ; Calc offset to shape tab
A54D 4A       03746                 LSR A                 ;
A54E A8       03747                 TAY                   ; Divide by 2 to get offse
A54F B92FBE   03748                 LDA PLSHAPOFFTAB,Y    ; Update new PLAYER shape
A552 95E4     03749                 STA PL0SHAPOFF,X      ;
A554 B97FBE   03750                 LDA PLSHAPHEIGHTTAB,Y ; Update new PLAYER shape
A557 9DEE0C   03751                 STA PL0HEIGHTNEW,X    ;
              03752
              03753 ;*** Calculate PLAYER color/brightness value ********************
A55A 98       03754                 TYA                   ; Pick color (B7..4) using
A55B 4A       03755                 LSR A                 ;
A55C 4A       03756                 LSR A                 ;
A55D 4A       03757                 LSR A                 ;
A55E A8       03758                 TAY                   ;
A55F B9D1BF   03759                 LDA PLSHAPCOLORTAB,Y  ;
A562 C008     03760                 CPY #8                ; Pick random color if ZYL
A564 D003     03761                 BNE SKIP029           ;
A566 4D0AD2   03762                 EOR RANDOM            ;
A569 A46A     03763 SKIP029         LDY L.RANGEINDEX      ;
A56B 59DBBF   03764                 EOR PLSHAPBRITTAB,Y   ; Pick brightness (B3..0)
              03765
A56E 456B     03766                 EOR L.COLORMASK       ; Modify color/brightness
              03767
A570 BCDFB8   03768                 LDY PLCOLOROFFTAB,X   ; Get PLAYER color offset
A573 99EE00   03769                 STA PL0COLOR,Y        ; Store color in PLAYER co
A576 4CE7A4   03770                 JMP LOOP023           ; Next PLAYER space object
              03771
              03772 ;*** (16) Flash red alert ***************************************
```

```
A579 A0AF     03773 JUMP003         LDY #$AF                ; Prep PLAYFIELD2 color {B
A57B A681     03774                 LDX SHIELDSCOLOR        ; Prep Shields color {DARK
              03775
A57D A58B     03776                 LDA REDALERTLIFE        ; Skip if red alert is ove
A57F F00C     03777                 BEQ SKIP030             ;
              03778
A581 C68B     03779                 DEC REDALERTLIFE        ; Decrement lifetime of re
A583 A04F     03780                 LDY #$4F                ; Prep PLAYFIELD2 color {B
              03781
A585 2920     03782                 AND #$20                ; Switch colors every 64 g
A587 F004     03783                 BEQ SKIP030             ;
              03784
A589 A242     03785                 LDX #$42                ; Load BACKGROUND color {D
A58B A060     03786                 LDY #$60                ; Load PLAYFIELD2 color {D
              03787
A58D 84F4     03788 SKIP030         STY PF2COLOR            ; Store PLAYFIELD2 color
A58F 86F6     03789                 STX BGRCOLOR            ; Store BACKGROUND color
              03790
              03791 ;*** (17) Update color of PLAYFIELD space objects (stars, explosio
A591 A679     03792                 LDX MAXSPCOBJIND        ; Loop over all PLAYFIELD
A593 BD400A   03793 LOOP025         LDA ZPOSHI,X            ; Prep z-coordinate (high
A596 A4D0     03794                 LDY SHIPVIEW            ;
A598 C001     03795                 CPY #1                  ; Skip if not in Aft view
A59A D009     03796                 BNE SKIP032             ;
              03797
A59C C9F0     03798                 CMP #$F0                ; Skip if star not too far
A59E B003     03799                 BCS SKIP031             ;
A5A0 2064B7   03800                 JSR INITPOSVEC          ; Re-init position vector
A5A3 49FF     03801 SKIP031         EOR #$FF                ; Invert z-coordinate (hig
              03802
A5A5 C910     03803 SKIP032         CMP #16                 ; Convert z-coordinate (hi
A5A7 9002     03804                 BCC SKIP033             ; ...into range index 0..1
A5A9 A90F     03805                 LDA #15                 ;
              03806
A5AB 0A       03807 SKIP033         ASL A                   ; Compute index to pixel c
A5AC 291C     03808                 AND #$1C                ; Use bits B3..1 from rang
A5AE 0572     03809                 ORA COUNT8              ; Combine with random bits
              03810
A5B0 A8       03811                 TAY                     ;
A5B1 B990BA   03812                 LDA FOURCOLORPIXEL,Y    ; Load 1-byte bit pattern
A5B4 856A     03813                 STA L.FOURCOLORPIX      ; ...and temporarily save
              03814
A5B6 BD2A0C   03815                 LDA PIXELCOLUMN,X       ; Load pixel mask to mask
A5B9 2903     03816                 AND #$03                ; Use B1..0 from pixel col
A5BB A8       03817                 TAY                     ;
A5BC B9B0BA   03818                 LDA PIXELMASKTAB,Y      ; ...to pick mask to filte
A5BF 256A     03819                 AND L.FOURCOLORPIX      ; ...AND with 1-byte bit p
A5C1 9DEE0C   03820                 STA PIXELBYTE,X         ; ...store byte (used in r
              03821
A5C4 CA       03822                 DEX                     ;
A5C5 E005     03823                 CPX #NUMSPCOBJ.PL       ;
A5C7 B0CA     03824                 BCS LOOP025             ; Next PLAYFIELD space obj
              03825
              03826 ;*** (18) Skip input handling if in demo mode ******************
A5C9 2464     03827                 BIT ISDEMOMODE          ; If in demo mode skip to
A5CB 5003     03828                 BVC SKIP034             ;
A5CD 4C9BA6   03829                 JMP SKIP040             ;
              03830
              03831 ;*** (19) Handle keyboard input ********************************
```

```
A5D0 20FEAF   03832 SKIP034         JSR KEYBOARD           ; Handle keyboard input
              03833
              03834 ;*** (20) Handle joystick input ********************************
A5D3 AD00D3   03835                 LDA PORTA              ; Load Joystick 0 directio
A5D6 A8       03836                 TAY                    ; ...Bits B0..3 -> Right,
A5D7 2903     03837                 AND #$03               ; ...Bit = 0/1 -> Stick pr
A5D9 AA       03838                 TAX                    ; JOYSTICKY := +1 -> Up
A5DA BDF5BA   03839                 LDA STICKINCTAB,X      ; JOYSTICKY :=  0 -> Cente
A5DD 85C9     03840                 STA JOYSTICKY          ; JOYSTICKY := -1 -> Down
A5DF 98       03841                 TYA                    ;
A5E0 4A       03842                 LSR A                  ;
A5E1 4A       03843                 LSR A                  ;
A5E2 2903     03844                 AND #$03               ;
A5E4 AA       03845                 TAX                    ; JOYSTICKX := -1 -> Left
A5E5 BDF5BA   03846                 LDA STICKINCTAB,X      ; JOYSTICKX :=  0 -> Cente
A5E8 85C8     03847                 STA JOYSTICKX          ; JOYSTICKX := +1 -> Right
              03848
              03849 ;*** (21) Check if our starship's photon torpedoes have hit a targ
A5EA 203DAF   03850                 JSR COLLISION          ; Check if our starship's
              03851
              03852 ;*** (22) Handle joystick trigger ******************************
A5ED 2029AE   03853                 JSR TRIGGER            ; Handle joystick trigger
              03854
              03855 ;*** (23) Handle Attack Computer and Tracking Computer ***********
A5F0 2C9509   03856                 BIT GCSTATCOM          ; Skip if Attack Computer
A5F3 7040     03857                 BVS SKIP038            ;
              03858
A5F5 A57E     03859                 LDA DRAINATTCOMP       ; Skip if Attack Computer
A5F7 F03C     03860                 BEQ SKIP038            ;
              03861
A5F9 A5D0     03862                 LDA SHIPVIEW           ; Skip if not in Front vie
A5FB D003     03863                 BNE SKIP035            ;
              03864
A5FD 20BFA7   03865                 JSR UPDATTCOMP         ; Update Attack Computer D
              03866
A600 AE5C09   03867 SKIP035         LDX TRACKDIGIT         ; Load index of tracked sp
              03868
A603 A5BF     03869                 LDA ZYLONATTACKER      ; Skip if ship of current
A605 3005     03870                 BMI SKIP036            ;
A607 AA       03871                 TAX                    ; ...else override Trackin
A608 0980     03872                 ORA #$80               ;
A60A 85BF     03873                 STA ZYLONATTACKER      ; ...and mark Zylon torped
              03874
A60C B5E9     03875 SKIP036         LDA PL0LIFE,X          ; Skip if tracked space ob
A60E D00B     03876                 BNE SKIP037            ;
              03877
A610 8A       03878                 TXA                    ;
A611 4901     03879                 EOR #$01               ;
A613 AA       03880                 TAX                    ;
A614 B5E9     03881                 LDA PL0LIFE,X          ; Check if other Zylon shi
A616 D003     03882                 BNE SKIP037            ; ...yes -> Keep new index
A618 AE5C09   03883                 LDX TRACKDIGIT         ; ...no  -> Revert to old
              03884
A61B 8E5C09   03885 SKIP037         STX TRACKDIGIT         ; Store index of tracked s
              03886
A61E A57C     03887                 LDA ISTRACKCOMPON      ; Skip if tracking compute
A620 F013     03888                 BEQ SKIP038            ;
              03889
A622 A5D0     03890                 LDA SHIPVIEW           ; Skip if in Long-Range Sc
```

```
A624 C902     03891                 CMP #2                ;
A626 B00D     03892                 BCS SKIP038           ;
              03893
A628 4901     03894                 EOR #$01              ;
A62A DDAD09   03895                 CMP ZPOSSIGN,X        ; Skip if tracked space ob
A62D F006     03896                 BEQ SKIP038           ; ...view direction
              03897
A62F AA       03898                 TAX                   ;
A630 BDCFBE   03899                 LDA TRACKKEYSTAB,X    ; Pick 'F' or 'A' (Front o
A633 85CA     03900                 STA KEYCODE           ; ...and store it (= emula
              03901
              03902 ;*** (24) Handle docking to starbase ****************************
A635 20E6AC   03903 SKIP038         JSR DOCKING           ; Handle docking to starba
              03904
              03905 ;*** (25) Handle maneuvering ************************************
A638 2079AA   03906                 JSR MANEUVER          ; Handle maneuvering photo
              03907
              03908 ;*** (26) Was our starship hit by Zylon photon torpedo? **********
A63B A57B     03909                 LDA ISSTARBASESECT    ; Skip hit check if in sta
A63D D05C     03910                 BNE SKIP040           ;
              03911
A63F A5EB     03912                 LDA PL2LIFE           ; Skip hit check if PLAYER
A641 F058     03913                 BEQ SKIP040           ; ...not alive
              03914
A643 AC420A   03915                 LDY PL2ZPOSHI         ; Our starship was not hit
A646 C8       03916                 INY                   ; ...z-coordinate is not i
A647 C002     03917                 CPY #$02              ;
A649 B050     03918                 BCS SKIP040           ;
              03919
A64B AC730A   03920                 LDY PL2XPOSHI         ; ...x-coordinate is not i
A64E C8       03921                 INY                   ;
A64F C002     03922                 CPY #$02              ;
A651 B048     03923                 BCS SKIP040           ;
              03924
A653 ACA40A   03925                 LDY PL2YPOSHI         ; ...y-coordinate is not i
A656 C8       03926                 INY                   ;
A657 C002     03927                 CPY #$02              ;
A659 B040     03928                 BCS SKIP040           ;
              03929
              03930 ;*** (27) Our starship was hit! **********************************
A65B 20E1AE   03931                 JSR DAMAGE            ; Damage or destroy some s
              03932
A65E A002     03933                 LDY #2                ; Trigger explosion at PLA
A660 206BAC   03934                 JSR INITEXPL          ;
              03935
A663 A27F     03936                 LDX #$7F              ; Prep HITBADNESS := SHIEL
A665 A581     03937                 LDA SHIELDSCOLOR      ; Skip if Shields are up (
A667 D01E     03938                 BNE SKIP039           ;
              03939
A669 A20A     03940                 LDX #$0A              ; Set Front view
A66B 2045B0   03941                 JSR SETVIEW           ;
              03942
A66E A023     03943                 LDY #$23              ; Set title phrase "SHIP D
A670 A208     03944                 LDX #8                ; Set mission bonus offset
A672 200AB1   03945                 JSR GAMEOVER          ; Game over
              03946
A675 A25F     03947                 LDX #$5F              ; Hide Control Panel Displ
A677 A080     03948                 LDY #$80              ;
A679 A908     03949                 LDA #$08              ;
```

```
A67B 20F1AD   03950                 JSR MODDLST             ;
              03951
A67E 200DAE   03952                 JSR CLRPLAYFIELD        ; Clear PLAYFIELD
              03953
A681 A240     03954                 LDX #64                 ; Enable STARSHIP EXPLOSIO
A683 86E3     03955                 STX NOISEHITLIFE        ;
              03956
A685 A2FF     03957                 LDX #$FF                ; Prep HITBADNESS := STARS
              03958
A687 868A     03959 SKIP039         STX HITBADNESS          ; Store HITBADNESS
A689 A900     03960                 LDA #0                  ; Zylon photon torpedo lif
A68B 85EB     03961                 STA PL2LIFE             ;
A68D A902     03962                 LDA #2                  ; Init Zylon photon torped
A68F 85BE     03963                 STA TORPEDODELAY        ;
              03964
A691 A201     03965                 LDX #1                  ; ENERGY := ENERGY - 100 a
A693 206FB8   03966                 JSR DECENERGY           ;
              03967
A696 A20A     03968                 LDX #$0A                ; Play noise sound pattern
A698 20A8AE   03969                 JSR NOISE               ;
              03970
              03971 ;*** (28) Handle function keys **********************************
A69B A463     03972 SKIP040         LDY FKEYCODE            ; Prep old function key co
A69D AD1FD0   03973                 LDA CONSOL              ; POKEY: Load function key
              03974
A6A0 49FF     03975                 EOR #$FF                ; Store inverted and maske
A6A2 2903     03976                 AND #$03                ;
A6A4 8563     03977                 STA FKEYCODE            ;
A6A6 F01A     03978                 BEQ SKIP042             ; Skip if no function key
              03979
A6A8 88       03980                 DEY                     ;
A6A9 1017     03981                 BPL SKIP042             ; Skip if SELECT or START
A6AB 8566     03982                 STA IDLECNTHI           ; Reset idle counter to a
A6AD C902     03983                 CMP #2                  ; Skip if SELECT function
A6AF B006     03984                 BCS SKIP041             ;
              03985
A6B1 A900     03986                 LDA #0                  ; START function key press
A6B3 A8       03987                 TAY                     ; Prep empty title phrase
A6B4 4C5EA1   03988                 JMP INITSTART           ; Reenter game loop via IN
              03989
A6B7 E662     03990 SKIP041         INC MISSIONLEVEL        ; SELECT function key pres
A6B9 A562     03991                 LDA MISSIONLEVEL        ; Cycle through next of 4
A6BB 2903     03992                 AND #$03                ;
A6BD 8562     03993                 STA MISSIONLEVEL        ;
A6BF 4C5AA1   03994                 JMP INITSELECT          ; Reenter game loop via IN
              03995
              03996 ;*** (29) Update Control Panel Display **************************
A6C2 2004B8   03997 SKIP042         JSR UPDPANEL            ; Update Control Panel Dis
              03998
              03999 ;*** (30) Handle hyperwarp **************************************
A6C5 209BA8   04000                 JSR HYPERWARP           ; Handle hyperwarp
              04001
              04002 ;*** (31) Update title line *************************************
A6C8 2016B2   04003                 JSR UPDTITLE            ; Update title line
              04004
              04005 ;*** (32) Flush game loop iteration ******************************
A6CB 20E4B4   04006                 JSR FLUSHGAMELOOP       ; Move Zylon units, age to
              04007
              04008 ;*** (33) Jump back to begin of game loop ************************
```

```
A6CE 4CF3A1   04009                 JMP GAMELOOP           ; Next game loop iteration
              04010
              04011 ;*****************************************************************
              04012 ;*
              04013 ;*                                  VBIHNDLR
              04014 ;*
              04015 ;*                       Vertical Blank Interrupt Handler
              04016 ;*
              04017 ;*****************************************************************
              04018
              04019 ; DESCRIPTION
              04020 ;
              04021 ; This subroutine is executed during the Vertical Blank Interrupt
              04022 ; TV beam has reached the bottom-right corner of the TV screen and
              04023 ; off to return to the top-left position. This situation is called
              04024 ; blank phase".
              04025 ;
              04026 ; This subroutine signals its execution with flag ISVBISYNC ($67)
              04027 ; examined by GAMELOOP ($A1F3) to synchronize the execution of the
              04028 ; with the start of this subroutine). Then it switches the charact
              04029 ; ROM character set, sets the BACKGROUND color depending on the se
              04030 ; Zylon photon torpedo hit and view mode, copies PLAYER and PLAYFI
              04031 ; registers to their corresponding hardware registers, clears the
              04032 ; collision registers, calls the sound effects code in subroutine
              04033 ; and increments the idle counter. If the idle counter reaches the
              04034 ; the title phrase is cleared and the game is switched to demo mod
              04035 ;
              04036 ; BUG (at $A6EC): Because the values of SHIPVIEW ($D0) are $00, $0
              04037 ; $80, a value of 3 overspecifies the comparison. Suggested fix: R
              04038 ; with CMP #2, which may make the code clearer.
              04039 ;
              04040 ; BUG (at $A712): Demo mode is entered via a JMP instruction, whic
              04041 ; directly into GAMELOOP ($A1F3). Thus code execution never return
              04042 ; registers pushed on the stack during entry of this subroutine. S
              04043 ; None.
              04044
A6D1 A9FF     04045 VBIHNDLR        LDA #$FF               ; Signals entering Vertica
A6D3 8567     04046                 STA ISVBISYNC          ;
              04047
A6D5 A9E0     04048                 LDA #>ROMCHARSET       ; Switch character set to
A6D7 8D09D4   04049                 STA CHBASE             ;
              04050
A6DA A6F6     04051                 LDX BGRCOLOR           ; Preload BACKGROUND color
A6DC AD0AD2   04052                 LDA RANDOM             ; Preload random number
A6DF 248A     04053                 BIT HITBADNESS         ; Check if our starship wa
A6E1 5007     04054                 BVC SKIP044            ; If HITBADNESS has a valu
A6E3 3004     04055                 BMI SKIP043            ; $00 -> NO HIT
A6E5 2972     04056                 AND #$72               ; $7F -> SHIELDS HIT
A6E7 0940     04057                 ORA #$40               ; $FF -> STARSHIP DESTROYE
A6E9 AA       04058 SKIP043         TAX                    ;
A6EA A5D0     04059 SKIP044         LDA SHIPVIEW           ; Skip if in Front or Aft
A6EC C903     04060                 CMP #3                 ; (!)
A6EE 9002     04061                 BCC SKIP045            ;
A6F0 A2A0     04062                 LDX #$A0               ; Preload BACKGROUND color
A6F2 86F6     04063 SKIP045         STX BGRCOLOR           ; Store BACKGROUND color
              04064
A6F4 A208     04065                 LDX #8                 ; Copy all color registers
A6F6 B5EE     04066 LOOP026         LDA PL0COLOR,X         ;
A6F8 9D12D0   04067                 STA COLPM0,X           ;
```

```
A6FB CA       04068                 DEX                     ;
A6FC 10F8     04069                 BPL LOOP026             ;
              04070
A6FE 8D1ED0   04071                 STA HITCLR              ; Clear Player/Missile col
              04072
A701 20ABB2   04073                 JSR SOUND               ; Call sound effects
              04074
A704 E677     04075                 INC IDLECNTLO           ; Increment 16-bit idle co
A706 D00D     04076                 BNE SKIP046             ;
A708 A566     04077                 LDA IDLECNTHI           ;
A70A 3009     04078                 BMI SKIP046             ;
A70C E666     04079                 INC IDLECNTHI           ;
A70E 1005     04080                 BPL SKIP046             ; Skip if idle counter val
              04081
A710 A000     04082                 LDY #$00                ; Prep empty title phrase
A712 4C5CA1   04083                 JMP INITDEMO            ; Enter demo mode (!)
              04084
A715 4C4BA7   04085 SKIP046         JMP JUMP004             ; Return via DLI return co
              04086
              04087 ;*****************************************************************
              04088 ;*
              04089 ;*                                  DLSTHNDLR
              04090 ;*
              04091 ;*                       Display List Interrupt Handler
              04092 ;*
              04093 ;*****************************************************************
              04094
              04095 ; DESCRIPTION
              04096 ;
              04097 ; This subroutine is executed during the Display List Interrupt (D
              04098 ; switches the character set to the ROM character set if the DLI o
              04099 ; line 96 (video line 192), otherwise to the custom character set.
              04100 ; happens in the Galactic Chart view where the ROM character set i
              04101 ; Galactic Chart Panel Display.
              04102 ;
              04103 ; Then, the DLI PLAYFIELD colors are copied to the corresponding h
              04104 ; registers and the values of the collision hardware registers for
              04105 ; (our starship's photon torpedoes) are copied to the correspondin
              04106 ; variables PL3HIT ($82) and PL4HIT ($83).
              04107
A718 48       04108 DLSTHNDLR       PHA                     ; Push A
A719 8A       04109                 TXA                     ;
A71A 48       04110                 PHA                     ; Push X
A71B 98       04111                 TYA                     ;
A71C 48       04112                 PHA                     ; Push Y
              04113
A71D A9E0     04114                 LDA #>ROMCHARSET        ; Switch to ROM charset if
A71F AC0BD4   04115                 LDY VCOUNT              ; ...else switch to custom
A722 C060     04116                 CPY #96                 ;
A724 F002     04117                 BEQ SKIP047             ;
A726 A9A0     04118                 LDA #>CHARSET           ;
A728 8D09D4   04119 SKIP047         STA CHBASE              ;
              04120
A72B A204     04121                 LDX #4                  ; Loop over all PLAYFIELD
A72D 8D0AD4   04122                 STA WSYNC               ; Stop and wait for horizo
A730 B5F7     04123 LOOP027         LDA PF0COLORDLI,X       ; Copy DLI PLAYFIELD color
A732 9D16D0   04124                 STA COLPF0,X            ;
A735 CA       04125                 DEX                     ;
A736 10F8     04126                 BPL LOOP027             ; Next PLAYFIELD color
```

```
                04127
A738 AD08D0     04128                   LDA M0PL                ; Merge MISSILE-to-PLAYER
A73B 0D09D0     04129                   ORA M1PL                ;
A73E 0D0AD0     04130                   ORA M2PL                ;
A741 0D0BD0     04131                   ORA M3PL                ;
A744 8583       04132                   STA PL4HIT              ; ...and store them in PL4
A746 AD0FD0     04133                   LDA P3PL                ; Copy PLAYER3-to-PLAYER c
A749 8582       04134                   STA PL3HIT              ;
                04135
A74B 68         04136 JUMP004           PLA                     ; Pop Y
A74C A8         04137                   TAY                     ;
A74D 68         04138                   PLA                     ; Pop X
A74E AA         04139                   TAX                     ;
A74F 68         04140                   PLA                     ; Pop A
A750 40         04141                   RTI                     ; Return from interrupt
                04142
                04143 ;*****************************************************************
                04144 ;*
                04145 ;*                                   IRQHNDLR
                04146 ;*
                04147 ;*                         Interrupt Request (IRQ) Handler
                04148 ;*
                04149 ;*****************************************************************
                04150
                04151 ; DESCRIPTION
                04152 ;
                04153 ; This subroutine is executed during immediate interrupt requests
                04154 ; as after pressing a key on the keyboard. It clears and disables
                04155 ; except the interrupt raised by a pressed key. If a key has been
                04156 ; hardware code is collected and the bits of the SHIFT and CONTROL
                04157 ; added. The resulting keyboard code is stored in KEYCODE ($CA).
                04158
A751 48         04159 IRQHNDLR          PHA                     ; Push A
A752 A900       04160                   LDA #0                  ; POKEY: Disable all IRQs
A754 8D0ED2     04161                   STA IRQEN               ;
A757 A940       04162                   LDA #$40                ; POKEY: Enable keyboard i
A759 8D0ED2     04163                   STA IRQEN               ;
A75C AD09D2     04164                   LDA KBCODE              ; POKEY: Load keyboard key
A75F 09C0       04165                   ORA #$C0                ; Combine with SHIFT and C
A761 85CA       04166                   STA KEYCODE             ; Store keyboard code
A763 68         04167                   PLA                     ; Pop A
A764 40         04168                   RTI                     ; Return from interrupt
                04169
                04170 ;*****************************************************************
                04171 ;*
                04172 ;*                                   DRAWLINES
                04173 ;*
                04174 ;*                       Draw horizontal and vertical lines
                04175 ;*
                04176 ;*****************************************************************
                04177
                04178 ; DESCRIPTION
                04179 ;
                04180 ; Draws the Attack Computer Display (in Front view), cross hairs (
                04181 ; Aft view), and our starship's shape (in Long-Range Scan view) on
                04182 ; (if the Attack Computer is not destroyed) by being passed an off
                04183 ; DRAWLINESTAB ($BAF9). This table consists of a list of 3-byte el
                04184 ; terminated by an end marker byte ($FE). Each such element define
                04185 ; horizontal or vertical line, and is passed via memory addresses
```

```
                    04186 ; PENROW ($A5), and PENCOLUMN ($A6) to subroutine DRAWLINE ($A782)
                    04187 ; executes the actual drawing. See subroutine DRAWLINE ($A782) and
                    04188 ; DRAWLINESTAB ($BAF9) for a description of the 3-byte elements.
                    04189 ;
                    04190 ; With every call of this subroutine the blip cycle counter is ini
                    04191 ; the start of the DELAY phase (see subroutine UPDATTCOMP ($A7BF))
                    04192 ;
                    04193 ; NOTE: The entry to this subroutine is in mid-code, not at the be
                    04194 ;
                    04195 ; INPUT
                    04196 ;
                    04197 ;   X = Offset into DRAWLINESTAB ($BAF9). Used values are:
                    04198 ;     $00 -> Draw Attack Computer Display and cross hairs (Front v
                    04199 ;     $2A -> Draw Aft view cross hairs (Aft view)
                    04200 ;     $31 -> Draw our starship's shape (Long-Range Scan view)
                    04201
A765 99A400         04202 LOOP028         STA DIRLEN,Y            ; Store byte of 3-byte ele
A768 E8             04203                 INX                     ;
A769 88             04204                 DEY                     ;
A76A 100E           04205                 BPL SKIP048             ; Next byte of 3-byte elem
A76C 2082A7         04206                 JSR DRAWLINE            ; Draw line on PLAYFIELD
                    04207
A76F A905           04208 DRAWLINES       LDA #5                  ; Init blip cycle to DELAY
A771 85A2           04209                 STA BLIPCYCLECNT        ; ...delays drawing each r
                    04210
A773 2C9509         04211                 BIT GCSTATCOM           ; Return if Attack Compute
A776 7009           04212                 BVS SKIP049             ;
                    04213
A778 A002           04214                 LDY #2                  ;
A77A BDF9BA         04215 SKIP048         LDA DRAWLINESTAB,X      ; Load byte of 3-byte elem
A77D C9FE           04216                 CMP #$FE                ; Loop until end marker by
A77F D0E4           04217                 BNE LOOP028             ;
A781 60             04218 SKIP049         RTS                     ; Return
                    04219
                    04220 ;*****************************************************************
                    04221 ;*
                    04222 ;*                                  DRAWLINE
                    04223 ;*
                    04224 ;*                  Draw a single horizontal or vertical line
                    04225 ;*
                    04226 ;*****************************************************************
                    04227
                    04228 ; DESCRIPTION
                    04229 ;
                    04230 ; Draws a single horizontal or vertical transparent line.
                    04231 ;
                    04232 ; There are two entries to this subroutine:
                    04233 ;
                    04234 ; (1)  DRAWLINE ($A782) is entered from subroutine DRAWLINES ($A76
                    04235 ;      line in COLOR1.
                    04236 ;
                    04237 ; (2)  DRAWLINE2 ($A784) is entered from subroutine UPDATTCOMP ($A
                    04238 ;      the blip in COLOR2 in the Attack Computer Display.
                    04239 ;
                    04240 ; The position, direction, and length of the line is defined by th
                    04241 ; passed in memory addresses DIRLEN ($A4), PENROW ($A5), and PENCO
                    04242 ;
                    04243 ; A drawing operation draws one transparent line. It uses both the
                    04244 ; register number of the overwritten (old) and the overwriting (ne
```

```
                04245 ; decide on the new pixel color register number. This results in a
                04246 ; drawing effect. See the table below for all resulting combinatio
                04247 ; registers.
                04248 ;
                04249 ; +-----------+---------------+
                04250 ; |           |   Old Color   |
                04251 ; |           |   Register    |
                04252 ; | New Color +---------------+
                04253 ; | Register  | 0 | 1 | 2 | 3 |
                04254 ; +-----------+---+---+---+---+
                04255 ; |         0 | 0 | 1 | 2 | 3 |
                04256 ; +-----------+---+---+---+---+
                04257 ; |         1 | 1 | 1 | 3 | 3 |
                04258 ; +-----------+---+---+---+---+
                04259 ; |         2 | 2 | 3 | 2 | 3 |
                04260 ; +-----------+---+---+---+---+
                04261 ; |         3 | 3 | 3 | 3 | 3 |
                04262 ; +-----------+---+---+---+---+
                04263 ;
                04264 ; For example, COLOR1 overwritten by COLOR2 yields COLOR3. If you
                04265 ; at the blip (in COLOR2) on the Attack Computer Display (in COLOR
                04266 ; of the Attack Computer Display shine through (in COLOR3) where t
                04267 ;
                04268 ; INPUT
                04269 ;
                04270 ;   DIRLEN    ($A4) = B7 = 0 -> Draw line to the right
                04271 ;                     B7 = 1 -> Draw line downward
                04272 ;                     B6..0  -> Length of line in pixels
                04273 ;   PENROW    ($A5) = Start pixel row number of line
                04274 ;   PENCOLUMN ($A6) = Start pixel column number of line
                04275
    =006A       04276 L.PIXELBYTEOFF  = $6A                   ; Within-row-offset to byt
    =006B       04277 L.BITPAT        = $6B                   ; 1-byte bit pattern for 4
    =006E       04278 L.DIRSAV        = $6E                   ; Saves DIRLEN
                04279
A782 A955       04280 DRAWLINE        LDA #$55                ; Copy 1-byte bit pattern
A784 856B       04281 DRAWLINE2       STA L.BITPAT            ;
A786 A5A4       04282                 LDA DIRLEN              ; Copy direction (and leng
A788 856E       04283                 STA L.DIRSAV            ;
A78A 297F       04284                 AND #$7F                ; Strip direction bit
A78C 85A4       04285                 STA DIRLEN              ; Store length of line
                04286
A78E A4A5       04287 LOOP029         LDY PENROW              ; Loop over length of line
A790 B90008     04288                 LDA PFMEMROWLO,Y        ; Point MEMPTR to start of
A793 8568       04289                 STA MEMPTR              ; ...in PLAYFIELD memory
A795 B96408     04290                 LDA PFMEMROWHI,Y        ;
A798 8569       04291                 STA MEMPTR+1            ;
                04292
A79A A5A6       04293                 LDA PENCOLUMN           ; Calc and store pen's byt
A79C 4A         04294                 LSR A                   ;
A79D 4A         04295                 LSR A                   ;
A79E 856A       04296                 STA L.PIXELBYTEOFF      ;
                04297
A7A0 A5A6       04298                 LDA PENCOLUMN           ; Calc pixel-within-byte i
A7A2 2903       04299                 AND #$03                ;
A7A4 A8         04300                 TAY                     ;
                04301
A7A5 B9B0BA     04302                 LDA PIXELMASKTAB,Y      ; Pick mask to filter pixe
A7A8 256B       04303                 AND L.BITPAT            ; ...AND with bit pattern
```

```
A7AA A46A     04304                LDY L.PIXELBYTEOFF      ;
A7AC 1168     04305                ORA (MEMPTR),Y          ; Blend byte with new pixe
A7AE 9168     04306                STA (MEMPTR),Y          ; ...and store it back in
              04307
A7B0 246E     04308                BIT L.DIRSAV            ; Check direction bit B7
A7B2 1004     04309                BPL SKIP050             ;
A7B4 E6A5     04310                INC PENROW              ; If B7 = 1 -> Increment p
A7B6 D002     04311                BNE SKIP051             ;
A7B8 E6A6     04312 SKIP050        INC PENCOLUMN           ; If B7 = 0 -> Increment p
              04313
A7BA C6A4     04314 SKIP051        DEC DIRLEN              ;
A7BC D0D0     04315                BNE LOOP029             ; Next pixel of line
A7BE 60       04316                RTS                     ; Return
              04317
              04318 ;*****************************************************************
              04319 ;*
              04320 ;*                                 UPDATTCOMP
              04321 ;*
              04322 ;*                       Update Attack Computer Display
              04323 ;*
              04324 ;*****************************************************************
              04325
              04326 ; DESCRIPTION
              04327 ;
              04328 ; Draws the blip of the tracked space object and the lock-on marke
              04329 ; Attack Computer Display. The actual drawing follows a cycle of 1
              04330 ; iterations (numbered by this subroutine as "blip cycles" 0..10),
              04331 ; divided into three phases:
              04332 ;
              04333 ; (1)  Blip cycle 0..4: Draw blip shape row-by-row
              04334 ;
              04335 ;      Draw the blip's shape into the Attack Computer Display, one
              04336 ;      cycle. After 5 blip cycles the blip shape is complete and c
              04337 ;      visible because between blip cycles, that is, game loop ite
              04338 ;      PLAYFIELD is not erased (only the PLAYFIELD space objects a
              04339 ;      is executed by branching to entry DRAWLINE2 ($A784) of subr
              04340 ;      DRAWLINE ($A782). The blip shape is retrieved from table BL
              04341 ;      ($BF6E).
              04342 ;
              04343 ; (2)  Blip cycle 5..9: Delay
              04344 ;
              04345 ;      Delay the execution of blip cycle 10.
              04346 ;
              04347 ; (3)  Blip cycle 10: Update Attack Computer Display
              04348 ;
              04349 ;      After verifying that the tracked space object is alive, cal
              04350 ;      blip's relative top-left pixel column and row number. The r
              04351 ;      values are in -11..11 and -6..4, relative to the blip's top
              04352 ;      reference position at pixel column number 131 and pixel row
              04353 ;      respectively.
              04354 ;
              04355 ;      Filter the Attack Computer Display area: Only pixels of COL
              04356 ;      inner frame area (a 28 pixel wide x 15 pixel high rectangle
              04357 ;      top-left corner at pixel column number 120 and pixel row nu
              04358 ;      the filter operation. This effectively erases the blip.
              04359 ;
              04360 ;      If the blip is within -2..+2 pixels off its horizontal refe
              04361 ;      (pixel column numbers 129..132) then the tracked space obje
              04362 ;      lock-on. Draw the x lock-on marker.
```

```
;
;       If the tracked space object is in x lock-on and the blip is
;       pixels off its vertical reference position (pixel column nu
;       then the tracked space object is in x and y lock-on. Draw a
;       lock-on marker.
;
;       If the tracked space object is in x and y lock-on and the t
;       object's z-coordinate < +3072 (+$0C**) <KM> then the tracke
;       is in x, y and z lock-on. Draw also the z lock-on marker.
;
;       If the tracked space object is in x, y, and z lock-on (and
;       optimal firing range) set the ISINLOCKON ($A3) flag.
;
;       The following sketches show the Attack Computer Display are
;       with the Attack Computer Display frame:
;
;           119                                   119
;       70 ################################    70 ####################
;          #         ....#....          #       #              #
;          #         ....#....          #       #              #
;          #         ....#....          #       #              #
;          #         ....#....          #       #              #
;          #      ###############       #       #......#############
;          #XXXX  #  .........  #   XXXX#       #......#...........
;          #      #  ..$......  #       #       #......#....$......
;          ########  .........  #########       ########............
;          #      #  .........  #       #       #......#...........
;          #      #  .........  #       #       #YYYY..#...........
;          #      ###############       #       #......#############
;          #         ....#....          #       #............#....
;          #         ....#....          #       #              #
;          #         ....#....          #       #              #
;          #         ....#....          #       #              #
;          ################################      ####################
;
;          X = x lock-on marker                  Y = y lock-on marke
;          . = x lock-on blip zone               . = y lock-on blip
;          $ = Blip's top-left reference         $ = Blip's top-left
;              position                              position
;
;          119
;       70 ################################
;          #              #                #
;          #              #                #
;          #              #                #
;          #              #                #
;          #      ###############          #
;          #      #              #         #
;          #      #    $         #         #
;          ########              #########
;          #      #              #         #
;          #      #              #         #
;          #      ###############          #
;          #             #                 #
;          #             #                 #
;          #        ZZ   #  ZZ             #
;          #        ZZ   #  ZZ             #
;          ################################
;
```

```
                04422 ;         Z = z lock-on marker
                04423 ;         $ = Blip's top-left reference
                04424 ;             position
                04425
    =006C       04426 L.SHIFTSHAP     = $6C                 ; Saves shifted byte of bl
                04427
A7BF AE5C09     04428 UPDATTCOMP      LDX TRACKDIGIT        ; Load index of tracked sp
A7C2 A4A2       04429                 LDY BLIPCYCLECNT      ; Load blip cycle counter
A7C4 C005       04430                 CPY #5                ;
A7C6 B024       04431                 BCS SKIP054           ; Skip drawing blip if bli
                04432
                04433 ;*** Blip cycle 0..4: Draw blip shape one row each cycle *********
A7C8 A5A0       04434                 LDA BLIPCOLUMN        ; Init pen's pixel column
A7CA 85A6       04435                 STA PENCOLUMN         ; ...with top position of
A7CC B96EBF     04436                 LDA BLIPSHAPTAB,Y     ; Load bit pattern of one
A7CF 0A         04437 LOOP030         ASL A                 ; Shift bit pattern one po
A7D0 856C       04438                 STA L.SHIFTSHAP       ; Temporarily save shifted
A7D2 900D       04439                 BCC SKIP052           ; Skip if shifted-out bit
                04440
A7D4 A981       04441                 LDA #$81              ; Store "draw a line of 1
A7D6 85A4       04442                 STA DIRLEN            ; ...for call to DRAWLINE2
                04443
A7D8 A5A1       04444                 LDA BLIPROW           ; Init pen's pixel row num
A7DA 85A5       04445                 STA PENROW            ; ...with leftmost positio
A7DC A9AA       04446                 LDA #$AA              ; Load 1-byte bit pattern
A7DE 2084A7     04447                 JSR DRAWLINE2         ; Draw pixel on PLAYFIELD
                04448
A7E1 E6A6       04449 SKIP052         INC PENCOLUMN         ; Move pen one pixel to th
A7E3 A56C       04450                 LDA L.SHIFTSHAP       ; Reload shifted shape byt
A7E5 D0E8       04451                 BNE LOOP030           ; Next horizontal pixel of
                04452
A7E7 E6A1       04453                 INC BLIPROW           ; Move pen one pixel downw
A7E9 E6A2       04454 SKIP053         INC BLIPCYCLECNT      ; Increment blip cycle cou
A7EB 60         04455                 RTS                   ; Return
                04456
                04457 ;*** Blip cycle 5..9: Delay **************************************
A7EC C00A       04458 SKIP054         CPY #10               ; Return if blip cycle < 1
A7EE 90F9       04459                 BCC SKIP053           ;
                04460
                04461 ;*** Blip cycle 10: Calculate new blip pixel row and column number
A7F0 B5E9       04462                 LDA PL0LIFE,X         ; Skip if tracked object n
A7F2 F03C       04463                 BEQ SKIP059           ;
                04464
A7F4 BD710A     04465                 LDA XPOSHI,X          ; Map x-coordinate of trac
A7F7 BCDE09     04466                 LDY XPOSSIGN,X        ; Skip if tracked object o
A7FA F008       04467                 BEQ SKIP055           ;
                04468
A7FC C90C       04469                 CMP #12               ; Skip if x of tracked obj
A7FE 900A       04470                 BCC SKIP056           ;
A800 A90B       04471                 LDA #11               ; Prep relative pixel colu
A802 1006       04472                 BPL SKIP056           ;
                04473
A804 C9F5       04474 SKIP055         CMP #-11              ; Skip if x of tracked obj
A806 B002       04475                 BCS SKIP056           ;
A808 A9F5       04476                 LDA #-11              ; Prep relative pixel colu
                04477
A80A 18         04478 SKIP056         CLC                   ; Add 131 (= blip's top-le
A80B 6983       04479                 ADC #131              ;
A80D 85A0       04480                 STA BLIPCOLUMN        ; BLIPCOLUMN := 131 + -11.
```

```
                04481
A80F BDA20A     04482                   LDA YPOSHI,X            ; Map y-coordinate of trac
A812 49FF       04483                   EOR #$FF                ; Mirror y-coordinate on y
A814 BC0F0A     04484                   LDY YPOSSIGN,X          ; Skip if tracked obj on l
A817 D008       04485                   BNE SKIP057             ;
                04486
A819 C905       04487                   CMP #5                  ; Skip if mirrored y of tr
A81B 900A       04488                   BCC SKIP058             ;
A81D A904       04489                   LDA #4                  ; Prep relative pixel row
A81F 1006       04490                   BPL SKIP058             ;
                04491
A821 C9FA       04492 SKIP057           CMP #-6                 ; Skip if mirrored y of tr
A823 B002       04493                   BCS SKIP058             ;
A825 A9FA       04494                   LDA #-6                 ; Prep relative pixel row
                04495
A827 18         04496 SKIP058           CLC                     ; Add 77 (= blip's top-lef
A828 694D       04497                   ADC #77                 ;
A82A 85A1       04498                   STA BLIPROW             ; BLIPROW := 77 + -6..4
                04499
A82C A900       04500                   LDA #0                  ; Reset blip cycle
A82E 85A2       04501                   STA BLIPCYCLECNT        ;
                04502
                04503 ;*** Filter Attack Computer Display frame area ******************
                04504                                           ; PLAYFIELD address of top
    =1B36       04505 PFMEM.C120R71     = PFMEM+71*40+120/4     ; Display's inner frame @
                04506
A830 A936       04507 SKIP059           LDA #<PFMEM.C120R71     ; Point MEMPTR to start of
A832 8568       04508                   STA MEMPTR              ; ...inner top-left corner
A834 A91B       04509                   LDA #>PFMEM.C120R71     ; ...in PLAYFIELD memory
A836 8569       04510                   STA MEMPTR+1            ;
                04511
A838 A20E       04512                   LDX #14                 ; Traverse a 28 x 15 pixel
A83A A006       04513 LOOP031           LDY #6                  ;
A83C B168       04514 LOOP032           LDA (MEMPTR),Y          ; Load byte (4 pixels) fro
A83E 2955       04515                   AND #$55                ; Filter COLOR1 pixels
A840 9168       04516                   STA (MEMPTR),Y          ; Store byte (4 pixels) ba
A842 88         04517                   DEY                     ;
A843 10F7       04518                   BPL LOOP032             ; Next 4 pixels in x-direc
                04519
A845 18         04520                   CLC                     ; Add 40 to MEMPTR
A846 A568       04521                   LDA MEMPTR              ; (40 bytes = 160 pixels =
A848 6928       04522                   ADC #40                 ;
A84A 8568       04523                   STA MEMPTR              ;
A84C 9002       04524                   BCC SKIP060             ;
A84E E669       04525                   INC MEMPTR+1            ;
                04526
A850 CA         04527 SKIP060           DEX                     ;
A851 10E7       04528                   BPL LOOP031             ; Next row of pixels in y-
                04529
                04530 ;*** Prepare lock-on marker checks ******************************
A853 AE5C09     04531                   LDX TRACKDIGIT          ; Preload index of tracked
A856 C8         04532                   INY                     ; Y := 0, preloaded value
                04533
                04534 ;*** Draw lock-on markers ***************************************
                04535                                           ; PLAYFIELD addresses of
    =1BFE       04536 PFMEM.C120R76     = PFMEM+76*40+120/4     ; ...x lock-on marker @ pi
    =1C04       04537 PFMEM.C144R76     = PFMEM+76*40+144/4     ; ...x lock-on marker @ pi
    =1C9E       04538 PFMEM.C120R80     = PFMEM+80*40+120/4     ; ...y lock-on marker @ pi
    =1CA4       04539 PFMEM.C144R80     = PFMEM+80*40+144/4     ; ...y lock-on marker @ pi
```

```
    =1D40     04540 PFMEM.C128R84   = PFMEM+84*40+128/4     ; ...z lock-on marker @ pi
    =1D68     04541 PFMEM.C128R85   = PFMEM+85*40+128/4     ; ...z lock-on marker @ pi
    =1D42     04542 PFMEM.C136R84   = PFMEM+84*40+136/4     ; ...z lock-on marker @ pi
    =1D6A     04543 PFMEM.C136R85   = PFMEM+85*40+136/4     ; ...z lock-on marker @ pi
              04544
A857 A588     04545                 LDA LOCKONLIFE          ; If lock-on lifetime expi
A859 F004     04546                 BEQ SKIP061             ;
              04547
A85B C688     04548                 DEC LOCKONLIFE          ; else decrem. lock-on lif
A85D D039     04549                 BNE SKIP062             ;
              04550
A85F A5A0     04551 SKIP061         LDA BLIPCOLUMN          ; Skip x, y, and z lock-on
A861 C981     04552                 CMP #129                ; ...top-left pixel column
A863 9033     04553                 BCC SKIP062             ;
A865 C985     04554                 CMP #133                ;
A867 B02F     04555                 BCS SKIP062             ;
              04556
A869 A9AA     04557                 LDA #$AA                ; Draw x lock-on marker (4
A86B 8DFE1B   04558                 STA PFMEM.C120R76       ; ...at pixel column 120,
A86E 8D041C   04559                 STA PFMEM.C144R76       ; ...at pixel column 144,
              04560
A871 A5A1     04561                 LDA BLIPROW             ; Skip y and z lock-on mar
A873 C94B     04562                 CMP #75                 ; ...top-left pixel row nu
A875 9021     04563                 BCC SKIP062             ;
A877 C94F     04564                 CMP #79                 ;
A879 B01D     04565                 BCS SKIP062             ;
              04566
A87B A9AA     04567                 LDA #$AA                ; Draw y lock-on marker (4
A87D 8D9E1C   04568                 STA PFMEM.C120R80       ; ...at pixel column 120,
A880 8DA41C   04569                 STA PFMEM.C144R80       ; ...at pixel column 144,
              04570
A883 BD400A   04571                 LDA ZPOSHI,X            ; Skip z lock-on marker if
A886 C90C     04572                 CMP #12                 ;
A888 B00E     04573                 BCS SKIP062             ;
              04574
A88A A0A0     04575                 LDY #$A0                ; Draw z lock-on marker (2
A88C 8C401D   04576                 STY PFMEM.C128R84       ; ...at pixel column 128,
A88F 8C681D   04577                 STY PFMEM.C128R85       ; ...at pixel column 128,
A892 8C421D   04578                 STY PFMEM.C136R84       ; ...at pixel column 136,
A895 8C6A1D   04579                 STY PFMEM.C136R85       ; ...at pixel column 136,
              04580
A898 84A3     04581 SKIP062         STY ISINLOCKON          ; Store lock-on flag (> 0
A89A 60       04582                 RTS                     ; Return
              04583
              04584 ;****************************************************************
              04585 ;*
              04586 ;*                                HYPERWARP
              04587 ;*
              04588 ;*                             Handle hyperwarp
              04589 ;*
              04590 ;****************************************************************
              04591
              04592 ; DESCRIPTION
              04593 ;
              04594 ; Handles the hyperwarp sequence, which transports our starship fr
              04595 ; to another. It can be divided into four phases:
              04596 ;
              04597 ; (1)  ACCELERATION PHASE
              04598 ;
```

```
04599 ;      The ACCELERATION PHASE is entered after the hyperwarp seque
04600 ;      engaged in subroutine KEYBOARD ($AFFE) by pressing the 'H'
04601 ;
04602 ;      The Hyperwarp Target Marker appears and our starship begins
04603 ;      accelerate. When our starship's velocity reaches 128 <KM/H>
04604 ;      readout of the Control Panel Display displays "50"), the ST
04605 ;      is entered.
04606 ;
04607 ;      The Hyperwarp Target Marker is represented by a space objec
04608 ;      distance away in front of our starship as PLAYER3. It has a
04609 ;      144 game loop iterations and is tracked. Thus, tracking han
04610 ;      subroutine UPDATTCOMP ($A7BF) provides drawing the x and y
04611 ;      markers in the Attack Computer Display when the Hyperwarp T
04612 ;      is centered.
04613 ;
04614 ;      A temporary arrival location on the Galactic Chart was save
04615 ;      hyperwarp was engaged in subroutine KEYBOARD ($AFFE). Durin
04616 ;      ACCELERATION PHASE (and the subsequent STAR TRAIL PHASE) th
04617 ;      constantly updated depending on how much the Hyperwarp Targ
04618 ;      veers off its center position.
04619 ;
04620 ;      The actual arrival hyperwarp marker row and column numbers
04621 ;      Galactic Chart are the sum of the temporary arrival hyperwa
04622 ;      and column numbers stored when engaging the hyperwarp in su
04623 ;      KEYBOARD ($AFFE) and the number of Player/Missile (PM) pixe
04624 ;      Hyperwarp Target Marker is off-center vertically and horizo
04625 ;      respectively, at the end of the STAR TRAIL PHASE.
04626 ;
04627 ;      NOTE: The used vertical center value of 119 PM pixels is th
04628 ;      number of the top edge of the centered Hyperwarp Target Mar
04629 ;      to bottom: 8 PM pixels to the start of Display List + 16 PM
04630 ;      lines + 100 PM pixels to the vertical PLAYFIELD center - 5
04631 ;      relative offset of the Hyperwarp Target Marker's shape cent
04632 ;      shape's top edge = 119 PM pixels). Recall also that PLAYERs
04633 ;      single-line resolution have PM pixels that are half as high
04634 ;      wide.
04635 ;
04636 ;      NOTE: The used horizontal center value of 125 PM pixels is
04637 ;      row number of the left edge of the centered Hyperwarp Targe
04638 ;      left to right: 127 PM pixels to the PLAYFIELD center - 3 PM
04639 ;      relative offset of the Hyperwarp Target Marker's shape cent
04640 ;      shape's left edge = 125 PM pixels).
04641 ;
04642 ;      If during the ACCELERATION PHASE (and the subsequent STAR T
04643 ;      you switch the Front view to another view, the Hyperwarp Ta
04644 ;      changes to a random position which results in arriving at a
04645 ;      destination sector.
04646 ;
04647 ;      During the ACCELERATION PHASE (and the subsequent STAR TRAI
04648 ;      all but NOVICE missions, the Hyperwarp Target Marker veers
04649 ;      random velocity in x and y direction, which is changed duri
04650 ;      loop iterations. Table VEERMASKTAB ($BED7) limits the maxim
04651 ;      velocity depending on the mission level:
04652 ;
04653 ;      +-----------+-----------------------------+
04654 ;      |  Mission  |      Veer-Off Velocity      |
04655 ;      +-----------+-----------------------------+
04656 ;      | NOVICE    |                   0  <KM/H> |
04657 ;      | PILOT     |  -63..-16, +16..+63  <KM/H> |
```

```
04658 ;      | WARRIOR   |  -95..-16, +16..+95  <KM/H> |
04659 ;      | COMMANDER | -127..-16, +16..+127 <KM/H> |
04660 ;      +-----------+-----------------------------+
04661 ;
04662 ; (2)  STAR TRAIL PHASE
04663 ;
04664 ;      When our starship's velocity reaches a velocity of 128 <KM/
04665 ;      VELOCITY readout of the Control Panel Display displays "50"
04666 ;      to all effects of the ACCELERATION PHASE, multiple star tra
04667 ;      appear while our starship continues to accelerate. Each sta
04668 ;      initialized in subroutine INITTRAIL ($A9B4).
04669 ;
04670 ; (3)  HYPERSPACE PHASE
04671 ;
04672 ;      When our starship's velocity reaches a velocity of 254 <KM/
04673 ;      VELOCITY readout of the Control Panel Display displays "99"
04674 ;      enters the HYPERSPACE PHASE (the VELOCITY readout of the Co
04675 ;      Display displays the infinity symbol).
04676 ;
04677 ;      During the first pass of the HYPERSPACE PHASE the hyperwarp
04678 ;      to HYPERSPACE. This makes the stars and the Hyperwarp Targe
04679 ;      disappear in GAMELOOP ($A1F3). Then, the beeper sound patte
04680 ;      TRANSIT is played in subroutine BEEP ($B3A6), the hyperwarp
04681 ;      required hyperwarp energy is calculated in subroutine CALCW
04682 ;      and the title line is preloaded with "HYPERSPACE". Code exe
04683 ;      via calling subroutine CLEANUPWARP ($A98D) where game varia
04684 ;      already initialized to their post-hyperwarp values.
04685 ;
04686 ;      During subsequent passes of the HYPERSPACE PHASE, the calcu
04687 ;      hyperwarp energy is decremented in chunks of 10 energy unit
04688 ;      execution returns via calling subroutine DECENERGY ($B86F),
04689 ;      decrements our starship's energy. After the calculated hype
04690 ;      is spent the DECELERATION PHASE is entered.
04691 ;
04692 ; (4)  DECELERATION PHASE
04693 ;
04694 ;      The title line flashes "HYPERWARP COMPLETE", the star field
04695 ;      our starship decelerates to a stop. The Engines and the hyp
04696 ;      disengaged and stopped in subroutine ENDWARP ($A987), the a
04697 ;      coordinates on the Galactic Chart are initialized, as well
04698 ;      vicinity mask.
04699 ;
04700 ;      The vicinity mask limits the position vector components (co
04701 ;      space objects in the arrival sector relative to our starshi
04702 ;      vicinity mask is picked from table VICINITYMASKTAB ($BFB3)
04703 ;      calculated by the arrival y-coordinate modulo 8: The more y
04704 ;      the arrival hyperwarp marker in the vertical center of a se
04705 ;      Galactic Chart, the closer space objects in this sector wil
04706 ;      starship. For example, if you placed the arrival hyperwarp
04707 ;      in the vertical middle of the sector the index will be 3, t
04708 ;      objects inside the arrival sector will be in the vicinity o
04709 ;      of our starship. The following table lists the possible coo
04710 ;      depending on the calculated index:
04711 ;
04712 ;      +-------+----------------------+
04713 ;      | Index |    ABS(Coordinate)   |
04714 ;      +-------+----------------------+
04715 ;      |   0   | <= 65535 ($FF**) <KM> |
04716 ;      |   1   | <= 65535 ($FF**) <KM> |
```

```
                04717 ;       |   2   | <= 16383 ($3F**) <KM> |
                04718 ;       |   3   | <=  4095 ($0F**) <KM> |
                04719 ;       |   4   | <= 16383 ($3F**) <KM> |
                04720 ;       |   5   | <= 32767 ($7F**) <KM> |
                04721 ;       |   6   | <= 65535 ($FF**) <KM> |
                04722 ;       |   7   | <= 65535 ($FF**) <KM> |
                04723 ;       +-------+----------------------+
                04724 ;
                04725 ;       If there is a starbase in the arrival sector, its x and y c
                04726 ;       initialized to random values within the interval defined by
                04727 ;       mask by using subroutine RNDINVXY ($B7BE). Its z-coordinate
                04728 ;       a value >= +$71** (+28928) <KM>. Its velocity vector compon
                04729 ;       to 0 <KM/H>.
                04730 ;
                04731 ;       If there are Zylon ships in the arrival sector then a red a
                04732 ;       initialized by setting the red alert lifetime to 255 game l
                04733 ;       iterations, playing the beeper sound pattern RED ALERT in s
                04734 ;       ($B3A6) and setting the title phrase to "RED ALERT".
                04735
A89B A4C0       04736 HYPERWARP       LDY WARPSTATE           ; Return if hyperwarp not
A89D F061       04737                 BEQ SKIP066             ;
                04738
A89F A570       04739                 LDA VELOCITYLO          ; If velocity >= 254 <KM/H
A8A1 C9FE       04740                 CMP #254                ;
A8A3 B05C       04741                 BCS SKIP067             ;
                04742
A8A5 C980       04743                 CMP #128                ; If velocity < 128 <KM/H>
A8A7 9003       04744                 BCC SKIP063             ;
                04745
                04746 ;*** STAR TRAIL PHASE ***********************************************
A8A9 20B4A9     04747                 JSR INITTRAIL           ; Init star trail
                04748
                04749 ;*** ACCELERATION PHASE *********************************************
A8AC A903       04750 SKIP063         LDA #3                  ; Track Hyperwarp Target M
A8AE 8D5C09     04751                 STA TRACKDIGIT          ;
                04752
A8B1 A990       04753                 LDA #SHAP.HYPERWARP     ; PLAYER3 is HYPERWARP TAR
A8B3 8D8F0C     04754                 STA PL3SHAPTYPE         ;
A8B6 85EC       04755                 STA PL3LIFE             ; PLAYER3 lifetime := 144
                04756
A8B8 A91F       04757                 LDA #$1F                ; PLAYER3 z-coordinate :=
A8BA 8D430A     04758                 STA PL3ZPOSHI           ;
                04759
A8BD 38         04760                 SEC                     ; New arrival hyperwarp ma
A8BE ADFC0B     04761                 LDA PL3ROWNEW           ; WARPARRVROW := WARPTEMPR
A8C1 E977       04762                 SBC #119                ; ... - 119 PM pixels (top
A8C3 18         04763                 CLC                     ; ...Hyperwarp Target Mark
A8C4 65C5       04764                 ADC WARPTEMPROW         ;
A8C6 297F       04765                 AND #$7F                ; Limit WARPARRVROW to 0..
A8C8 858E       04766                 STA WARPARRVROW         ;
                04767
A8CA 38         04768                 SEC                     ; New arrival hyperwarp ma
A8CB AD2D0C     04769                 LDA PL3COLUMN           ; WARPARRVCOLUMN := WARPTE
A8CE E97D       04770                 SBC #125                ; ... - 125 PM pixels (lef
A8D0 18         04771                 CLC                     ; ...Hyperwarp Target Mark
A8D1 65C4       04772                 ADC WARPTEMPCOLUMN      ;
A8D3 297F       04773                 AND #$7F                ; Limit WARPARRVCOLUMN to
A8D5 858F       04774                 STA WARPARRVCOLUMN      ;
                04775
```

```
A8D7 A562     04776                LDA MISSIONLEVEL        ; Skip if NOVICE mission
A8D9 F011     04777                BEQ SKIP065             ;
              04778
A8DB AD0AD2   04779                LDA RANDOM              ; Prep random number
A8DE A4D0     04780                LDY SHIPVIEW            ; Skip if in Front view
A8E0 F006     04781                BEQ SKIP064             ;
              04782
A8E2 8D2D0C   04783                STA PL3COLUMN           ; Randomize PM pixel row a
A8E5 8DFC0B   04784                STA PL3ROWNEW           ; ...of Hyperwarp Target M
              04785
A8E8 C910     04786 SKIP064        CMP #16                 ; Return in 94% (240:256)
A8EA B014     04787                BCS SKIP066             ;
              04788
              04789 ;*** Veer off Hyperwarp Target Marker and return *****************
A8EC AD0AD2   04790 SKIP065        LDA RANDOM              ; Prep random x-velocity o
A8EF 0910     04791                ORA #$10                ; Velocity value >= 16 <KM
A8F1 25C6     04792                AND VEERMASK            ; Limit velocity value by
A8F3 8D9A0B   04793                STA PL3XVEL             ; PLAYER3 x-velocity := ve
              04794
A8F6 AD0AD2   04795                LDA RANDOM              ; Prep random y-velocity o
A8F9 0910     04796                ORA #$10                ; Velocity value >= 16 <KM
A8FB 25C6     04797                AND VEERMASK            ; Limit velocity value by
A8FD 8DCB0B   04798                STA PL3YVEL             ; PLAYER3 y-velocity := ve
A900 60       04799 SKIP066        RTS                     ; Return
              04800
              04801 ;*** HYPERSPACE PHASE ***************************************
A901 98       04802 SKIP067        TYA                     ; Skip if already in HYPER
A902 3011     04803                BMI SKIP068             ;
              04804
              04805 ;*** HYPERSPACE PHASE (First pass) ******************************
A904 A9FF     04806                LDA #$FF                ; Set hyperwarp state to H
A906 85C0     04807                STA WARPSTATE           ;
              04808
A908 A200     04809                LDX #$00                ; Play beeper sound patter
A90A 20A6B3   04810                JSR BEEP                ;
              04811
A90D 20A7B1   04812                JSR CALCWARP            ; Calc hyperwarp energy
              04813
A910 A01B     04814                LDY #$1B                ; Prep title phrase "HYPER
A912 4C8DA9   04815                JMP CLEANUPWARP         ; Return via CLEANUPWARP
              04816
              04817 ;*** HYPERSPACE PHASE (Second and later passes) *****************
A915 C691     04818 SKIP068        DEC WARPENERGY          ; Decrement energy in chun
A917 F005     04819                BEQ SKIP069             ; Skip to DECELERATION PHA
              04820
A919 A202     04821                LDX #2                  ; ENERGY := ENERGY - 10 an
A91B 4C6FB8   04822                JMP DECENERGY           ;
              04823
              04824 ;*** DECELERATION PHASE ***************************************
A91E A019     04825 SKIP069        LDY #$19                ; Prep title phrase "HYPER
A920 2087A9   04826                JSR ENDWARP             ; Stop our starship
              04827
A923 A58F     04828                LDA WARPARRVCOLUMN      ; Make the arrival hyperwa
A925 858D     04829                STA WARPDEPRCOLUMN      ; ...the departure hyperwa
A927 A58E     04830                LDA WARPARRVROW         ; Make the arrival hyperwa
A929 858C     04831                STA WARPDEPRROW         ; ...the departure hyperwa
              04832
A92B 4A       04833                LSR A                   ; B3..1 of arrival hyperwa
A92C 2907     04834                AND #$07                ; ...pick vicinity mask
```

```
A92E AA       04835                TAX                     ;
A92F BDB3BF   04836                LDA VICINITYMASKTAB,X   ;
A932 85C7     04837                STA VICINITYMASK        ; Store vicinity mask (lim
              04838
A934 A492     04839                LDY ARRVSECTOR          ; Make the arrival sector
A936 8490     04840                STY CURRSECTOR          ;
              04841
              04842 ;*** Init starbase in arrival sector ****************************
A938 A900     04843                LDA #0                  ; Clear starbase-in-sector
A93A 857B     04844                STA ISSTARBASESECT      ;
              04845
A93C BEC908   04846                LDX GCMEMMAP,Y          ; Skip if no starbase in a
A93F 102E     04847                BPL SKIP070             ;
              04848
A941 A9FF     04849                LDA #$FF                ; Set starbase-in-sector f
A943 857B     04850                STA ISSTARBASESECT      ;
              04851
              04852 ;*** Set position vector and velocity vector of starbase *********
A945 A000     04853                LDY #0                  ;
A947 A900     04854 LOOP033        LDA #0                  ; Loop over all coordinate
A949 99680B   04855                STA PL2ZVEL,Y           ; Starbase velocity vector
A94C A901     04856                LDA #1                  ;
A94E 99AF09   04857                STA PL2ZPOSSIGN,Y       ; Starbase coordinate sign
A951 AD0AD2   04858                LDA RANDOM              ; Prep random number...
A954 25C7     04859                AND VICINITYMASK        ; ...limit number range by
A956 99420A   04860                STA PL2ZPOSHI,Y         ; ...store in starbase coo
              04861
A959 98       04862                TYA                     ;
A95A 18       04863                CLC                     ;
A95B 6931     04864                ADC #NUMSPCOBJ.ALL      ;
A95D A8       04865                TAY                     ;
A95E C993     04866                CMP #NUMSPCOBJ.ALL*3    ;
A960 90E5     04867                BCC LOOP033             ; Next starbase coordinate
              04868
A962 AD420A   04869                LDA PL2ZPOSHI           ; Force starbase z-coordin
A965 0971     04870                ORA #$71                ;
A967 8D420A   04871                STA PL2ZPOSHI           ;
A96A A202     04872                LDX #2                  ; Randomly invert starbase
A96C 4CBEB7   04873                JMP RNDINVXY            ; ...and return
              04874
              04875 ;*** Flash red alert if Zylon sector entered ********************
A96F F00E     04876 SKIP070        BEQ SKIP071             ; Skip if no Zylon ships i
              04877
A971 A9FF     04878                LDA #255                ; Red alert lifetime := 25
A973 858B     04879                STA REDALERTLIFE        ;
              04880
A975 A206     04881                LDX #$06                ; Play beeper sound patter
A977 20A6B3   04882                JSR BEEP                ;
              04883
A97A A075     04884                LDY #$75                ; Set title phrase "RED AL
A97C 2023B2   04885                JSR SETTITLE            ;
              04886
A97F 60       04887 SKIP071        RTS                     ; Return
              04888
              04889 ;*****************************************************************
              04890 ;*
              04891 ;*                                ABORTWARP
              04892 ;*
              04893 ;*                             Abort hyperwarp
```

```
                04894 ;*
                04895 ;*****************************************************************
                04896
                04897 ; DESCRIPTION
                04898 ;
                04899 ; Aborts hyperwarp.
                04900 ;
                04901 ; This subroutine is entered from subroutine KEYBOARD ($AFFE). It
                04902 ; energy units for aborting the hyperwarp and preloads the title p
                04903 ; "HYPERWARP ABORTED". Code execution continues into subroutine EN
                04904 ; ($A987).
                04905
A980 A201       04906 ABORTWARP       LDX #1                  ; ENERGY := ENERGY - 100 a
A982 206FB8     04907                 JSR DECENERGY           ;
                04908
A985 A017       04909                 LDY #$17                ; Prep title phrase "HYPER
                04910
                04911 ;*****************************************************************
                04912 ;*
                04913 ;*                                   ENDWARP
                04914 ;*
                04915 ;*                                End hyperwarp
                04916 ;*
                04917 ;*****************************************************************
                04918
                04919 ; DESCRIPTION
                04920 ;
                04921 ; Ends hyperwarp.
                04922 ;
                04923 ; This subroutine stops our starship's Engines and resets the hype
                04924 ; Code execution continues into subroutine CLEANUPWARP ($A98D).
                04925
A987 A900       04926 ENDWARP         LDA #0                  ; Stop Engines
A989 8571       04927                 STA NEWVELOCITY         ;
A98B 85C0       04928                 STA WARPSTATE           ; Disengage hyperwarp
                04929
                04930 ;*****************************************************************
                04931 ;*
                04932 ;*                                 CLEANUPWARP
                04933 ;*
                04934 ;*                        Clean up hyperwarp variables
                04935 ;*
                04936 ;*****************************************************************
                04937
                04938 ; DESCRIPTION
                04939 ;
                04940 ; Cleans up after a hyperwarp.
                04941 ;
                04942 ; This subroutine restores many hyperwarp related variables to the
                04943 ; post-hyperwarp values: The number of used space objects is set t
                04944 ; value of 16 (5 PLAYER space objects + 12 PLAYFIELD space objects
                04945 ; counted 0..16), our starship's velocity (high byte) is cleared a
                04946 ; explosion lifetime, the hit badness, the PLAYER3 shape type (Hyp
                04947 ; Marker), the Engines energy drain rate, and the lifetimes of the
                04948 ; docking state is reset as well as the tracking digit. The title
                04949 ; updated with either "HYPERSPACE" or "HYPERWARP ABORTED".
                04950 ;
                04951 ; INPUT
                04952 ;
```

```
                04953 ;   Y = Title phrase offset. Used values are:
                04954 ;     $17 -> "HYPERWARP ABORTED"
                04955 ;     $1B -> "HYPERSPACE"
                04956
A98D A910       04957 CLEANUPWARP     LDA #NUMSPCOBJ.NORM-1   ; Set normal number of spa
A98F 8579       04958                 STA MAXSPCOBJIND        ; (5 PLAYER spc objs + 12
                04959
A991 A900       04960                 LDA #0                  ;
A993 85C1       04961                 STA VELOCITYHI          ; Turn off hyperwarp veloc
A995 8573       04962                 STA EXPLLIFE            ; Explosion lifetime := 0
A997 858A       04963                 STA HITBADNESS          ; HITBADNESS := NO HIT
A999 8D8F0C     04964                 STA PL3SHAPTYPE         ; Clear PLAYER3 shape type
A99C 8580       04965                 STA DRAINENGINES        ; Clear Engines energy dra
A99E C017       04966                 CPY #$17                ; Skip if hyperwarp was ab
A9A0 F004       04967                 BEQ SKIP072             ;
                04968
A9A2 85E9       04969                 STA PL0LIFE             ; Zylon ship 0 lifetime :=
A9A4 85EA       04970                 STA PL1LIFE             ; Zylon ship 1 lifetime :=
                04971
A9A6 85EB       04972 SKIP072         STA PL2LIFE             ; Zylon photon torpedo lif
A9A8 85EC       04973                 STA PL3LIFE             ; Hyperwarp Target Marker
A9AA 85ED       04974                 STA PL4LIFE             ; Photon torpedo 1 lifetim
A9AC 8575       04975                 STA DOCKSTATE           ; DOCKSTATE := NO DOCKING
A9AE 8D5C09     04976                 STA TRACKDIGIT          ; Clear index of tracked s
A9B1 4C23B2     04977                 JMP SETTITLE            ; Set title phrase and ret
                04978
                04979 ;*****************************************************************
                04980 ;*
                04981 ;*                                  INITTRAIL
                04982 ;*
                04983 ;*         Initialize star trail during STAR TRAIL PHASE of hyperw
                04984 ;*
                04985 ;*****************************************************************
                04986
                04987 ; DESCRIPTION
                04988 ;
                04989 ; BACKGROUND
                04990 ;
                04991 ; Star trails are displayed during the STAR TRAIL PHASE, that is,
                04992 ; ACCELERATION PHASE and before the HYPERSPACE PHASE of the hyperw
                04993 ;
                04994 ; A star trail is formed by 6 stars represented by 6 PLAYFIELD spa
                04995 ; with continuous position vector indices in 17..48 (indices are w
                04996 ; when greater than 48). Between the creations of two star trails
                04997 ; of 4 game loop iterations.
                04998 ;
                04999 ; DETAILS
                05000 ;
                05001 ; This subroutine first decrements this star trail creation delay,
                05002 ; the delay is still counting down. If the delay falls below 0 the
                05003 ; accelerating our starship's velocity toward hyperwarp speed and
                05004 ; new star trail:
                05005 ;
                05006 ; First, it raises the maximum index of used space objects to 48 (
                05007 ; number of displayed space objects to 49), resets the star trail
                05008 ; to 4 game loop iterations, and then forms a new star trail of 6
                05009 ; represented by 6 PLAYFIELD space objects. The x and y coordinate
                05010 ; stars are the same, picked randomly from tables WARPSTARXTAB ($B
                05011 ; WARPSTARYTAB ($BB3E), respectively, with their signs changed ran
```

```
                05012 ; z-coordinates are computed in increasing depth from at least +46
                05013 ; <KM> in intervals of +80 (+$0050) <KM>. Their velocity vector co
                05014 ; set to 0 <KM/H>.
                05015
    =0068       05016 L.RANGE         = $68                 ; z-coordinate of star in
    =006E       05017 L.TRAILCNT      = $6E                 ; Star's index in star tra
                05018
A9B4 C6C2       05019 INITTRAIL       DEC TRAILDELAY        ; Decrement star trail del
A9B6 1068       05020                 BPL SKIP074           ; Return if delay still co
                05021
A9B8 A901       05022                 LDA #1                ; Turn on hyperwarp veloci
A9BA 85C1       05023                 STA VELOCITYHI        ;
                05024
A9BC A930       05025                 LDA #NUMSPCOBJ.ALL-1  ; Max index of space objec
A9BE 8579       05026                 STA MAXSPCOBJIND      ;
                05027
A9C0 A903       05028                 LDA #3                ; Star trail delay := 3(+1
A9C2 85C2       05029                 STA TRAILDELAY        ;
                05030
A9C4 A6C3       05031                 LDX TRAILIND          ; Next avail. space obj in
                05032
A9C6 A912       05033                 LDA #$12              ; Star z-coordinate := >=
A9C8 8569       05034                 STA L.RANGE+1         ;
                05035
A9CA AD0AD2     05036                 LDA RANDOM            ; Calc random index to pic
A9CD 2903       05037                 AND #$03              ;
A9CF A8         05038                 TAY                   ;
A9D0 B93ABB     05039                 LDA WARPSTARXTAB,Y    ; Pick x-coordinate (high
A9D3 9D710A     05040                 STA XPOSHI,X          ;
A9D6 B93EBB     05041                 LDA WARPSTARYTAB,Y    ;
A9D9 9DA20A     05042                 STA YPOSHI,X          ; Pick y-coordinate (high
A9DC 20BEB7     05043                 JSR RNDINVXY          ; Randomize signs of x and
                05044
A9DF 8A         05045                 TXA                   ; Save space object index
A9E0 A8         05046                 TAY                   ;
A9E1 A905       05047                 LDA #5                ; Loop over 5(+1) stars th
A9E3 856E       05048                 STA L.TRAILCNT        ; Store star counter of st
                05049
A9E5 18         05050 LOOP034         CLC                   ; Place stars in z-coordin
A9E6 A568       05051                 LDA L.RANGE           ;
A9E8 6950       05052                 ADC #80               ;
A9EA 8568       05053                 STA L.RANGE           ;
A9EC 9DD30A     05054                 STA ZPOSLO,X          ;
A9EF A569       05055                 LDA L.RANGE+1         ;
A9F1 6900       05056                 ADC #0                ;
A9F3 8569       05057                 STA L.RANGE+1         ;
A9F5 9D400A     05058                 STA ZPOSHI,X          ;
                05059
A9F8 A900       05060                 LDA #0                ; Star's velocity vector c
A9FA 9D660B     05061                 STA ZVEL,X            ;
A9FD 9D970B     05062                 STA XVEL,X            ;
AA00 9DC80B     05063                 STA YVEL,X            ;
AA03 A901       05064                 LDA #1                ; Star's z-coordinate sign
AA05 9DAD09     05065                 STA ZPOSSIGN,X        ;
                05066
AA08 A963       05067                 LDA #99               ; Init pixel row and colum
AA0A 9DF90B     05068                 STA PIXELROWNEW,X     ; ...offscreen value (trig
AA0D 9D2A0C     05069                 STA PIXELCOLUMN,X     ; ...GAMELOOP's calls to S
                05070
```

```
AA10 20C1AC   05071                JSR COPYPOSXY          ; Copy x and y coordinate
              05072
AA13 CA       05073                DEX                    ; Decrement space object i
AA14 E011     05074                CPX #NUMSPCOBJ.NORM    ; If index reaches minimum
AA16 B002     05075                BCS SKIP073            ;
AA18 A230     05076                LDX #NUMSPCOBJ.ALL-1   ; ...wrap-around to maximu
AA1A C66E     05077 SKIP073        DEC L.TRAILCNT         ;
AA1C 10C7     05078                BPL LOOP034            ; Next star of star trail
              05079
AA1E 86C3     05080                STX TRAILIND           ; Save space object index
AA20 60       05081 SKIP074        RTS                    ; Return
              05082
              05083 ;****************************************************************
              05084 ;*
              05085 ;*                                PROJECTION
              05086 ;*
              05087 ;*         Calculate pixel column (or row) number from position ve
              05088 ;*
              05089 ;****************************************************************
              05090
              05091 ; Calculates the pixel column (or row) number of a position vector
              05092 ; component relative to the PLAYFIELD center by computing the pers
              05093 ; projection quotient
              05094 ;
              05095 ;     QUOTIENT := DIVIDEND / DIVISOR * 128
              05096 ;
              05097 ; with
              05098 ;
              05099 ;     DIVIDEND := ABS(x-coordinate (or y-coordinate)) / 2
              05100 ;     DIVISOR  := ABS(z-coordinate) / 2
              05101 ;
              05102 ; If the QUOTIENT is in 0..255, it is used as an index to pick the
              05103 ; (or row) number from table MAPTO80 ($0DE9), returning values in
              05104 ;
              05105 ; If the QUOTIENT is larger than 255 ("dividend overflow") or if t
              05106 ; z-coordinate = 0 ("division by zero") then the error value 255 i
              05107 ;
              05108 ; INPUT
              05109 ;
              05110 ;   X                   = Position vector index. Used values are:
              05111 ;   DIVIDEND ($6A..$6B) = Dividend (positive 16-bit value), contai
              05112 ;                        absolute value of the x (or y) coordinat
              05113 ;
              05114 ; OUTPUT
              05115 ;
              05116 ;   A = Pixel column (or row) number relative to PLAYFIELD center.
              05117 ;       are:
              05118 ;     0..80 -> Pixel number
              05119 ;     255   -> Error value indicating "dividend overflow" or "divi
              05120
    =0068     05121 L.DIVISOR       = $68                  ; Divisor (16-bit value)
    =006D     05122 L.QUOTIENT      = $6D                  ; Division result (unsigne
    =006E     05123 L.LOOPCNT       = $6E                  ; Division loop counter. U
              05124
AA21 A900     05125 PROJECTION      LDA #0                 ; Init quotient result
AA23 856D     05126                 STA L.QUOTIENT         ;
              05127
AA25 A907     05128                 LDA #7                 ; Init division loop count
AA27 856E     05129                 STA L.LOOPCNT          ;
```

```
                05130
AA29 466B       05131                   LSR DIVIDEND+1          ; DIVIDEND := x-coordinate
AA2B 666A       05132                   ROR DIVIDEND            ; (division by 2 to make B
                05133
AA2D A5D0       05134                   LDA SHIPVIEW            ; Skip if in Aft view
AA2F D00F       05135                   BNE SKIP075             ;
                05136
AA31 BD400A     05137                   LDA ZPOSHI,X            ; If in Front view -> DIVI
AA34 4A         05138                   LSR A                   ; (division by 2 to make B
AA35 8569       05139                   STA L.DIVISOR+1         ;
AA37 BDD30A     05140                   LDA ZPOSLO,X            ;
AA3A 6A         05141                   ROR A                   ;
AA3B 8568       05142                   STA L.DIVISOR           ;
AA3D 4C52AA     05143                   JMP LOOP035             ;
                05144
AA40 38         05145 SKIP075           SEC                     ; If in Aft view -> DIVISO
AA41 A900       05146                   LDA #0                  ; (division by 2 to make B
AA43 FDD30A     05147                   SBC ZPOSLO,X            ;
AA46 8568       05148                   STA L.DIVISOR           ;
AA48 A900       05149                   LDA #0                  ;
AA4A FD400A     05150                   SBC ZPOSHI,X            ;
AA4D 4A         05151                   LSR A                   ;
AA4E 8569       05152                   STA L.DIVISOR+1         ;
AA50 6668       05153                   ROR L.DIVISOR           ;
                05154
AA52 066D       05155 LOOP035           ASL L.QUOTIENT          ; QUOTIENT := DIVIDEND / D
AA54 38         05156                   SEC                     ;
AA55 A56A       05157                   LDA DIVIDEND            ;
AA57 E568       05158                   SBC L.DIVISOR           ;
AA59 A8         05159                   TAY                     ;
AA5A A56B       05160                   LDA DIVIDEND+1          ;
AA5C E569       05161                   SBC L.DIVISOR+1         ;
AA5E 9006       05162                   BCC SKIP076             ;
                05163
AA60 856B       05164                   STA DIVIDEND+1          ;
AA62 846A       05165                   STY DIVIDEND            ;
AA64 E66D       05166                   INC L.QUOTIENT          ;
                05167
AA66 066A       05168 SKIP076           ASL DIVIDEND            ;
AA68 266B       05169                   ROL DIVIDEND+1          ;
AA6A 9003       05170                   BCC SKIP077             ;
                05171
AA6C A9FF       05172                   LDA #255                ; Return 255 if division b
AA6E 60         05173                   RTS                     ;
                05174
AA6F C66E       05175 SKIP077           DEC L.LOOPCNT           ;
AA71 10DF       05176                   BPL LOOP035             ; Next division loop itera
                05177
AA73 A46D       05178                   LDY L.QUOTIENT          ; Prep with quotient
AA75 B9E90D     05179                   LDA MAPTO80,Y           ; Pick and return pixel co
AA78 60         05180 SKIP078           RTS                     ; ...relative to PLAYFIELD
                05181
                05182 ;*****************************************************************
                05183 ;*
                05184 ;*                                MANEUVER
                05185 ;*
                05186 ;*     Maneuver our starship's and Zylon photon torpedoes and Zylo
                05187 ;*
                05188 ;*****************************************************************
```

```
05189
05190 ; DESCRIPTION
05191 ;
05192 ; This subroutine maneuvers both of our starship's photon torpedoe
05193 ; Zylon photon torpedo, and the one or two Zylon ships that are si
05194 ; displayed on the screen. It also creates meteors and new Zylon s
05195 ; subroutine is executed only if our starship is not in a starbase
05196 ; hyperwarp is not engaged.
05197 ;
05198 ; BACKGROUND
05199 ;
05200 ; When a Zylon ship is initialized, a "flight pattern" is assigned
05201 ; are 3 flight patterns (0, 1, and 4) which are picked from table
05202 ; ($BF91).
05203 ;
05204 ; The flight pattern determines the maximum velocity with which a
05205 ; move along each axis of the 3D coordinate system, that is, the m
05206 ; of a velocity vector component. Velocity vector components for Z
05207 ; picked from the Zylon velocity table ZYLONVELTAB ($BF99):
05208 ;
05209 ; +-----------------+-----+-----+-----+-----+-----+-----+-----+---
05210 ; | Velocity Index  |  0  |  1  |  2  |  3  |  4  |  5  |  6  |  7
05211 ; +-----------------+-----+-----+-----+-----+-----+-----+-----+---
05212 ; | Velocity <KM/H> | +62 | +30 | +16 |  +8 |  +4 |  +2 |  +1 |  0
05213 ; +-----------------+-----+-----+-----+-----+-----+-----+-----+---
05214 ; +-----------------+-----+-----+-----+-----+-----+-----+-----+---
05215 ; | Velocity Index  |  8  |  9  |  10 |  11 |  12 |  13 |  14 |  1
05216 ; +-----------------+-----+-----+-----+-----+-----+-----+-----+---
05217 ; | Velocity <KM/H> |  0  |  -1 |  -2 |  -4 |  -8 | -16 | -30 | -6
05218 ; +-----------------+-----+-----+-----+-----+-----+-----+-----+---
05219 ;
05220 ; The index into the Zylon velocity table ZYLONVELTAB ($BF99) corr
05221 ; the maximum velocity is called the "maximum velocity index". The
05222 ; table shows the flight patterns, their maximum velocity indices,
05223 ; corresponding velocities:
05224 ;
05225 ; +----------------+------------------+------------------+
05226 ; | Flight Pattern | Maximum Velocity | Maximum Velocity |
05227 ; |                |      Index       |                  |
05228 ; +----------------+------------------+------------------+
05229 ; |       0        |         0        |    +62 <KM/H>    |
05230 ; |       0        |        15        |    -62 <KM/H>    |
05231 ; |       1        |         1        |    +30 <KM/H>    |
05232 ; |       1        |        14        |    -30 <KM/H>    |
05233 ; |       4        |         4        |     +4 <KM/H>    |
05234 ; |       4        |        11        |     -4 <KM/H>    |
05235 ; +----------------+------------------+------------------+
05236 ;
05237 ; Because flight pattern 0 produces the fastest-moving Zylon ships
05238 ; maneuver aggressively, it is called the "attack flight pattern".
05239 ;
05240 ; Each Zylon ship has a set of 3 maximum velocity indices, one for
05241 ; velocity vector components.
05242 ;
05243 ; Each Zylon ship has also one more set of 3 velocity indices, cal
05244 ; velocity indices", one for each of its velocity vector component
05245 ; used to pick the current values of the velocity vector component
05246 ; Zylon velocity table ZYLONVELTAB ($BF99).
05247 ;
```

```
05248 ; In order to maneuver Zylon ships this subroutine uses the concep
05249 ; "milestone velocity indices". By using delay timers, called "Zyl
05250 ; this subroutine gradually increases or decreases the Zylon veloc
05251 ; with every game loop iteration to eventually match the correspon
05252 ; velocity indices. By incrementing a Zylon velocity index a Zylon
05253 ; accelerates toward the negative direction of a coordinate axis.
05254 ; decrementing a Zylon velocity index a Zylon ship accelerates tow
05255 ; positive direction of a coordinate axis. If one milestone veloci
05256 ; matched or a "milestone timer" has counted down to 0, a new mile
05257 ; index is calculated and the matching of the current Zylon veloci
05258 ; with the new milestone velocity indices repeats.
05259 ;
05260 ; DETAILS
05261 ;
05262 ; For quick lookup, the following table lists the PLAYERs and what
05263 ; they represent in this subroutine:
05264 ;
05265 ; +--------+----------------------------------+
05266 ; | PLAYER |           Represents             |
05267 ; +--------+----------------------------------+
05268 ; |    0   | Zylon Ship 0                     |
05269 ; |    1   | Zylon Ship 1                     |
05270 ; |    2   | Zylon Photon Torpedo, Meteor     |
05271 ; |    3   | Our starship's Photon Torpedo 0  |
05272 ; |    4   | Our starship's Photon Torpedo 1  |
05273 ; +--------+----------------------------------+
05274 ;
05275 ; This subroutine executes the following steps:
05276 ;
05277 ; (1)  Update the x and y velocity vector components of both of ou
05278 ;      photon torpedoes 0 and 1.
05279 ;
05280 ;      The x and y velocity vector components of both of our stars
05281 ;      torpedoes 0 and 1 are only updated if they are tracking (ho
05282 ;      target.
05283 ;
05284 ;      To update the y-velocity vector components of both of our s
05285 ;      photon torpedoes 0 and 1 the PLAYER row number difference b
05286 ;      PLAYER of tracked target space object and the current locat
05287 ;      PLAYER of our starship's photon torpedo 0 is passed to subr
05288 ;      HOMINGVEL ($AECA). It returns the new y-velocity vector com
05289 ;      for both of our starship's photon torpedoes in <KM/H>. If t
05290 ;      located below our starship's photon torpedo 0 a value of 0
05291 ;      used.
05292 ;
05293 ;      NOTE: The new y-velocity vector components depend only on t
05294 ;      number of our starship's photon torpedo 0.
05295 ;
05296 ;      To update the x-velocity vector components of both of our s
05297 ;      photon torpedoes, the above calculation is repeated for the
05298 ;      numbers of each of our starship's photon torpedoes 0 and 1.
05299 ;
05300 ; (2)  Make the Zylon ships follow the rotation of our starship.
05301 ;
05302 ;      If you rotate our starship away from Zylon ships they adjus
05303 ;      such that they reappear in our starship's view.
05304 ;
05305 ;      This is achieved by 4 Zylon timers, one for each of both Zy
05306 ;      current x and y Zylon velocity indices. The Zylon timers ar
```

```
05307 ;      every game loop iteration. If any of them reach a value of
05308 ;      corresponding Zylon velocity index is incremented or decrem
05309 ;      depending on the current joystick position.
05310 ;
05311 ;      For example, if the Zylon timer for the x-velocity of Zylon
05312 ;      reaches 0 and at the same time the joystick is pushed left
05313 ;      x-Zylon velocity index of this Zylon ship is incremented. T
05314 ;      accelerates the Zylon ship toward negative x-direction ("le
05315 ;      Zylon ship follows our starship's rotation. This works in A
05316 ;      where the direction of change of the Zylon velocity index i
05317 ;      After setting the new Zylon velocity index, it is used to p
05318 ;      Zylon timer value for this Zylon velocity index:
05319 ;
05320 ;      +--------------------------------+----+----+----+----+----+
05321 ;      | Velocity Index                 |  0 |  1 |  2 |  3 |  4 |
05322 ;      +--------------------------------+----+----+----+----+----+
05323 ;      | Zylon Timer Value (Game Loops) |  0 |  2 |  4 |  6 |  8 |
05324 ;      +--------------------------------+----+----+----+----+----+
05325 ;      +--------------------------------+----+----+----+----+----+
05326 ;      | Velocity Index                 |  8 |  9 | 10 | 11 | 12 |
05327 ;      +--------------------------------+----+----+----+----+----+
05328 ;      | Zylon Timer Value (Game Loops) | 14 | 12 | 10 |  8 |  6 |
05329 ;      +--------------------------------+----+----+----+----+----+
05330 ;
05331 ; (3)  Update the x and y velocity vector components of the single
05332 ;      torpedo.
05333 ;
05334 ;      If a Zylon photon torpedo is moving toward our starship the
05335 ;      and y velocity vector components. They are picked from tabl
05336 ;      ZYLONHOMVELTAB ($BF85) and depend on the mission level. The
05337 ;      velocity vector components are always set such that the Zyl
05338 ;      torpedo is guided toward our starship.
05339 ;
05340 ; (4)  Create a meteor?
05341 ;
05342 ;      If PLAYER2, the PLAYER to represent a meteor, is either ini
05343 ;      alive, then attempt in 7 out of 8 game loop iterations to c
05344 ;      meteor.
05345 ;
05346 ;      With a probability of 2% (4:256) a new meteor is created: I
05347 ;      is set to METEOR, its position vector components to random
05348 ;      subroutine INITPOSVEC ($B764), its lifetime to 60 game loop
05349 ;      and its velocity vector components (velocities) to x-velocit
05350 ;      y-velocity: 0 <KM/H>, z-velocity: -8 <KM/H>. Then code exec
05351 ;
05352 ; (5)  Toggle Zylon ship control.
05353 ;
05354 ;      Every other game loop iteration, the game takes control of
05355 ;      the other Zylon ship.
05356 ;
05357 ; (6)  Create new Zylon ship?
05358 ;
05359 ;      If the game-controlled Zylon ship is not alive, check if bo
05360 ;      are not alive and this is an empty sector. If so, then atte
05361 ;      a meteor. Otherwise create a new Zylon ship with infinite l
05362 ;      Randomly pick its shape type from table ZYLONSHAPTAB ($BF89
05363 ;      BASESTAR, ZYLON CRUISER, or ZYLON FIGHTER) and its flight p
05364 ;      table ZYLONFLPATTAB ($BF91) (attack flight pattern 0 is alw
05365 ;      a NOVICE mission). Then set the milestone timer to 1 game l
```

```
05366 ;      and the position vector of the Zylon ship to a position of
05367 ;      +28928 (+$71**) <KM> in front of our starship. The y-coordi
05368 ;      on the value of VICINITYMASK ($C7). The x-coordinate is the
05369 ;      y-coordinate plus at least 4864..5119 ($13**) <KM>. Randoml
05370 ;      signs of the x and y coordinates.
05371 ;
05372 ; (7)  Set the current flight pattern to attack flight pattern?
05373 ;
05374 ;      The current flight pattern of the Zylon ship will change to
05375 ;      pattern if it is close enough (z-coordinate < +8192 (+$20**
05376 ;      one of the following conditions is met:
05377 ;
05378 ;      o   The Zylon ship is located behind our starship.
05379 ;
05380 ;      o   The shape of the Zylon ship is not initial and does not
05381 ;          appear as a blip in the Long-Range Scan view.
05382 ;
05383 ; (8)  Update the back-attack flag and the milestone velocity indi
05384 ;
05385 ;      The milestone timer is decremented for the game-controlled
05386 ;      this timer reaches a value of 0 the following steps are exe
05387 ;
05388 ;      o   The milestone timer is reset to a value of 120 game loo
05389 ;
05390 ;      o   The back-attack flag is updated. It determines if the g
05391 ;          Zylon ship not only attacks from the front of our stars
05392 ;          from the back. A back-attack takes place with a probabi
05393 ;          (48:256) in WARRIOR or COMMANDER missions.
05394 ;
05395 ;      o   Course corrections are prepared for the game-controlled
05396 ;          computing the new milestone vector indices, resulting i
05397 ;          vector components for this Zylon ship. The new mileston
05398 ;          indices for each velocity vector component are randomly
05399 ;          depending of the flight pattern. Recall that the Zylon
05400 ;          is changed gradually to match the milestone velocity in
05401 ;          corresponds to a maximum velocity vector component when
05402 ;          index to pick a velocity vector component from Zylon ve
05403 ;          ZYLONVELTAB ($BF99):
05404 ;
05405 ;          +----------------+----------------+------------------+
05406 ;          | Flight Pattern | New Milestone  | Maximum Velocity |
05407 ;          |                | Velocity Index | Vector Component |
05408 ;          +----------------+----------------+------------------+
05409 ;          |       0        |        0       |    +62 <KM/H>    |
05410 ;          |       0        |       15       |    -62 <KM/H>    |
05411 ;          |       1        |        1       |    +30 <KM/H>    |
05412 ;          |       1        |       14       |    -30 <KM/H>    |
05413 ;          |       4        |        4       |     +4 <KM/H>    |
05414 ;          |       4        |       11       |     -4 <KM/H>    |
05415 ;          +----------------+----------------+------------------+
05416 ;
05417 ; (9)  Update milestone velocity indices in attack flight pattern.
05418 ;
05419 ;      If a Zylon ship executes the attack flight pattern, its mil
05420 ;      velocity indices are changed depending on the current locat
05421 ;      Zylon ship as follows:
05422 ;
05423 ;      +--------------+-------------+----------------+------------
05424 ;      | x-Coordinate |  Where on   |   Milestone    |  Velocity
```

```
05425 ;      |              |   Screen    | Velocity Index |
05426 ;      +--------------+-------------+----------------+------------
05427 ;      | x <  0 <KM>  | left half   |       0        | +62 <KM/H>
05428 ;      | x >= 0 <KM>  | right half  |      15        | -62 <KM/H>
05429 ;      +--------------+-------------+----------------+------------
05430 ;      +--------------+-------------+----------------+------------
05431 ;      | y-Coordinate |  Where on   |   Milestone    |  Velocity
05432 ;      |              |   Screen    | Velocity Index |
05433 ;      +--------------+-------------+----------------+------------
05434 ;      | y <  0 <KM>  | bottom half |       0        | +62 <KM/H>
05435 ;      | y >= 0 <KM>  | top half    |      15        | -62 <KM/H>
05436 ;      +--------------+-------------+----------------+------------
05437 ;
05438 ;      Thus, with respect to its x and y coordinates, the Zylon sh
05439 ;      around the center of the Front or Aft view.
05440 ;
05441 ;      This is the behavior of the Zylon ship along the z-axis:
05442 ;
05443 ;      If the Zylon ship attacks from the front:
05444 ;
05445 ;      +--------------------------+----------------+------------+-
05446 ;      |       z-Coordinate       |   Milestone    |  Velocity  |
05447 ;      |                          | Velocity Index |            |
05448 ;      +--------------------------+----------------+------------+-
05449 ;      | z <  +2560 (+$0A00) <KM> |       0        | +62 <KM/H> |
05450 ;      | z >= +2560 (+$0A00) <KM> |      15        | -62 <KM/H> |
05451 ;      +--------------------------+----------------+------------+-
05452 ;
05453 ;      In other words, the Zylon ship accelerates into positive z-
05454 ;      (outbound) up to a distance of +2560 (+$0A00) <KM>, then re
05455 ;      course and returns back to our starship (inbound).
05456 ;
05457 ;      If the Zylon ship attacks from the back:
05458 ;
05459 ;      +--------------------------+----------------+------------+-
05460 ;      |       z-Coordinate       |   Milestone    |  Velocity  |
05461 ;      |                          | Velocity Index |            |
05462 ;      +--------------------------+----------------+------------+-
05463 ;      | z <  -2816 (-$F500) <KM> |       0        | +62 <KM/H> |
05464 ;      | z >= -2816 (-$F500) <KM> |      15        | -62 <KM/H> |
05465 ;      +--------------------------+----------------+------------+-
05466 ;
05467 ;      In other words, the Zylon ship accelerates into negative z-
05468 ;      (outbound) up to a distance of -2816 (-$(0B00)) <KM>, then
05469 ;      course and returns back to our starship (inbound).
05470 ;
05471 ; (10) Change Zylon velocity index toward milestone velocity index
05472 ;
05473 ;      Compare all 3 Zylon velocity indices of the game-controlled
05474 ;      with their corresponding milestone velocity indices. Increm
05475 ;      decrement the former to better match the latter. Use the ne
05476 ;      velocity indices to pick the current velocity values from Z
05477 ;      table ZYLONVELTAB ($BF99).
05478 ;
05479 ; (11) Launch a Zylon photon torpedo?
05480 ;
05481 ;      Prepare launching a Zylon photon torpedo if either of the f
05482 ;      conditions are met:
05483 ;
```

```
                05484 ;       o   PLAYER2 is not used as a photon torpedo
                05485 ;
                05486 ;       o   The y-coordinate of the Zylon ship is in the range of -
                05487 ;           (-$0300..+$2FF) <KM>.
                05488 ;
                05489 ;       or if
                05490 ;
                05491 ;       o   The Zylon photon torpedo is not alive
                05492 ;
                05493 ;       o   The corresponding Zylon photon torpedo delay timer has
                05494 ;           value of 0
                05495 ;
                05496 ;       o   The y-coordinate of the Zylon ship is in the range of -
                05497 ;           (-$0300..+$2FF) <KM>.
                05498 ;
                05499 ;       At this point the z-velocity vector component of the Zylon
                05500 ;       is preloaded with a value of -80 or +80 <KM/H> depending on
                05501 ;       ship being in front or behind of our starship, respectively
                05502 ;
                05503 ;       Launch a Zylon photon torpedo if both of the following cond
                05504 ;       met:
                05505 ;
                05506 ;       o   The Zylon ship is in front or behind of our starship, w
                05507 ;           exception of a Zylon ship behind our starship in a NOVI
                05508 ;           (our starship will never be shot in the back in a NOVIC
                05509 ;
                05510 ;       o   The z-coordinate of the Zylon ship (no matter if in fro
                05511 ;           our starship) is closer than 8192 ($20**) <KM>.
                05512 ;
                05513 ;       Finally, the Zylon photon torpedo is launched with a lifeti
                05514 ;       loop iterations. Its position vector is copied from the lau
                05515 ;       ship in subroutine COPYPOSVEC ($ACAF). In addition, the Zyl
                05516 ;       earmarked for the tracking computer.
                05517
    =006A       05518 L.CTRLDZYLON    = $6A                 ; Index of currently game-
                05519                                       ; Used values are:
                05520                                       ;   0 -> Control Zylon shi
                05521                                       ;   1 -> Control Zylon shi
    =0080       05522 NEG             = $80                 ; Negative sign bit for ve
                05523
AA79 A5C0       05524 MANEUVER        LDA WARPSTATE         ; Return if in starbase se
AA7B 057B       05525                 ORA ISSTARBASESECT    ;
AA7D D0F9       05526                 BNE SKIP078           ;
                05527
                05528 ;*** Update x and y velocity of both our starship's photon torpedo
AA7F A586       05529                 LDA ISTRACKING        ; Skip this if ship's torp
AA81 F030       05530                 BEQ SKIP080           ;
                05531
AA83 A689       05532                 LDX PLTRACKED         ; Load PLAYER index of tra
                05533
AA85 38         05534                 SEC                   ; Prep A := PLAYER row num
AA86 BDF90B     05535                 LDA PL0ROWNEW,X       ; ...- PLAYER row number p
AA89 EDFC0B     05536                 SBC PL3ROWNEW         ;
AA8C 9002       05537                 BCC SKIP079           ; Skip if target above our
AA8E A900       05538                 LDA #0                ; Prep A := 0
AA90 20CAAE     05539 SKIP079         JSR HOMINGVEL         ; Get y-velocity for homin
AA93 8DCB0B     05540                 STA PL3YVEL           ; Store y-velocity photon
AA96 8DCC0B     05541                 STA PL4YVEL           ; Store y-velocity photon
                05542
```

```
AA99 38       05543                 SEC                     ; Prep A := PLAYER column
AA9A AD2D0C   05544                 LDA PL3COLUMN           ; ...- PLAYER column numbe
AA9D FD2A0C   05545                 SBC PL0COLUMN,X         ;
AAA0 20CAAE   05546                 JSR HOMINGVEL           ; Get x-velocity for homin
AAA3 8D9A0B   05547                 STA PL3XVEL             ; Store x-velocity of phot
              05548
AAA6 38       05549                 SEC                     ; Prep A := PLAYER column
AAA7 AD2E0C   05550                 LDA PL4COLUMN           ; ...- PLAYER column numbe
AAAA FD2A0C   05551                 SBC PL0COLUMN,X         ;
AAAD 20CAAE   05552                 JSR HOMINGVEL           ; Get x-velocity for homin
AAB0 8D9B0B   05553                 STA PL4XVEL             ; Store x-velocity of phot
              05554
              05555 ;*** Make Zylon ships follow rotation of our starship ************
AAB3 A203     05556 SKIP080         LDX #3                  ; Loop over x and y veloci
AAB5 D6BA     05557 LOOP036         DEC ZYLONTIMX0,X        ; Decrement Zylon timer
AAB7 1027     05558                 BPL SKIP085             ; Next timer if this one s
              05559
AAB9 8A       05560                 TXA                     ; Prep joystick (x or y) v
AABA 4A       05561                 LSR A                   ;
AABB A8       05562                 TAY                     ;
AABC B9C800   05563                 LDA JOYSTICKX,Y         ;
              05564
AABF A4D0     05565                 LDY SHIPVIEW            ; Skip if in Front view
AAC1 F005     05566                 BEQ SKIP081             ;
              05567
AAC3 49FF     05568                 EOR #$FF                ; Invert joystick value (w
AAC5 18       05569                 CLC                     ; (two's-complement)
AAC6 6901     05570                 ADC #1                  ;
              05571
AAC8 18       05572 SKIP081         CLC                     ; Add joystick value to Zy
AAC9 75B4     05573                 ADC ZYLONVELINDX0,X     ;
AACB 1002     05574                 BPL SKIP082             ;
AACD A900     05575                 LDA #0                  ;
AACF C910     05576 SKIP082         CMP #16                 ; Limit new Zylon velocity
AAD1 9002     05577                 BCC SKIP083             ;
AAD3 A90F     05578                 LDA #15                 ;
AAD5 95B4     05579 SKIP083         STA ZYLONVELINDX0,X     ; ...and store new Zylon v
              05580
AAD7 C908     05581                 CMP #8                  ; Calc new Zylon timer val
AAD9 9002     05582                 BCC SKIP084             ;
AADB 490F     05583                 EOR #$0F                ;
AADD 0A       05584 SKIP084         ASL A                   ;
AADE 95BA     05585                 STA ZYLONTIMX0,X        ; ...and store new Zylon t
              05586
AAE0 CA       05587 SKIP085         DEX                     ;
AAE1 10D2     05588                 BPL LOOP036             ; Next Zylon timer
              05589
              05590 ;*** Update x and y velocity of single Zylon photon torpedo ******
AAE3 AD8E0C   05591                 LDA PL2SHAPTYPE         ; Skip if PLAYER2 not PHOT
AAE6 D01B     05592                 BNE SKIP088             ;
              05593
AAE8 A462     05594                 LDY MISSIONLEVEL        ; Depending on mission lev
AAEA B985BF   05595                 LDA ZYLONHOMVELTAB,Y    ; ...pick (initially negat
              05596
AAED AEA40A   05597                 LDX PL2YPOSHI           ; If photon torpedo in upp
AAF0 1002     05598                 BPL SKIP086             ; ...don't toggle velocity
AAF2 297F     05599                 AND #$7F                ; ...toggle velocity sign
AAF4 8DCA0B   05600 SKIP086         STA PL2YVEL             ; Store new y-velocity of
              05601
```

```
AAF7 0980     05602                 ORA #NEG                ; Restore negative sign bi
              05603
AAF9 AE730A   05604                 LDX PL2XPOSHI           ; If photon torpedo in rig
AAFC 1002     05605                 BPL SKIP087             ; ...don't toggle velocity
AAFE 297F     05606                 AND #$7F                ; ...toggle velocity sign
AB00 8D990B   05607 SKIP087         STA PL2XVEL             ; Store new x-velocity of
              05608
              05609 ;*** Create new meteor? ***************************************
AB03 A576     05610 SKIP088         LDA COUNT256            ; Attempt meteor creation
AB05 2903     05611                 AND #$03                ;
AB07 F02E     05612                 BEQ SKIP092             ;
              05613
AB09 A5E6     05614 SKIP089         LDA PL2SHAPOFF          ; If PLAYER2 shape is init
AB0B F004     05615                 BEQ SKIP090             ;
              05616
AB0D A5EB     05617                 LDA PL2LIFE             ; Return if PLAYER2 alive
AB0F D025     05618                 BNE SKIP091             ;
              05619
AB11 AD0AD2   05620 SKIP090         LDA RANDOM              ; Return in 98% (252:256)
AB14 C904     05621                 CMP #4                  ;
AB16 B01E     05622                 BCS SKIP091             ;
              05623
              05624 ;*** Create new meteor! ***************************************
AB18 A960     05625                 LDA #SHAP.METEOR        ; PLAYER2 is METEOR (shape
AB1A 8D8E0C   05626                 STA PL2SHAPTYPE         ;
AB1D A202     05627                 LDX #2                  ; Randomize position vecto
AB1F 2064B7   05628                 JSR INITPOSVEC          ;
AB22 A93C     05629                 LDA #60                 ; Meteor lifetime := 60 ga
AB24 85EB     05630                 STA PL2LIFE             ;
AB26 A988     05631                 LDA #NEG!8              ; SUMMARY:
AB28 8D680B   05632                 STA PL2ZVEL             ; x-velocity :=  0 <KM/H>
AB2B A900     05633                 LDA #0                  ; y-velocity :=  0 <KM/H>
AB2D 8D2C0C   05634                 STA PL2COLUMN           ; z-velocity := -8 <KM/H>
AB30 8D990B   05635                 STA PL2XVEL             ;
AB33 8DCA0B   05636                 STA PL2YVEL             ; PLAYER2 column number :=
AB36 60       05637 SKIP091         RTS                     ; Return
              05638
              05639 ;*** Toggle Zylon ship control ********************************
AB37 A5A7     05640 SKIP092         LDA CTRLDZYLON          ; Toggle control to the ot
AB39 4901     05641                 EOR #$01                ;
AB3B 85A7     05642                 STA CTRLDZYLON          ;
              05643
              05644 ;*** Create a new Zylon ship? **********************************
AB3D AA       05645                 TAX                     ; Save index of controlled
AB3E B5E9     05646                 LDA PL0LIFE,X           ; Skip creating Zylon ship
AB40 D042     05647                 BNE SKIP094             ;
              05648
AB42 A5E9     05649                 LDA PL0LIFE             ; If both Zylon ships are
AB44 05EA     05650                 ORA PL1LIFE             ;
AB46 2901     05651                 AND #$01                ;
AB48 A490     05652                 LDY CURRSECTOR          ; ...and this an empty sec
AB4A D9C908   05653                 CMP GCMEMMAP,Y          ;
AB4D B0BA     05654                 BCS SKIP089             ; ...attempt to create met
              05655
              05656 ;*** Create a new Zylon ship! **********************************
AB4F A9FF     05657                 LDA #255                ; Zylon ship lifetime := 2
AB51 95E9     05658                 STA PL0LIFE,X           ;
              05659
AB53 AD0AD2   05660                 LDA RANDOM              ; Pick a Zylon ship shape
```

```
AB56 2907     05661                AND #$07               ;
AB58 A8       05662                TAY                    ;
AB59 B989BF   05663                LDA ZYLONSHAPTAB,Y     ;
AB5C 9D8C0C   05664                STA PL0SHAPTYPE,X      ;
              05665
AB5F A562     05666                LDA MISSIONLEVEL       ; Init Zylon's flight patt
AB61 F003     05667                BEQ SKIP093            ;
AB63 B991BF   05668                LDA ZYLONFLPATTAB,Y    ;
AB66 95A8     05669 SKIP093        STA ZYLONFLPAT0,X      ;
              05670
AB68 A901     05671                LDA #1                 ; Zylon ship's milestone t
AB6A 95AA     05672                STA MILESTTIM0,X       ;
              05673
AB6C 9DAD09   05674                STA ZPOSSIGN,X         ; Put Zylon ship in front
AB6F AD0AD2   05675                LDA RANDOM             ;
AB72 25C7     05676                AND VICINITYMASK       ; y-coordinate (high byte)
AB74 9DA20A   05677                STA YPOSHI,X           ;
AB77 6913     05678                ADC #19                ; x-coordinate (high byte)
AB79 9D710A   05679                STA XPOSHI,X           ;
AB7C 0971     05680                ORA #$71               ; z-coordinate (high byte)
AB7E 9D400A   05681                STA ZPOSHI,X           ;
AB81 20BEB7   05682                JSR RNDINVXY           ; Randomly invert x and y
              05683
              05684 ;*** Set current flight pattern to attack flight pattern? ********
AB84 BD400A   05685 SKIP094        LDA ZPOSHI,X           ; Skip if Zylon too distan
AB87 C920     05686                CMP #$20               ;
AB89 B011     05687                BCS SKIP096            ;
              05688
AB8B BDAD09   05689                LDA ZPOSSIGN,X         ; Set attack flight patter
AB8E F008     05690                BEQ SKIP095            ;
              05691
AB90 B5E4     05692                LDA PL0SHAPOFF,X       ; Skip if Zylon shape init
AB92 F008     05693                BEQ SKIP096            ;
              05694
AB94 C929     05695                CMP #$29               ; Skip if Zylon shape is L
AB96 F004     05696                BEQ SKIP096            ;
              05697
AB98 A900     05698 SKIP095        LDA #0                 ; Set attack flight patter
AB9A 95A8     05699                STA ZYLONFLPAT0,X      ;
              05700
              05701 ;*** Update back-attack flag and milestone velocity indices ******
AB9C D6AA     05702 SKIP096        DEC MILESTTIM0,X       ; Skip if milestone timer
AB9E 1024     05703                BPL SKIP099            ;
              05704
ABA0 A978     05705                LDA #120               ; Milestone timer := 120 g
ABA2 95AA     05706                STA MILESTTIM0,X       ;
              05707
ABA4 A562     05708                LDA MISSIONLEVEL       ; Back-attack flag := 1 in
ABA6 AC0AD2   05709                LDY RANDOM             ; ...WARRIOR or COMMANDER
ABA9 C030     05710                CPY #48                ; ...              := 0 ot
ABAB 9001     05711                BCC SKIP097            ;
ABAD 4A       05712                LSR A                  ;
ABAE 4A       05713 SKIP097        LSR A                  ;
ABAF 95B8     05714                STA ISBACKATTACK0,X    ;
              05715
              05716                                       ; Loop over all 3 mileston
ABB1 B5A8     05717                LDA ZYLONFLPAT0,X      ; Set new milestone velocit
ABB3 2C0AD2   05718 LOOP037        BIT RANDOM             ; If Zylon flight pattern
ABB6 1002     05719                BPL SKIP098            ; ...0 -> milestone velocit
```

```
ABB8 490F     05720                 EOR #$0F                ; ...1 -> milestone veloci
ABBA 95AC     05721 SKIP098         STA MILESTVELINDZ0,X    ; ...4 -> milestone veloci
ABBC E8       05722                 INX                     ;
ABBD E8       05723                 INX                     ;
ABBE E006     05724                 CPX #6                  ;
ABC0 90F1     05725                 BCC LOOP037             ; Next Zylon milestone vel
              05726
              05727 ;*** Update milestone velocity indices in attack flight pattern **
ABC2 A6A7     05728                 LDX CTRLDZYLON          ; Reload index of controll
              05729
ABC4 B5A8     05730 SKIP099         LDA ZYLONFLPAT0,X       ; Skip if not in attack fl
ABC6 D032     05731                 BNE SKIP105             ;
              05732
ABC8 A4A7     05733                 LDY CTRLDZYLON          ; Reload index of controll
              05734
              05735                                         ; Loop over all 3 mileston
ABCA C031     05736 LOOP038         CPY #$31                ; Skip to handle x and y v
ABCC B013     05737                 BCS SKIP101             ;
              05738                                         ; SUMMARY:
ABCE B9B800   05739                 LDA ISBACKATTACK0,Y     ; Handle z-velocity index:
ABD1 4A       05740                 LSR A                   ;
ABD2 B9400A   05741                 LDA ZPOSHI,Y            ; If Zylon attacks from fr
ABD5 B006     05742                 BCS SKIP100             ; z <  $0A00 <KM> -> mil v
ABD7 C90A     05743                 CMP #$0A                ; z >= $0A00 <KM> -> mil v
ABD9 900E     05744                 BCC SKIP103             ;
ABDB B004     05745                 BCS SKIP101             ; If Zylon attacks from ba
ABDD C9F5     05746 SKIP100         CMP #$F5                ; z >= $F500 <KM> -> mil v
ABDF B004     05747                 BCS SKIP102             ; z <  $F500 <KM> -> mil v
              05748
ABE1 B9AD09   05749 SKIP101         LDA ZPOSSIGN,Y          ; Handle x and y velocity
ABE4 4A       05750                 LSR A                   ;
ABE5 A90F     05751 SKIP102         LDA #15                 ; x >= 0 <KM> -> mil vel i
ABE7 B002     05752                 BCS SKIP104             ; x <  0 <KM> -> mil vel i
ABE9 A900     05753 SKIP103         LDA #0                  ; y >= 0 <KM> -> mil vel i
ABEB 95AC     05754 SKIP104         STA MILESTVELINDZ0,X    ; y <  0 <KM> -> mil vel i
              05755
ABED 18       05756                 CLC                     ; Adjust position vector c
ABEE 98       05757                 TYA                     ;
ABEF 6931     05758                 ADC #NUMSPCOBJ.ALL      ;
ABF1 A8       05759                 TAY                     ;
              05760
ABF2 E8       05761                 INX                     ;
ABF3 E8       05762                 INX                     ;
ABF4 E006     05763                 CPX #6                  ;
ABF6 90D2     05764                 BCC LOOP038             ; Next milestone velocity
              05765
              05766 ;*** Acceleration: Change Zylon velocity index toward milestone ve
ABF8 A6A7     05767                 LDX CTRLDZYLON          ; Reload index of controll
ABFA A4A7     05768 SKIP105         LDY CTRLDZYLON          ; Reload index of controll
              05769
              05770                                         ; Loop over all 3 mileston
ABFC B5B2     05771 LOOP039         LDA ZYLONVELINDZ0,X     ; Compare Zylon velocity i
ABFE D5AC     05772                 CMP MILESTVELINDZ0,X    ;
AC00 F008     05773                 BEQ SKIP107             ; Skip if equal
AC02 B004     05774                 BCS SKIP106             ;
AC04 F6B2     05775                 INC ZYLONVELINDZ0,X     ; Increm. Zylon velocity i
AC06 9002     05776                 BCC SKIP107             ;
AC08 D6B2     05777 SKIP106         DEC ZYLONVELINDZ0,X     ; Decrem. Zylon velocity i
              05778
```

```
AC0A 866A     05779 SKIP107         STX L.CTRLDZYLON        ; Save index of controlled
AC0C AA       05780                 TAX                     ;
AC0D BD99BF   05781                 LDA ZYLONVELTAB,X       ; Pick new velocity value
AC10 A66A     05782                 LDX L.CTRLDZYLON        ; Reload index of controll
AC12 99660B   05783                 STA ZVEL,Y              ; Store new velocity vecto
              05784
AC15 98       05785                 TYA                     ; Next velocity vector com
AC16 18       05786                 CLC                     ;
AC17 6931     05787                 ADC #NUMSPCOBJ.ALL      ;
AC19 A8       05788                 TAY                     ;
              05789
AC1A E8       05790                 INX                     ;
AC1B E8       05791                 INX                     ;
AC1C E006     05792                 CPX #6                  ;
AC1E 90DC     05793                 BCC LOOP039             ; Next milestone velocity
              05794
              05795 ;*** Launch Zylon photon torpedo? ******************************
              05796
              05797 ;*** Check PLAYER2 shape and lifetime ***************************
AC20 A6A7     05798                 LDX CTRLDZYLON          ; Reload index of controll
              05799
AC22 AD8E0C   05800                 LDA PL2SHAPTYPE         ; Skip if PLAYER2 not PHOT
AC25 D00B     05801                 BNE SKIP109             ;
              05802
AC27 A5EB     05803                 LDA PL2LIFE             ; Return if Zylon photon t
AC29 D006     05804                 BNE SKIP108             ;
              05805
AC2B A5BE     05806                 LDA TORPEDODELAY        ; Count down Zylon photon
AC2D F003     05807                 BEQ SKIP109             ; ...before launching next
AC2F C6BE     05808                 DEC TORPEDODELAY        ;
AC31 60       05809 SKIP108         RTS                     ; Return
              05810
              05811 ;*** Check y-coordinate of Zylon ship ***************************
AC32 18       05812 SKIP109         CLC                     ; Return if Zylon ship's y
AC33 BDA20A   05813                 LDA YPOSHI,X            ; ...in -768..+767 (-$(030
AC36 6902     05814                 ADC #2                  ;
AC38 C905     05815                 CMP #5                  ;
AC3A B0F5     05816                 BCS SKIP108             ;
              05817
              05818 ;*** Set Zylon photon torpedo's z-velocity **********************
AC3C A0D0     05819                 LDY #NEG!80             ; Prep Zylon torpedo's z-v
              05820
AC3E BDAD09   05821                 LDA ZPOSSIGN,X          ; Prep Zylon ship's sign o
AC41 4A       05822                 LSR A                   ;
AC42 BD400A   05823                 LDA ZPOSHI,X            ; Prep Zylon ship's z-coor
AC45 B008     05824                 BCS SKIP110             ; Skip if Zylon ship in fr
AC47 49FF     05825                 EOR #$FF                ; ...else invert loaded Zy
              05826
AC49 A462     05827                 LDY MISSIONLEVEL        ; Return (no torpedo from
AC4B F0E4     05828                 BEQ SKIP108             ;
              05829
AC4D A050     05830                 LDY #80                 ; Preload Zylon torpedo's
              05831
              05832 ;*** Is Zylon ship in range? ***********************************
AC4F C920     05833 SKIP110         CMP #$20                ; Return if Zylon ship too
AC51 B0DE     05834                 BCS SKIP108             ; ... (ABS(z-coordinate) >
              05835
AC53 8C680B   05836                 STY PL2ZVEL             ; Store Zylon photon torpe
              05837
```

```
                05838 ;*** Launch Zylon photon torpedo! ******************************
                05839
AC56 A900       05840                 LDA #0                  ; PLAYER2 is PHOTON TORPED
AC58 8D8E0C     05841                 STA PL2SHAPTYPE         ;
AC5B 8D2C0C     05842                 STA PL2COLUMN           ; Zylon torpedo PLAYER col
AC5E A93E       05843                 LDA #62                 ;
AC60 85EB       05844                 STA PL2LIFE             ; Zylon torpedo lifetime :
                05845
AC62 A202       05846                 LDX #2                  ; Prep source index for po
AC64 A4A7       05847                 LDY CTRLDZYLON          ; Prep destination index f
AC66 84BF       05848                 STY ZYLONATTACKER       ; Save Zylon ship index fo
AC68 4CAFAC     05849                 JMP COPYPOSVEC          ; Copy position vector fro
                05850
                05851 ;*****************************************************************
                05852 ;*
                05853 ;*                                  INITEXPL
                05854 ;*
                05855 ;*                             Initialize explosion
                05856 ;*
                05857 ;*****************************************************************
                05858
                05859 ; DESCRIPTION
                05860 ;
                05861 ; Initializes the explosion's lifetime, the explosion fragments' p
                05862 ; velocity vectors as well as their pixel row and column numbers.
                05863 ;
                05864 ; An explosion has a lifetime of 128 game loop iterations. It cons
                05865 ; explosion fragment space objects with indices 17..48. The positi
                05866 ; each explosion fragment is copied from the exploding PLAYER spac
                05867 ;
                05868 ; The pixel column number of each explosion fragment is initialize
                05869 ;
                05870 ;     PIXEL COLUMN NUMBER := PLAYER column number - 48 + RND(0..15
                05871 ;
                05872 ; To convert PLAYER column numbers (in Player/Missile (PM) pixels)
                05873 ; column numbers, the PLAYER column number of the left PLAYFIELD b
                05874 ; is subtracted and a random number is added.
                05875 ;
                05876 ; BUG (at $AC76): The added random number should not be in 0..15 b
                05877 ; because the exploding PLAYER is 8 pixels wide. The PLAYER column
                05878 ; represents the left edge of the PLAYER shape. When using a rando
                05879 ; 0..15, half of the pixels are located off to the right of the PL
                05880 ; the PLAYER area. Suggested fix: Replace instruction AND #$0F wit
                05881 ;
                05882 ; The pixel row number of each explosion fragment is initialized t
                05883 ;
                05884 ;     PIXEL ROW NUMBER := (PLAYER row number - RND(0..15)) / 2 - 1
                05885 ;
                05886 ; BUG (at $AC88): To convert PLAYER row numbers (in PM pixels) int
                05887 ; numbers, the PLAYER row number to the top PLAYFIELD border (= 16
                05888 ; subtracted first, then the division by 2 (instruction LRS A) sho
                05889 ; to reduce the double-line PM resolution to the single-line PLAYF
                05890 ; resolution. Suggested fix: Swap instruction LRS A with SBC #16 w
                05891 ; the following formula for the pixel row number:
                05892 ;
                05893 ;     PIXEL ROW NUMBER := (PLAYER row number - 16 + RND(0..15)) /
                05894 ;
                05895 ; Incidentally, adding a random number in 0..15 is correct. PLAYER
                05896 ; represents the top edge of the PLAYER shape, which is typically
```

```
                05897 ; tall when representing a close space object.
                05898 ;
                05899 ; The velocity vector of explosion fragments is set to random x, y
                05900 ; velocity vector components in -7..+7 <KM/H>.
                05901 ;
                05902 ; INPUT
                05903 ;
                05904 ;   Y = PLAYER index from which the explosion originates. Used val
                05905 ;     0 -> Explosion of PLAYER0 (Zylon ship 0)
                05906 ;     1 -> Explosion of PLAYER1 (Zylon ship 1)
                05907 ;     2 -> Explosion of PLAYER2 (Zylon photon torpedo, starbase, o
                05908
AC6B A980       05909 INITEXPL        LDA #128                ; Explosion lifetime := 12
AC6D 8573       05910                 STA EXPLLIFE            ;
                05911
AC6F A230       05912                 LDX #NUMSPCOBJ.ALL-1    ; Max index of space objec
AC71 8679       05913                 STX MAXSPCOBJIND        ;
                05914
                05915                                         ; Loop over all explosion
                05916                                         ; (index 48..17)
AC73 AD0AD2     05917 LOOP040         LDA RANDOM              ; PIXEL COLUMN NUM := PLAY
AC76 290F       05918                 AND #$0F                ; (!)
AC78 792A0C     05919                 ADC PL0COLUMN,Y         ;
AC7B E930       05920                 SBC #48                 ;
AC7D 9D2A0C     05921                 STA PIXELCOLUMN,X       ;
                05922
AC80 AD0AD2     05923                 LDA RANDOM              ; PIXEL ROW NUM := (PLAYER
AC83 290F       05924                 AND #$0F                ;
AC85 79F90B     05925                 ADC PL0ROWNEW,Y         ;
AC88 4A         05926                 LSR A                   ; (!)
AC89 E910       05927                 SBC #16                 ;
AC8B 9DF90B     05928                 STA PIXELROWNEW,X       ;
                05929
AC8E 20AFAC     05930                 JSR COPYPOSVEC          ; Copy position vector of
                05931
AC91 AD0AD2     05932                 LDA RANDOM              ; z-velocity := RND(-7..+7
AC94 2987       05933                 AND #NEG!7              ;
AC96 9D660B     05934                 STA ZVEL,X              ;
AC99 AD0AD2     05935                 LDA RANDOM              ; x-velocity := RND(-7..+7
AC9C 2987       05936                 AND #NEG!7              ;
AC9E 9D970B     05937                 STA XVEL,X              ;
ACA1 AD0AD2     05938                 LDA RANDOM              ; y-velocity := RND(-7..+7
ACA4 2987       05939                 AND #NEG!7              ;
ACA6 9DC80B     05940                 STA YVEL,X              ;
                05941
ACA9 CA         05942                 DEX                     ; Next explosion fragment
ACAA E010       05943                 CPX #16                 ;
ACAC D0C5       05944                 BNE LOOP040             ;
ACAE 60         05945                 RTS                     ; Return
                05946
                05947 ;*****************************************************************
                05948 ;*
                05949 ;*                                COPYPOSVEC
                05950 ;*
                05951 ;*                           Copy a position vector
                05952 ;*
                05953 ;*****************************************************************
                05954
                05955 ; DESCRIPTION
```

```
                05956 ;
                05957 ; Copies a position vector.
                05958 ;
                05959 ; Actually, this subroutine copies the z-coordinate only, then cod
                05960 ; continues into subroutine COPYPOSXY ($ACC1) to copy the x and y
                05961 ;
                05962 ; INPUT
                05963 ;
                05964 ;   X = Destination position vector index. Used values are: 0..48.
                05965 ;   Y = Source position vector index. Used values are: 0..48.
                05966
ACAF B9AD09     05967 COPYPOSVEC      LDA ZPOSSIGN,Y          ;
ACB2 9DAD09     05968                 STA ZPOSSIGN,X          ;
ACB5 B9400A     05969                 LDA ZPOSHI,Y            ;
ACB8 9D400A     05970                 STA ZPOSHI,X            ;
ACBB B9D30A     05971                 LDA ZPOSLO,Y            ;
ACBE 9DD30A     05972                 STA ZPOSLO,X            ;
                05973
                05974 ;*****************************************************************
                05975 ;*
                05976 ;*                                    COPYPOSXY
                05977 ;*
                05978 ;*          Copy x and y components (coordinates) of position vect
                05979 ;*
                05980 ;*****************************************************************
                05981
                05982 ; DESCRIPTION
                05983 ;
                05984 ; Copies the x and y components (coordinates) of a position vector
                05985 ;
                05986 ; INPUT
                05987 ;
                05988 ;   X = Destination position vector index. Used values are: 0..48.
                05989 ;   Y = Source position vector index. Used values are: 0..48.
                05990
ACC1 B9DE09     05991 COPYPOSXY       LDA XPOSSIGN,Y          ;
ACC4 9DDE09     05992                 STA XPOSSIGN,X          ;
ACC7 B9710A     05993                 LDA XPOSHI,Y            ;
ACCA 9D710A     05994                 STA XPOSHI,X            ;
ACCD B90F0A     05995                 LDA YPOSSIGN,Y          ;
ACD0 9D0F0A     05996                 STA YPOSSIGN,X          ;
ACD3 B9A20A     05997                 LDA YPOSHI,Y            ;
ACD6 9DA20A     05998                 STA YPOSHI,X            ;
ACD9 B9040B     05999                 LDA XPOSLO,Y            ;
ACDC 9D040B     06000                 STA XPOSLO,X            ;
ACDF B9350B     06001                 LDA YPOSLO,Y            ;
ACE2 9D350B     06002                 STA YPOSLO,X            ;
ACE5 60         06003 SKIP111         RTS                     ; Return
                06004
                06005 ;*****************************************************************
                06006 ;*
                06007 ;*                                    DOCKING
                06008 ;*
                06009 ;*      Handle docking at starbase, launch and return of transfer
                06010 ;*
                06011 ;*****************************************************************
                06012
                06013 ; DESCRIPTION
                06014 ;
```

```
06015 ; Handles docking at a starbase, launching and returning the trans
06016 ; and repairing our starship's subsystems.
06017 ;
06018 ; This subroutine changes, if in Front view, the PLAYER-PLAYFIELD
06019 ; that PLAYERs like the starbase appear behind the cross hairs, wh
06020 ; of the PLAYFIELD.
06021 ;
06022 ; BUG (at $ACEE): In Front view, the specific order of PLAYERs (PL
06023 ; PLAYFIELD colors (PF0..4) is, from front to back:
06024 ;
06025 ;     PL4 > PF0, PF1, PF2 > PL0 > PL1 > PL2 > PL3 > PF4 (BGR)
06026 ;
06027 ; This makes the starbase appear behind the cross hairs, but also
06028 ; stars, as both cross hairs and stars are part of the PLAYFIELD -
06029 ; noticed glitch.
06030 ;
06031 ; Note also that, as an exception of the rule, PLAYER4 (transfer v
06032 ; displayed before the PLAYFIELD. Thus, the transfer vessel appear
06033 ; the cross hairs!
06034 ;
06035 ; In Aft view, the arrangement is reversed: PLAYERs are arranged i
06036 ; PLAYFIELD. The specific order of PLAYERs (PL0..4) and PLAYFIELD
06037 ; (PF0..4) is, from front to back:
06038 ;
06039 ;     PL0 > PL1 > PL2 > PL3 > PL4 > PF0, PF1, PF2 > PF4 (BGR)
06040 ;
06041 ; In this case, both the starbase and the transfer vessel appear i
06042 ; cross hairs! Suggested fix: None, technically not possible.
06043 ;
06044 ;
06045 ; The starbase is tracked and the PLAYER0..2 shape types are set t
06046 ; RIGHT, STARBASE LEFT, and STARBASE CENTER, respectively, combini
06047 ; 3-part starbase shape.
06048 ;
06049 ; If this sector is still marked as a starbase sector but no more
06050 ; Galactic Chart (if in the meantime either Zylon units have surro
06051 ; sector and destroyed the starbase or you have destroyed the star
06052 ; photon torpedo) then the noise sound pattern SHIELD EXPLOSION is
06053 ; subroutine NOISE ($AEA8) and code execution returns.
06054 ;
06055 ; Otherwise a minimum distance to the starbase of +32 (+$0020) <KM
06056 ; and the conditions for a successful docking are checked:
06057 ;
06058 ; DOCKING CONDITIONS
06059 ;
06060 ; A docking is successful if all of the following conditions are m
06061 ;
06062 ; (1)  The PLAYER2 (STARBASE CENTER) column number is in 120..135.
06063 ;
06064 ;      BUG (at $AD39): At first glance, the PLAYER column interval
06065 ;      corresponds to an almost symmetric interval of -8..+7 PM pi
06066 ;      to the horizontal center of the PLAYFIELD, at PLAYER column
06067 ;      (48 PM pixels offset to left PLAYFIELD border + 80 PM pixel
06068 ;      PLAYFIELD center). This is correct only if the PLAYER colum
06069 ;      to designate the horizontal center of the PLAYER. However i
06070 ;      its left edge! Thus the used pixel column number range 120.
06071 ;      an asymmetric horizontal docking position: A docking is suc
06072 ;      horizontal position of the starbase shape's center is rough
06073 ;      pixels relative to the horizontal center of the PLAYFIELD.
```

```
06074 ;      Replace SBC #120 with SBC #117. This leads to an interval o
06075 ;      pixels relative to the horizontal center of the PLAYFIELD a
06076 ;      symmetry in the horizontal docking position.
06077 ;
06078 ; (2)  The PLAYER2 (STARBASE CENTER) row number is in 104..119.
06079 ;
06080 ;      BUG (at $AD43): The PLAYER row interval of 104..119 corresp
06081 ;      asymmetric interval of -20..-5 PM pixels relative to the ve
06082 ;      of the PLAYFIELD, at pixel row number 80 or PLAYER row numb
06083 ;      lets you dock at a starbase that "sits" on top of the horiz
06084 ;      hairs but not at one that "hangs" from them. Suggested fix:
06085 ;      #104 with SBC #108. This leads to an interval of -8..+7 pix
06086 ;      to the vertical center of the PLAYFIELD (assuming a PLAYER2
06087 ;      pixel height, which is typical during docking) and better s
06088 ;      vertical docking position.
06089 ;
06090 ; (3)  The starbase is in correct distance in front of our starshi
06091 ;      starbase's z-coordinate must be < +512 (+$02**) <KM>.
06092 ;
06093 ; (4)  Our starship is horizontally level with the starbase: The s
06094 ;      y-coordinate must be < +256 (+$01**) <KM>.
06095 ;
06096 ; (5)  Our starship is at a complete halt.
06097 ;
06098 ; DOCKING SUCCESSFUL
06099 ;
06100 ; If the conditions for a successful docking are met, the subseque
06101 ; transfer operation can be divided in the following states, start
06102 ; NOT DOCKED:
06103 ;
06104 ; (1)  NOT DOCKED
06105 ;
06106 ;      The docking state is set to ORBIT ESTABLISHED and the title
06107 ;      updated with "ORBIT ESTABLISHED".
06108 ;
06109 ; (2)  ORBIT ESTABLISHED
06110 ;
06111 ;      After waiting until the title line "ORBIT ESTABLISHED" has
06112 ;      the transfer vessel is initialized and launched: The PLAYER
06113 ;      is set to TRANSFER VESSEL. Its position vector is set to a
06114 ;      and in front of our starship, but behind the starbase:
06115 ;
06116 ;          x-coordinate :=     +0..+255 (+$00**) <KM>
06117 ;          y-coordinate :=   +256..+511 (+$01**) <KM>
06118 ;          z-coordinate := +4096..+4351 (+$10**) <KM>
06119 ;
06120 ;      Its velocity vector is set to
06121 ;
06122 ;          x-velocity   := +1 <KM/H>
06123 ;          y-velocity   := -1 <KM/H>
06124 ;          z-velocity   := -7 <KM/H>
06125 ;
06126 ;      This will move the transfer vessel from behind the starbase
06127 ;      direction toward and a little to the lower right of our sta
06128 ;      lifetime of the transfer vessel (and its return journey) is
06129 ;      game loop iterations. Finally, the docking state is set to
06130 ;      TRANSFER VESSEL.
06131 ;
06132 ; (3)  RETURN TRANSFER VESSEL
```

```
                06133 ;
                06134 ;      After checking if the transfer vessel has passed behind our
                06135 ;      beeper sound pattern ACKNOWLEDGE is played in subroutine BE
                06136 ;      the title line is updated with "TRANSFER COMPLETE", our sta
                06137 ;      subsystems are repaired, and our starship's ENERGY readout
                06138 ;      9999 energy units. by inverting the z-velocity the velocity
                06139 ;      transfer vessel is changed to
                06140 ;
                06141 ;          x-velocity   := +1 <KM/H>
                06142 ;          y-velocity   := -1 <KM/H>
                06143 ;          z-velocity   := +7 <KM/H>
                06144 ;
                06145 ;      thus launching the transfer vessel on its return journey to
                06146 ;      The docking state is set to TRANSFER COMPLETE. Finally, the
                06147 ;      updated in subroutine UPDSCREEN ($B07B).
                06148 ;
                06149 ; (4)  TRANSFER COMPLETE
                06150 ;
                06151 ;      This docking state marks the end of a successful docking an
                06152 ;      operation.
                06153 ;
                06154 ; DOCKING ABORTED
                06155 ;
                06156 ; If the docking conditions above are not met and the docking stat
                06157 ; ORBIT ESTABLISHED or RETURN TRANSFER VESSEL then the message "DO
                06158 ; is displayed and the docking state is set to NOT DOCKED.
                06159
ACE6 A57B       06160 DOCKING         LDA ISSTARBASESECT      ; Return if not in starbas
ACE8 F0FB       06161                 BEQ SKIP111             ;
                06162
ACEA A5D0       06163                 LDA SHIPVIEW            ; Skip if not in Front vie
ACEC D005       06164                 BNE SKIP112             ;
ACEE A914       06165                 LDA #$14                ; GTIA: Enable PLAYER4, pr
ACF0 8D1BD0     06166                 STA PRIOR               ; (Cross hairs in front of
                06167
ACF3 A902       06168 SKIP112         LDA #2                  ; Track starbase (PLAYER2)
ACF5 8D5C09     06169                 STA TRACKDIGIT          ;
                06170
                06171 ;** Initialize starbase shape ************************************
ACF8 A930       06172                 LDA #SHAP.STARBASEC     ; PLAYER2 is STARBASE CENT
ACFA 8D8E0C     06173                 STA PL2SHAPTYPE         ;
ACFD A920       06174                 LDA #SHAP.STARBASEL     ; PLAYER1 is STARBASE LEFT
ACFF 8D8D0C     06175                 STA PL1SHAPTYPE         ;
AD02 A940       06176                 LDA #SHAP.STARBASER     ; PLAYER0 is STARBASE RIGH
AD04 8D8C0C     06177                 STA PL0SHAPTYPE         ;
                06178
AD07 A9FF       06179                 LDA #255                ; Prep starbase lifetime :
                06180
AD09 A690       06181                 LDX CURRSECTOR          ; Skip if starbase in curr
AD0B BCC908     06182                 LDY GCMEMMAP,X          ;
AD0E 3002       06183                 BMI SKIP113             ;
                06184
AD10 A900       06185                 LDA #0                  ; Prep starbase lifetime :
                06186
AD12 85E9       06187 SKIP113         STA PL0LIFE             ; PLAYER0 lifetime := eith
AD14 85EA       06188                 STA PL1LIFE             ; PLAYER1 lifetime := eith
AD16 85EB       06189                 STA PL2LIFE             ; PLAYER2 lifetime := eith
AD18 857B       06190                 STA ISSTARBASESECT      ; Store starbase-in-sector
AD1A 300A       06191                 BMI SKIP114             ; Skip if starbase in curr
```

```
                06192
AD1C A002       06193                   LDY #2                  ; Init explosion at PLAYER
AD1E 206BAC     06194                   JSR INITEXPL            ;
                06195
AD21 A20A       06196                   LDX #$0A                ; Play noise sound pattern
AD23 4CA8AE     06197                   JMP NOISE               ;
                06198
                06199 ;*** Keep minimum distance to starbase ***************************
AD26 AD420A     06200 SKIP114           LDA PL2ZPOSHI           ; Skip if starbase z-coord
AD29 D00A       06201                   BNE SKIP115             ;
                06202
AD2B ADD50A     06203                   LDA PL2ZPOSLO           ; Approach starbase not cl
AD2E C920       06204                   CMP #32                 ;
AD30 B003       06205                   BCS SKIP115             ;
AD32 EED50A     06206                   INC PL2ZPOSLO           ; ...else push starbase ba
                06207
                06208 ;*** Check if in docking range ***********************************
AD35 AD2C0C     06209 SKIP115           LDA PL2COLUMN           ; Abort docking if PLAYER
AD38 38         06210                   SEC                     ; ...PLAYER2 (STARBASE CEN
AD39 E978       06211                   SBC #120                ; (!)
AD3B C910       06212                   CMP #16                 ;
AD3D B022       06213                   BCS SKIP116             ;
                06214
AD3F ADFB0B     06215                   LDA PL2ROWNEW           ; Abort docking if PLAYER
AD42 38         06216                   SEC                     ; ...PLAYER2 (STARBASE CEN
AD43 E968       06217                   SBC #104                ; (!)
AD45 C910       06218                   CMP #16                 ;
AD47 B018       06219                   BCS SKIP116             ;
                06220
AD49 AD420A     06221                   LDA PL2ZPOSHI           ; Abort docking if...
AD4C C902       06222                   CMP #2                  ; ... z-coordinate of star
AD4E B011       06223                   BCS SKIP116             ;
                06224
AD50 ADAF09     06225                   LDA PL2ZPOSSIGN         ; Abort docking...
AD53 2D110A     06226                   AND PL2YPOSSIGN         ; ...if starbase not in fr
AD56 4901       06227                   EOR #$01                ;
AD58 0570       06228                   ORA VELOCITYLO          ; ...if our starship's vel
AD5A 0DA40A     06229                   ORA PL2YPOSHI           ; ...if starbase not rough
AD5D 0571       06230                   ORA NEWVELOCITY         ; ...if our starship's new
AD5F F010       06231                   BEQ SKIP119             ; Else skip and handle doc
                06232
                06233 ;*** Docking aborted ********************************************
AD61 A575       06234 SKIP116           LDA DOCKSTATE           ; Skip if DOCKSTATE is NOT
AD63 C902       06235                   CMP #2                  ;
AD65 9005       06236                   BCC SKIP117             ;
                06237
AD67 A01F       06238                   LDY #$1F                ; Set title phrase "DOCKIN
AD69 2023B2     06239                   JSR SETTITLE            ;
                06240
AD6C A900       06241 SKIP117           LDA #0                  ; DOCKSTATE := NOT DOCKED
AD6E 8575       06242                   STA DOCKSTATE           ;
AD70 60         06243 SKIP118           RTS                     ; Return
                06244
                06245 ;*** Docking successful, check docking state ********************
AD71 2475       06246 SKIP119           BIT DOCKSTATE           ; Check DOCKSTATE
AD73 700D       06247                   BVS SKIP120             ; If DOCKSTATE = ORBIT EST
AD75 3042       06248                   BMI SKIP122             ; If DOCKSTATE = RETURN TR
AD77 A575       06249                   LDA DOCKSTATE           ;
AD79 D0F5       06250                   BNE SKIP118             ; Return if DOCKSTATE not
```

```
AD7B C675     06251                DEC DOCKSTATE          ; DOCKSTATE := ORBIT ESTAB
              06252
AD7D A01C     06253                LDY #$1C               ; Set title phrase "ORBIT
AD7F 4C23B2   06254                JMP SETTITLE           ;
              06255
              06256 ;*** Orbit established ***************************************
AD82 A200     06257 SKIP120        LDX #0                 ; Enqueue new, empty title
AD84 8665     06258                STX NEWTITLEPHR        ;
              06259
AD86 A4D1     06260                LDY TITLEPHR           ; Return if "ORBIT ESTABLI
AD88 D0E6     06261                BNE SKIP118            ;
              06262
              06263 ;*** Launch transfer vessel ************************************
AD8A A950     06264                LDA #SHAP.TRANSVSSL    ; PLAYER4 is TRANSFER VESS
AD8C 8D900C   06265                STA PL4SHAPTYPE        ;
              06266
AD8F A901     06267                LDA #1                 ; Place transfer vessel be
AD91 8DB109   06268                STA PL4ZPOSSIGN        ; x-coordinate :=    +0..+
AD94 8DE209   06269                STA PL4XPOSSIGN        ; y-coordinate :=  +256..+
AD97 8D130A   06270                STA PL4YPOSSIGN        ; z-coordinate := +4096..+
AD9A 8DA60A   06271                STA PL4YPOSHI          ;
AD9D 8D9B0B   06272                STA PL4XVEL            ; Move transfer vessel tow
ADA0 A910     06273                LDA #$10               ; x-velocity := +1 <KM/H>
ADA2 8D440A   06274                STA PL4ZPOSHI          ; y-velocity := -1 <KM/H>
ADA5 A900     06275                LDA #$00               ; z-velocity := -7 <KM/H>
ADA7 8D750A   06276                STA PL4XPOSHI          ;
ADAA A987     06277                LDA #NEG!7             ;
ADAC 8D6A0B   06278                STA PL4ZVEL            ;
ADAF A981     06279                LDA #NEG!1             ; DOCKSTATE := RETURN TRAN
ADB1 8575     06280                STA DOCKSTATE          ;
ADB3 8DCC0B   06281                STA PL4YVEL            ;
ADB6 85ED     06282                STA PL4LIFE            ; Transfer vessel lifetime
ADB8 60       06283 SKIP121        RTS                    ; Return
              06284
              06285 ;*** Return transfer vessel ************************************
ADB9 ADB109   06286 SKIP122        LDA PL4ZPOSSIGN        ; Return if transfer vesse
ADBC D0FA     06287                BNE SKIP121            ;
              06288
ADBE A20C     06289                LDX #$0C               ; Play beeper sound patter
ADC0 20A6B3   06290                JSR BEEP               ;
              06291
ADC3 A021     06292                LDY #$21               ; Set title phrase "TRANSF
ADC5 2023B2   06293                JSR SETTITLE           ;
              06294
ADC8 A205     06295                LDX #5                 ; Repair all 6 subsystems
ADCA BD8BBB   06296 LOOP041        LDA PANELTXTTAB+73,X   ;
ADCD 9D9209   06297                STA GCSTATPHO,X        ;
ADD0 CA       06298                DEX                    ;
ADD1 10F7     06299                BPL LOOP041            ;
              06300
ADD3 A989     06301                LDA #CCS.COL2!CCS.9    ; Set starship's ENERGY re
ADD5 A203     06302                LDX #3                 ;
ADD7 9D5509   06303 LOOP042        STA ENERGYD1,X         ;
ADDA CA       06304                DEX                    ;
ADDB 10FA     06305                BPL LOOP042            ;
              06306
ADDD A907     06307                LDA #7                 ; Move transfer vessel bac
ADDF 8D6A0B   06308                STA PL4ZVEL            ; x-velocity := -1 <KM/H>
ADE2 A981     06309                LDA #NEG!1             ; y-velocity := +1 <KM/H>
```

```
ADE4 8D9B0B   06310                STA PL4XVEL            ; z-velocity := +7 <KM/H>
ADE7 A901     06311                LDA #1                 ;
ADE9 8DCC0B   06312                STA PL4YVEL            ;
              06313
ADEC 8575     06314                STA DOCKSTATE          ; DOCKSTATE := TRANSFER CO
ADEE 4C7BB0   06315                JMP UPDSCREEN          ; Update screen and return
              06316
              06317 ;*****************************************************************
              06318 ;*
              06319 ;*                                  MODDLST
              06320 ;*
              06321 ;*                             Modify Display List
              06322 ;*
              06323 ;*****************************************************************
              06324
              06325 ; DESCRIPTION
              06326 ;
              06327 ; Modifies the Display List to show and hide title, headers, and t
              06328 ; Panel Display.
              06329 ;
              06330 ; INPUT
              06331 ;
              06332 ;   A = Number of bytes to copy into the Display List
              06333 ;   X = Offset into Display List DSPLST ($0280)
              06334 ;   Y = Offset into Display List fragment table DLSTFRAG ($BA62).
              06335 ;       then no bytes are copied but the specified locations of th
              06336 ;       are overwritten with Display List instruction $0D (one row
              06337 ;       GRAPHICS7).
              06338 ;
              06339 ;   Used values are:
              06340 ;
              06341 ;    A    X    Y
              06342 ;   $08  $5F  $00 -> Show Control Panel Display (bottom text windo
              06343 ;   $08  $5F  $80 -> Hide Control Panel Display (bottom text windo
              06344 ;   $07  $0F  $23 -> Show title line
              06345 ;   $07  $0F  $80 -> Hide title line
              06346 ;   $08  $02  $1B -> Show Display List header line of Front view
              06347 ;   $08  $02  $13 -> Show Display List header line of Aft view
              06348 ;   $08  $02  $0B -> Show Display List header line of Long-Range S
              06349 ;   $08  $02  $08 -> Show Display List header line of Galactic Cha
              06350
    =006A     06351 L.NUMBYTES     = $6A                  ; Number of bytes to copy
              06352
ADF1 78       06353 MODDLST        SEI                    ; Disable IRQ
ADF2 856A     06354                STA L.NUMBYTES         ; Save number of bytes to
              06355
ADF4 AD0BD4   06356 LOOP043        LDA VCOUNT             ; Wait for ANTIC line coun
ADF7 C97C     06357                CMP #124               ; ...bottom) before changi
ADF9 90F9     06358                BCC LOOP043            ;
              06359
ADFB B962BA   06360 LOOP044        LDA DLSTFRAG,Y         ; Load byte from Display L
ADFE C8       06361                INY                    ;
ADFF 1002     06362                BPL SKIP123            ; Skip if fragment table i
AE01 A90D     06363                LDA #$0D               ; Prep Display List instru
AE03 9D8002   06364 SKIP123        STA DSPLST,X           ; Store byte in Display Li
AE06 E8       06365                INX                    ;
AE07 C66A     06366                DEC L.NUMBYTES         ;
AE09 D0F0     06367                BNE LOOP044            ; Copy next byte
              06368
```

```
AE0B 58       06369                CLI                    ; Enable IRQ
AE0C 60       06370                RTS                    ; Return
              06371
              06372 ;*****************************************************************
              06373 ;*
              06374 ;*                              CLRPLAYFIELD
              06375 ;*
              06376 ;*                           Clear PLAYFIELD memory
              06377 ;*
              06378 ;*****************************************************************
              06379
              06380 ; DESCRIPTION
              06381 ;
              06382 ; Clears PLAYFIELD memory from $1000 to $1FFF.
              06383 ;
              06384 ; This subroutine sets the start address of the memory to be clear
              06385 ; execution continues into subroutine CLRMEM ($AE0F) where the mem
              06386 ; actually cleared.
              06387
AE0D A910     06388 CLRPLAYFIELD   LDA #$10
              06389
              06390 ;*****************************************************************
              06391 ;*
              06392 ;*                                 CLRMEM
              06393 ;*
              06394 ;*                              Clear memory
              06395 ;*
              06396 ;*****************************************************************
              06397
              06398 ; DESCRIPTION
              06399 ;
              06400 ; Clears memory from a given start address to memory address $1FFF
              06401 ; subroutine is called in the following situations:
              06402 ;
              06403 ; (1)  In routine INITCOLD ($A14A) at the beginning of the game to
              06404 ;      the game's variables
              06405 ;
              06406 ; (2)  In subroutine CLRPLAYFIELD ($AE0D) to clear PLAYFIELD memor
              06407 ;
              06408 ; As a side effect this subroutine also clears the saved number of
              06409 ; and the lock-on flag.
              06410 ;
              06411 ; INPUT
              06412 ;
              06413 ;   A = Start address (high byte) of memory to be cleared. Used va
              06414 ;     $02 -> Clear memory $0200..$1FFF during game initialization
              06415 ;     $10 -> Clear PLAYFIELD memory $1000..$1FFF
              06416
AE0F 8569     06417 CLRMEM         STA MEMPTR+1           ; Store start address (hig
AE11 A900     06418                LDA #0                 ; Store start address (low
AE13 A8       06419                TAY                    ;
AE14 8568     06420                STA MEMPTR             ;
              06421
AE16 85A3     06422                STA ISINLOCKON         ; Clear lock-on flag
AE18 857A     06423                STA OLDMAXSPCOBJIND    ; Clear saved number of sp
              06424
AE1A 9168     06425 LOOP045        STA (MEMPTR),Y         ; Clear memory location
AE1C C8       06426                INY                    ;
AE1D D0FB     06427                BNE LOOP045            ;
```

```
                06428
AE1F E669       06429                INC MEMPTR+1           ; Next page (= 256-byte bl
AE21 A469       06430                LDY MEMPTR+1           ;
AE23 C020       06431                CPY #$20               ;
AE25 A8         06432                TAY                    ;
AE26 90F2       06433                BCC LOOP045            ; Loop until memory addres
AE28 60         06434                RTS                    ; Return
                06435
                06436 ;****************************************************************
                06437 ;*
                06438 ;*                                  TRIGGER
                06439 ;*
                06440 ;*                            Handle joystick trigger
                06441 ;*
                06442 ;****************************************************************
                06443
                06444 ; DESCRIPTION
                06445 ;
                06446 ; This subroutine handles the joystick trigger and launches one of
                06447 ; starship's photon torpedo. If a target is in full lock-on then a
                06448 ; torpedo is prepared for automatic launch in the next game loop i
                06449 ;
                06450 ; DETAILS
                06451 ;
                06452 ; If the trigger is pressed then reset the idle counter and, if no
                06453 ; hyperwarp, launch a photon torpedo with the following steps:
                06454 ;
                06455 ; (1)  If the trigger was pressed in this game loop iteration, a p
                06456 ;      will be launched if a previously launched photon torpedo is
                06457 ;      way for at least 255 - 232 = 23 game loop iterations. This
                06458 ;      photon torpedoes too rapidly.
                06459 ;
                06460 ; (2)  Start tracking a space object. If it is in full lock-on, se
                06461 ;      lock-on timer, activate photon torpedo tracking, and tweak
                06462 ;      trigger state such that our other photon torpedo (if availa
                06463 ;      launched automatically in the next game loop iteration.
                06464 ;
                06465 ; (3)  If the Photon Torpedoes are destroyed, do nothing.
                06466 ;
                06467 ; (4)  If the Photon Torpedoes are damaged, launch a photon torped
                06468 ;      same barrel than the previous one.
                06469 ;
                06470 ; (5)  If the Photon Torpedoes are not damaged, launch a photon to
                06471 ;      other barrel.
                06472 ;
                06473 ; (6)  Set the lifetime of our starship's photon torpedo to infini
                06474 ;      PLAYER shape to PHOTON TORPEDO.
                06475 ;
                06476 ; (7)  Initialize the position vector of our starship's photon tor
                06477 ;
                06478 ;          x-coordinate := +256 (+$0100) <KM> (Right barrel)
                06479 ;                          -256 (-$FF00) <KM> (Left barrel)
                06480 ;          y-coordinate := -256 (-$FF00) <KM>
                06481 ;          z-coordinate :=   +1 (+$0001) <KM>
                06482 ;
                06483 ; (8)  Initialize the velocity vector of our starship's photon tor
                06484 ;
                06485 ;          x-velocity   :=   +0 <KM/H>
                06486 ;          y-velocity   :=   +0 <KM/H>
```

```
                06487 ;          z-velocity   := +102 <KM/H> (All views but Aft view)
                06488 ;                          -102 <KM/H> (Aft view)
                06489 ;
                06490 ; (9)  Subtract 10 energy units for launching our starship's photo
                06491 ;
                06492 ; (10) Play the noise sound pattern PHOTON TORPEDO LAUNCHED by con
                06493 ;      execution into subroutine NOISE ($AEA8).
                06494
AE29 A584       06495 TRIGGER         LDA OLDTRIG0           ; Prep last trigger state
                06496
AE2B AC10D0     06497                 LDY TRIG0              ; Copy current trigger sta
AE2E 8484       06498                 STY OLDTRIG0           ;
AE30 D00E       06499                 BNE SKIP124            ; Return if trigger curren
                06500
AE32 8466       06501                 STY IDLECNTHI          ; Reset idle counter
                06502
AE34 A6C0       06503                 LDX WARPSTATE          ; Return if hyperwarp enga
AE36 D008       06504                 BNE SKIP124            ;
                06505
AE38 A687       06506                 LDX BARRELNR           ; Prep barrel number (0 ->
                06507
AE3A C901       06508                 CMP #1                 ; If trigger is newly pres
AE3C F003       06509                 BEQ SKIP125            ; ...and launch our starsh
AE3E B018       06510                 BCS SKIP127            ; ...else launch our stars
AE40 60         06511 SKIP124         RTS                    ; Return
                06512
                06513 ;*** Set up our starship's photon torpedo tracking ***************
AE41 B5EC       06514 SKIP125         LDA PL3LIFE,X          ; Return if torpedo's life
AE43 C9E8       06515                 CMP #232               ;
AE45 B0F9       06516                 BCS SKIP124            ;
                06517
AE47 AC5C09     06518                 LDY TRACKDIGIT         ; Store index of tracked s
AE4A 8489       06519                 STY PLTRACKED          ;
                06520
AE4C A90C       06521                 LDA #12                ; Prep lock-on lifetime :=
AE4E A4A3       06522                 LDY ISINLOCKON         ; If target is in full loc
AE50 8486       06523                 STY ISTRACKING         ; ...activate photon torpe
                06524
AE52 F002       06525                 BEQ SKIP126            ; Skip if target not in fu
AE54 A900       06526                 LDA #0                 ; Prep lock-on lifetime :=
AE56 8588       06527 SKIP126         STA LOCKONLIFE         ; Store lock-on lifetime (
                06528
                06529 ;*** Launch our starship's photon torpedo ************************
AE58 8484       06530 SKIP127         STY OLDTRIG0           ; Update last trigger stat
AE5A 2C9209     06531                 BIT GCSTATPHO          ; Return if Photon Torpedo
AE5D 70E1       06532                 BVS SKIP124            ;
                06533
AE5F 3005       06534                 BMI SKIP128            ; If Photon Torpedoes dama
AE61 8A         06535                 TXA                    ; ...else switch barrel fr
AE62 4901       06536                 EOR #$01               ;
AE64 8587       06537                 STA BARRELNR           ;
                06538
AE66 8A         06539 SKIP128         TXA                    ; SUMMARY: Our starship's
AE67 9DE109     06540                 STA PL3XPOSSIGN,X      ; x-coordinate := +256 (+$
AE6A BD73BF     06541                 LDA BARRELXTAB,X       ; x-coordinate := -256 (-$
AE6D 9D740A     06542                 STA PL3XPOSHI,X        ; y-coordinate := -256 (-$
AE70 A9FF       06543                 LDA #255               ; z-coordinate :=   +1 (+$
AE72 95EC       06544                 STA PL3LIFE,X          ; ...lifetime := 255 game
AE74 9DA50A     06545                 STA PL3YPOSHI,X        ;
```

```
AE77 A900     06546                LDA #0                  ;
AE79 9D8F0C   06547                STA PL3SHAPTYPE,X       ; PLAYER3 or PLAYER4 is PH
AE7C 9D430A   06548                STA PL3ZPOSHI,X         ;
AE7F 9D070B   06549                STA PL3XPOSLO,X         ;
AE82 9D120A   06550                STA PL3YPOSSIGN,X       ;
AE85 9D380B   06551                STA PL3YPOSLO,X         ;
AE88 A901     06552                LDA #1                  ;
AE8A 9DB009   06553                STA PL3ZPOSSIGN,X       ;
AE8D 9DD60A   06554                STA PL3ZPOSLO,X         ;
              06555
AE90 A5D0     06556                LDA SHIPVIEW            ; SUMMARY: Our starship's
AE92 4A       06557                LSR A                   ; x-velocity :=   +0 <KM/H
AE93 6A       06558                ROR A                   ; y-velocity :=   +0 <KM/H
AE94 0966     06559                ORA #102                ; z-velocity := +102 <KM/H
AE96 9D690B   06560                STA PL3ZVEL,X           ; z-velocity := -102 <KM/H
AE99 A900     06561                LDA #0                  ;
AE9B 9D9A0B   06562                STA PL3XVEL,X           ;
AE9E 9DCB0B   06563                STA PL3YVEL,X           ;
              06564
AEA1 A202     06565                LDX #2                  ; ENERGY := ENERGY - 10 fo
AEA3 206FB8   06566                JSR DECENERGY           ;
              06567
AEA6 A200     06568                LDX #$00                ; Play noise sound pattern
              06569
              06570 ;*****************************************************************
              06571 ;*
              06572 ;*                                   NOISE
              06573 ;*
              06574 ;*                          Copy noise sound pattern
              06575 ;*
              06576 ;*****************************************************************
              06577
              06578 ; DESCRIPTION
              06579 ;
              06580 ; Copies a 10-byte noise sound pattern from table NOISEPATTAB ($BF
              06581 ; 8 bytes are copied to the noise sound pattern area NOISETORPTIM
              06582 ; ($DA)..NOISELIFE ($E1), the remaining 2 bytes are copied to audi
              06583 ; AUDCTL ($D208) and AUDF3 ($D204). The noise sound pattern is aut
              06584 ; played in subroutine SOUND ($B2AB).
              06585 ;
              06586 ; NOTE: The first 8 bytes of each pattern in table NOISEPATTAB ($B
              06587 ; copied in reverse order from memory. See subroutine SOUND ($B2AB
              06588 ; on the noise sound patterns stored in NOISEPATTAB ($BF20).
              06589 ;
              06590 ; Playing a SHIELD EXPLOSION or ZYLON EXPLOSION noise sound patter
              06591 ; currently playing PHOTON TORPEDO LAUNCHED noise sound pattern.
              06592 ;
              06593 ; Playing a PHOTON TORPEDO LAUNCHED noise sound pattern overrides
              06594 ; playing PHOTON TORPEDO LAUNCHED noise sound pattern if the latte
              06595 ; TICKs to play.
              06596 ;
              06597 ; INPUT
              06598 ;
              06599 ;   X = Offset into table NOISEPATTAB ($BF20) to index noise sound
              06600 ;       Used values are:
              06601 ;     $00 -> PHOTON TORPEDO LAUNCHED
              06602 ;     $0A -> SHIELD EXPLOSION (either our starship or a starbase e
              06603 ;     $14 -> ZYLON EXPLOSION
              06604
```

```
AEA8 8A       06605 NOISE           TXA                     ; Skip if SHIELD EXPLOSION
AEA9 D006     06606                 BNE SKIP129             ;
              06607
AEAB A5E1     06608                 LDA NOISELIFE           ; Return if PHOTON TORPEDO
AEAD C918     06609                 CMP #24                 ; ...playing for yet more
AEAF B018     06610                 BCS SKIP130             ;
              06611
AEB1 A007     06612 SKIP129         LDY #7                  ; Copy noise sound pattern
AEB3 BD20BF   06613 LOOP046         LDA NOISEPATTAB,X       ;
AEB6 99DA00   06614                 STA NOISETORPTIM,Y      ;
AEB9 E8       06615                 INX                     ;
AEBA 88       06616                 DEY                     ;
AEBB 10F6     06617                 BPL LOOP046             ;
              06618
AEBD BD20BF   06619                 LDA NOISEPATTAB,X       ; Copy AUDCTL from noise s
AEC0 8D08D2   06620                 STA AUDCTL              ;
AEC3 BD21BF   06621                 LDA NOISEPATTAB+1,X     ; Copy AUDF3 from noise so
AEC6 8D04D2   06622                 STA AUDF3               ;
              06623
AEC9 60       06624 SKIP130         RTS                     ; Return
              06625
              06626 ;*****************************************************************
              06627 ;*
              06628 ;*                                   HOMINGVEL
              06629 ;*
              06630 ;*      Calculate homing velocity of our starship's photon torpedo
              06631 ;*
              06632 ;*****************************************************************
              06633
              06634 ; DESCRIPTION
              06635 ;
              06636 ; Calculates the x (or y) velocity vector component of our starshi
              06637 ; torpedo 0 or 1 when it is tracking (homing in on) a target space
              06638 ;
              06639 ; Our starship's photon torpedo's x (or y) velocity vector compone
              06640 ; the PLAYER column (or row) number difference between the target
              06641 ; starship's photon torpedo PLAYER in Player/Missile (PM) pixels.
              06642 ; difference is used as an index to pick the new x (or y) velocity
              06643 ; component of our starship's photon torpedo from table HOMVELTAB
              06644 ;
              06645 ; +---------------+--------------+
              06646 ; | Difference in | New Velocity |
              06647 ; |   PM Pixels   |  Component   |
              06648 ; +---------------+--------------+
              06649 ; |    >= +7      |   -64 <KM/H> |
              06650 ; |       +6      |   -56 <KM/H> |
              06651 ; |       +5      |   -48 <KM/H> |
              06652 ; |       +4      |   -40 <KM/H> |
              06653 ; |       +3      |   -24 <KM/H> |
              06654 ; |       +2      |   -16 <KM/H> |
              06655 ; |       +1      |    -8 <KM/H> |
              06656 ; |        0      |     0 <KM/H> |
              06657 ; |       -1      |    +8 <KM/H> |
              06658 ; |       -2      |   +16 <KM/H> |
              06659 ; |       -3      |   +24 <KM/H> |
              06660 ; |       -4      |   +40 <KM/H> |
              06661 ; |       -5      |   +48 <KM/H> |
              06662 ; |       -6      |   +56 <KM/H> |
              06663 ; |    <= -7      |   +64 <KM/H> |
```

```
                06664 ; +---------------+-------------+
                06665 ;
                06666 ; INPUT
                06667 ;
                06668 ;   A     = PLAYER column (or row) number difference between the t
                06669 ;           and our starship's photon torpedo PLAYER in Player/Mis
                06670 ;
                06671 ;   CARRY = Sign of the PLAYER column (or row) number difference.
                06672 ;           are:
                06673 ;     0 -> Negative difference (target PLAYER column (or row) numb
                06674 ;          starship's photon torpedo PLAYER column (or row) number
                06675 ;     1 -> Positive difference (target PLAYER column (or row) numb
                06676 ;          starship's photon torpedo PLAYER column (or row) number
                06677 ;
                06678 ; OUTPUT
                06679 ;
                06680 ;   A = New velocity vector component of our starship's photon tor
                06681
    =006A       06682 L.VELSIGN       = $6A                 ; Saves velocity sign
                06683
AECA A080       06684 HOMINGVEL       LDY #NEG              ; Preload negative velocit
AECC B004       06685                 BCS SKIP131           ; Skip if difference is po
                06686
AECE 49FF       06687                 EOR #$FF              ; Invert to get absolute v
AED0 A000       06688                 LDY #0                ; Preload positive velocit
                06689
AED2 846A       06690 SKIP131         STY L.VELSIGN         ; Save velocity sign
AED4 C908       06691                 CMP #8                ;
AED6 9002       06692                 BCC SKIP132           ;
AED8 A907       06693                 LDA #7                ; Limit difference to 0..7
AEDA A8         06694 SKIP132         TAY                   ;
AEDB A56A       06695                 LDA L.VELSIGN         ; Reload velocity sign
AEDD 19C9BF     06696                 ORA HOMVELTAB,Y       ; Combine with homing velo
AEE0 60         06697                 RTS                   ; Return
                06698
                06699 ;*****************************************************************
                06700 ;*
                06701 ;*                                  DAMAGE
                06702 ;*
                06703 ;*             Damage or destroy one of our starship's subsystems
                06704 ;*
                06705 ;*****************************************************************
                06706
                06707 ; DESCRIPTION
                06708 ;
                06709 ; Damages or destroys one of our starship's subsystems. There are
                06710 ;
                06711 ; (1)  Photon Torpedoes
                06712 ; (2)  Engines
                06713 ; (3)  Shields
                06714 ; (4)  Attack Computer
                06715 ; (5)  Long-Range Scan
                06716 ; (6)  Subspace Radio
                06717 ;
                06718 ; Their status is stored and displayed in the Galactic Chart Panel
                06719 ; the colored letters PESCLR. The color of each letter represents
                06720 ; subsystem's status:
                06721 ;
                06722 ; +---------------+-----------------+
```

```
06723 ; | Letter Color  | Subsystem Status |
06724 ; +---------------+------------------+
06725 ; | {LIGHT GREEN} | OK               |
06726 ; | {CORN YELLOW} | Damaged          |
06727 ; | {PINK}        | Destroyed        |
06728 ; +---------------+------------------+
06729 ;
06730 ; This subroutine first makes sure that we are not in demo mode. T
06731 ; random value in 0..255 and the damage probability value. The lat
06732 ; depends on the mission level and is picked from table DAMAGEPROB
06733 ;
06734 ; +-----------+-------------------+---------------+
06735 ; |  Mission  |       Damage      |    Damage     |
06736 ; |   Level   | Probability Value |  Probability  |
06737 ; +-----------+-------------------+---------------+
06738 ; | NOVICE    |          0        |  0% (  0:256) |
06739 ; | PILOT     |         80        | 31% ( 80:256) |
06740 ; | WARRIOR   |        180        | 70% (180:256) |
06741 ; | COMMANDER |        254        | 99% (254:256) |
06742 ; +-----------+-------------------+---------------+
06743 ;
06744 ; If the random number is lower than the damage probability value,
06745 ; picked subsystem is about to get damaged (or destroyed). There i
06746 ; upfront probability of 25% (2:8) that no subsystem gets harmed.
06747 ;
06748 ; If the picked subsystem is already destroyed then another subsys
06749 ;
06750 ; Then the title phrase offset is picked from table DAMAGEPHRTAB (
06751 ; display the damaged subsystem in the title line. Next, color bit
06752 ; that indicate a damaged system.
06753 ;
06754 ; If the Zylon photon torpedo's lifetime >= 30 game loop iteration
06755 ; subsystem will not only be damaged but destroyed.
06756 ;
06757 ; NOTE: The Zylon photon torpedo lifetime decreases from 62 to 0 g
06758 ; iterations. With a remaining lifetime >= 30 game loop iterations
06759 ; considered strong enough to destroy one of our starship's subsys
06760 ; are two exceptions to this rule: If the Attack Computer was pick
06761 ; destroyed it will be damaged only - not destroyed - if the Long-
06762 ; been already destroyed, and vice versa.
06763 ;
06764 ; Then the title phrase offset from table DESTROYPHRTAB ($BF1A) is
06765 ; display the destroyed subsystem in the title line. Next, color b
06766 ; that indicate a destroyed system.
06767 ;
06768 ; The color of the subsystem's status letter is adjusted in the Ga
06769 ; Panel Display. Next, the title phrase describing the subsystem's
06770 ; enqueued for display in the title line. If the Attack Computer h
06771 ; destroyed it is switched off and the PLAYFIELD is cleared. The t
06772 ; updated with the "DAMAGE CONTROL" message. Finally, the beeper s
06773 ; DAMAGE REPORT is played in subroutine BEEP ($B3A6).
06774
AEE1 2464   06775 DAMAGE          BIT ISDEMOMODE          ; Return if in demo mode
AEE3 3057   06776                 BMI SKIP137             ;
06777
06778 ;*** Damage some subsystem ************************************
AEE5 A662   06779                 LDX MISSIONLEVEL        ; Prep mission level
AEE7 AD0AD2 06780 LOOP047         LDA RANDOM              ; Return if random number
AEEA DD10BF 06781                 CMP DAMAGEPROBTAB,X     ; ...(the latter depends o
```

```
AEED B04D     06782                BCS SKIP137            ;
              06783
AEEF 2907     06784                AND #$07               ; Randomly pick 1 of 6 sub
AEF1 C906     06785                CMP #6                 ; Return if no subsystem p
AEF3 B047     06786                BCS SKIP137            ;
              06787
AEF5 AA       06788                TAX                    ;
AEF6 BD9209   06789                LDA GCSTATPHO,X        ; Get picked subsystem sta
AEF9 0A       06790                ASL A                  ; Check bit B6 (= destroye
AEFA 30EB     06791                BMI LOOP047            ; Try again if subsystem a
              06792
AEFC A5EB     06793                LDA PL2LIFE            ; Load Zylon photon torped
AEFE C91E     06794                CMP #30                ; ...and compare it to 30
              06795
AF00 A980     06796                LDA #CCS.COL2          ; Preload COLOR2 text colo
AF02 BC14BF   06797                LDY DAMAGEPHRTAB,X     ; Preload title phrase off
              06798
AF05 9017     06799                BCC SKIP135            ; Skip if Zylon torpedo li
              06800
AF07 E003     06801                CPX #3                 ; Skip if selected subsyst
AF09 D005     06802                BNE SKIP133            ;
AF0B 2C9609   06803                BIT GCSTATLRS          ; Skip if Long-Range Scan
AF0E 700E     06804                BVS SKIP135            ;
AF10 E004     06805 SKIP133        CPX #4                 ; Skip if selected subsyst
AF12 D005     06806                BNE SKIP134            ;
AF14 2C9509   06807                BIT GCSTATCOM          ; Skip if Attack Computer
AF17 7005     06808                BVS SKIP135            ;
              06809
AF19 A9C0     06810 SKIP134        LDA #CCS.COL3          ; Preload COLOR3 text colo
AF1B BC1ABF   06811                LDY DESTROYPHRTAB,X    ; Preload title phrase off
              06812
AF1E 1D9209   06813 SKIP135        ORA GCSTATPHO,X        ; Combine status letter wi
AF21 9D9209   06814                STA GCSTATPHO,X        ;
AF24 8465     06815                STY NEWTITLEPHR        ; Enqueue damage status ti
AF26 2C9509   06816                BIT GCSTATCOM          ; Skip if Attack Computer
AF29 5007     06817                BVC SKIP136            ;
              06818
AF2B A900     06819                LDA #0                 ; Switch Attack Computer o
AF2D 857E     06820                STA DRAINATTCOMP       ;
AF2F 200DAE   06821                JSR CLRPLAYFIELD       ; Clear PLAYFIELD
              06822
AF32 A052     06823 SKIP136        LDY #$52               ; Set title phrase "DAMAGE
AF34 2023B2   06824                JSR SETTITLE           ;
              06825
AF37 A212     06826                LDX #$12               ; Play beeper sound patter
AF39 20A6B3   06827                JSR BEEP               ;
              06828
AF3C 60       06829 SKIP137        RTS                    ; Return
              06830
              06831 ;*****************************************************************
              06832 ;*
              06833 ;*                                  COLLISION
              06834 ;*
              06835 ;*            Detect a collision of our starship's photon torpedoe
              06836 ;*
              06837 ;*****************************************************************
              06838
              06839 ; DESCRIPTION
              06840 ;
```

```
; Both of our starship's photon torpedoes are checked if they have
; a space object represented by PLAYER0..2, such as a Zylon ship,
; torpedo, a starbase, or a meteor.
;
; For quick lookup, the following table lists the PLAYERs and what
; they represent:
;
; +--------+-----------------------------------------------------+
; | PLAYER |                     Represents                      |
; +--------+-----------------------------------------------------+
; |   0    | Zylon ship 0, Starbase Left                         |
; |   1    | Zylon ship 1, Starbase Right                        |
; |   2    | Zylon photon torpedo, Starbase Center, Meteor       |
; |   3    | Our starship's photon torpedo 0                     |
; |   4    | Our starship's photon torpedo 1, Transfer Vessel    |
; +--------+-----------------------------------------------------+
;
; NOTE: Only space objects represented by PLAYER0..2 are checked f
; The transfer vessel of the starbase, represented by PLAYER4, is
; and therefore cannot be destroyed by one of our starship's photo
;
; This subroutine first checks if our starship's photon torpedoes
; represented by alive PLAYERs with PHOTON TORPEDO shape.
;
; In order to detect a collision with a space object, our starship
; torpedo must compare its x, y, and z coordinates with the ones o
; object.
;
; Instead of comparing the x and y coordinates, however, this subr
; much more efficient method by inspecting the Player/Missile coll
; registers, as the x and y axis of the 3D coordinate system estab
; in which the TV screen lies. Each of our starship's photon torpe
; own Player/Missile collision register: PL3HIT ($82) for our star
; torpedo 0 and PL4HIT ($83) for our starship's photon torpedo 1.
; these registers the hit space object is determined:
;
; +-------------------------------------------------+-----------
; |          Bits B2..0 of Collision Register       |        Hit
; |              (0 -> Not Hit, 1 -> Hit)           |
; +-----------------+----------------+----------------+
; |     PLAYER2     |     PLAYER1    |    PLAYER0     |
; | (Zylon torpedo) | (Zylon ship 1) | (Zylon ship 0) |
; +-----------------+----------------+----------------+-----------
; |        0        |        0       |        0       | None
; |        0        |        0       |        1       | PLAYER0 (Z
; |        0        |        1       |        0       | PLAYER1 (Z
; |        0        |        1       |        1       | PLAYER1 (Z
; |        1        |        0       |        0       | PLAYER2 (Z
; |        1        |        0       |        1       | PLAYER2 (Z
; |        1        |        1       |        0       | PLAYER1 (Z
; |        1        |        1       |        1       | PLAYER1 (Z
; +-----------------+----------------+----------------+-----------
;
; If the lifetime of the hit space object has already expired, the
; ignored.
;
; A collision along the z-axis happens if the z-coordinate of our
; photon torpedo is close enough to the z-coordinate of the space
; is determined as follows:
```

```
                06900 ;
                06901 ; The absolute value of the z-coordinate of the space object is co
                06902 ; range index in 0..7. This index picks a minimum and a maximum z-
                06903 ; from tables HITMINZTAB ($BF7D) and HITMAXZTAB ($BF75). If the ab
                06904 ; of the z-coordinate of our starship's photon torpedo is inside t
                06905 ; then our starship's photon torpedo has hit the space object. The
                06906 ; table lists the relevant values:
                06907 ;
                06908 ; +----------------------+-------+--------------------------+----
                06909 ; | ABS(z-Coordinate)     | Range | Min ABS(z-Coordinate)    | Max
                06910 ; | of Space Object       | Index | of Photon Torpedo to Hit | of
                06911 ; +----------------------+-------+--------------------------+----
                06912 ; | <=   511 ($01**) <KM> |   0   |           0 ($00**) <KM> |
                06913 ; | <=  1023 ($03**) <KM> |   1   |           0 ($00**) <KM> |
                06914 ; | <=  1535 ($05**) <KM> |   2   |           0 ($00**) <KM> |
                06915 ; | <=  2047 ($07**) <KM> |   3   |         512 ($02**) <KM> |
                06916 ; | <=  2559 ($09**) <KM> |   4   |        1024 ($04**) <KM> |
                06917 ; | <=  3071 ($0B**) <KM> |   5   |        1536 ($06**) <KM> |
                06918 ; | <=  3583 ($0D**) <KM> |   6   |        2048 ($08**) <KM> |
                06919 ; | <= 65535 ($FF**) <KM> |   7   |        3072 ($0C**) <KM> |
                06920 ; +----------------------+-------+--------------------------+----
                06921 ;
                06922 ; If a collision has been detected, the "age" (= initial lifetime
                06923 ; lifetime) of our starship's photon torpedo is calculated. This a
                06924 ; delay playing the ZYLON EXPLOSION noise sound pattern. It is als
                06925 ; determine the strength of our starship's photon torpedo. Only ph
                06926 ; of an age < 15 game loop iterations can destroy a Zylon basestar
                06927 ;
                06928 ; Some clean-up work is done before the actual explosion: The lock
                06929 ; starship's photon torpedo lifetime, and the hit space object's P
                06930 ; is set to 0.
                06931 ;
                06932 ; If a meteor or a Zylon photon torpedo have been hit, then the sc
                06933 ; changed, skipping right to the explosion part. Otherwise, our st
                06934 ; photon torpedo tracking flag is cleared and the Galactic Chart M
                06935 ; If a starbase was destroyed, then 3 points are subtracted from t
                06936 ; Zylon ship was destroyed, then 6 points are added to the score a
                06937 ; KILL COUNTER readout of the Control Panel Display is incremented
                06938 ; explosion is initialized in subroutine INITEXPL ($AC6B).
                06939 ;
                06940 ; NOTE: This subroutine lacks proper explosion initialization if t
                06941 ; was hit. The actual explosion initialization is done in subrouti
                06942 ; ($ACE6) when the code finds out that the starbase sector is no m
                06943 ; such in the Galactic Chart.
                06944 ;
                06945 ; Finally, the Galactic Chart Map is searched for a remaining Zylo
                06946 ; none is found then the mission is complete and code execution co
                06947 ; subroutine GAMEOVER2 ($B121), ending the game.
                06948
    =006B       06949 L.PLHIT         = $6B                  ; Saves PLAYER (and space
    =006C       06950 L.VIEWDIR       = $6C                  ; Saves view direction. Us
                06951                                        ;   $00 -> Front view
                06952                                        ;   $FF -> Aft view
                06953
AF3D A202       06954 COLLISION       LDX #2                 ; Loop over our starship's
AF3F CA         06955 LOOP048         DEX                    ;
AF40 1001       06956                 BPL SKIP138            ; Branch into loop body be
AF42 60         06957                 RTS                    ; Return
                06958
```

```
                06959 ;*** Photon torpedo sanity checks ******************************
AF43 BD8F0C     06960 SKIP138         LDA PL3SHAPTYPE,X       ; Next photon torpedo if P
AF46 D0F7       06961                 BNE LOOP048             ;
                06962
AF48 B5EC       06963                 LDA PL3LIFE,X           ; Next photon torpedo if P
AF4A F0F3       06964                 BEQ LOOP048             ;
                06965
                06966 ;*** Check if our starship's photon torpedo has hit in x-y plane *
AF4C B582       06967                 LDA PL3HIT,X            ; Check Player/Missile col
AF4E 2907       06968                 AND #$07                ; Next torpedo if no torpe
AF50 F0ED       06969                 BEQ LOOP048             ;
                06970
AF52 4A         06971                 LSR A                   ; Find out which of PLAYER
AF53 C903       06972                 CMP #3                  ;
AF55 D001       06973                 BNE SKIP139             ;
AF57 4A         06974                 LSR A                   ;
AF58 A8         06975 SKIP139         TAY                     ; Save resulting index of
                06976
AF59 B9E900     06977                 LDA PL0LIFE,Y           ; Next torpedo if PLAYER0.
AF5C F0E1       06978                 BEQ LOOP048             ;
                06979
                06980 ;*** Has our starship's photon torpedo hit within valid z-coordina
AF5E A5D0       06981                 LDA SHIPVIEW            ; Skip if in Front view
AF60 F002       06982                 BEQ SKIP140             ;
AF62 A9FF       06983                 LDA #$FF                ; Calculate range index...
AF64 856C       06984 SKIP140         STA L.VIEWDIR           ; Saves view direction
AF66 59400A     06985                 EOR ZPOSHI,Y            ; Calc ABS(z-coordinate (h
AF69 C910       06986                 CMP #16                 ; Limit range index to 0..
AF6B 9002       06987                 BCC SKIP141             ;
AF6D A90F       06988                 LDA #15                 ;
AF6F 4A         06989 SKIP141         LSR A                   ;
AF70 846B       06990                 STY L.PLHIT             ; Save index of hit PLAYER
                06991
AF72 A8         06992                 TAY                     ;
AF73 A56C       06993                 LDA L.VIEWDIR           ; Reload view direction
AF75 5D430A     06994                 EOR PL3ZPOSHI,X         ; Calc ABS(z-coordinate (h
                06995
AF78 D975BF     06996                 CMP HITMAXZTAB,Y        ; Next torpedo if torpedo
AF7B B0C2       06997                 BCS LOOP048             ;
                06998
AF7D D97DBF     06999                 CMP HITMINZTAB,Y        ; Next torpedo if torpedo
AF80 90BD       07000                 BCC LOOP048             ;
                07001
                07002 ;*** Our starship's photon torpedo has hit within valid z-coordina
AF82 A46B       07003                 LDY L.PLHIT             ; Reload index of hit PLAY
AF84 38         07004                 SEC                     ; Calc "age" of photon tor
AF85 A9FF       07005                 LDA #255                ; delay playing ZYLON EXPL
AF87 F5EC       07006                 SBC PL3LIFE,X           ;
AF89 85E2       07007                 STA NOISEZYLONTIM       ;
                07008
AF8B C90F       07009                 CMP #15                 ; Skip if photon torpedo "
AF8D 9005       07010                 BCC SKIP142             ;
AF8F B98C0C     07011                 LDA PL0SHAPTYPE,Y       ; CARRY := PLAYER is ZYLON
AF92 C980       07012                 CMP #SHAP.ZBASESTAR     ; (and torpedo "age" good
                07013
                07014 ;*** Clean up our starship's photon torpedo and hit PLAYER *******
AF94 A900       07015 SKIP142         LDA #0                  ; Lock-on lifetime := 0 ga
AF96 8588       07016                 STA LOCKONLIFE          ;
AF98 95EC       07017                 STA PL3LIFE,X           ; Photon torpedo's lifetim
```

```
AF9A B04B     07018                 BCS SKIP144             ; If CARRY set do not scor
              07019
AF9C 99E900   07020                 STA PL0LIFE,Y           ; Hit PLAYER lifetime := 0
              07021
AF9F B98C0C   07022                 LDA PL0SHAPTYPE,Y       ; If hit PLAYER is...
AFA2 F043     07023                 BEQ SKIP144             ; ...a PHOTON TORPEDO (sha
AFA4 C960     07024                 CMP #SHAP.METEOR        ; ...or a METEOR (shape ty
AFA6 F03F     07025                 BEQ SKIP144             ; ...do not score, just do
              07026
AFA8 A900     07027                 LDA #0                  ; Clear photon torpedo tra
AFAA 8586     07028                 STA ISTRACKING          ;
              07029
              07030 ;*** Zylon ship (or starbase) destroyed! ************************
AFAC A690     07031                 LDX CURRSECTOR          ; Decrement Zylon count on
AFAE DEC908   07032                 DEC GCMEMMAP,X          ;
AFB1 1013     07033                 BPL SKIP143             ; Skip if destroyed space
              07034
              07035 ;*** Starbase destroyed! ***************************************
AFB3 A900     07036                 LDA #0                  ; Remove destroyed starbas
AFB5 9DC908   07037                 STA GCMEMMAP,X          ;
AFB8 38       07038                 SEC                     ; SCORE := SCORE - 3 for d
AFB9 A5CB     07039                 LDA SCORE               ;
AFBB E903     07040                 SBC #3                  ;
AFBD 85CB     07041                 STA SCORE               ;
AFBF A5CC     07042                 LDA SCORE+1             ;
AFC1 E900     07043                 SBC #0                  ;
AFC3 85CC     07044                 STA SCORE+1             ;
AFC5 60       07045                 RTS                     ; Return
              07046
              07047 ;*** Zylon ship destroyed! *************************************
AFC6 18       07048 SKIP143         CLC                     ; SCORE := SCORE + 6 for d
AFC7 A5CB     07049                 LDA SCORE               ;
AFC9 6906     07050                 ADC #6                  ;
AFCB 85CB     07051                 STA SCORE               ;
AFCD A5CC     07052                 LDA SCORE+1             ;
AFCF 6900     07053                 ADC #0                  ;
AFD1 85CC     07054                 STA SCORE+1             ;
              07055
AFD3 A201     07056                 LDX #1                  ; Increment Zylon KILL COU
AFD5 FE5009   07057 LOOP049         INC KILLCNTD1,X         ; ...of Control Panel Disp
AFD8 BD5009   07058                 LDA KILLCNTD1,X         ;
AFDB C94A     07059                 CMP #[CCS.COL1!CCS.9]+1 ;
AFDD 9008     07060                 BCC SKIP144             ;
AFDF A940     07061                 LDA #[CCS.COL1!CCS.0]   ;
AFE1 9D5009   07062                 STA KILLCNTD1,X         ;
AFE4 CA       07063                 DEX                     ;
AFE5 10EE     07064                 BPL LOOP049             ;
              07065
AFE7 206BAC   07066 SKIP144         JSR INITEXPL            ; Init explosion at hit PL
              07067
              07068 ;*** Any Zylon ships left? *************************************
AFEA A27F     07069                 LDX #127                ; Scan all sectors of Gala
AFEC BDC908   07070 LOOP050         LDA GCMEMMAP,X          ;
AFEF 3002     07071                 BMI SKIP145             ;
AFF1 D00A     07072                 BNE SKIP146             ; Return if Zylon sector f
AFF3 CA       07073 SKIP145         DEX                     ;
AFF4 10F6     07074                 BPL LOOP050             ;
              07075
              07076 ;*** Game over (Mission Complete) *******************************
```

```
AFF6 A03F     07077                LDY #$3F               ; Set title phrase "MISSIO
AFF8 A200     07078                LDX #0                 ; Set mission bonus offset
AFFA 2021B1   07079                JSR GAMEOVER2          ; Game over
AFFD 60       07080 SKIP146        RTS                    ; Return
              07081
              07082 ;****************************************************************
              07083 ;*
              07084 ;*                                KEYBOARD
              07085 ;*
              07086 ;*                           Handle Keyboard Input
              07087 ;*
              07088 ;****************************************************************
              07089
              07090 ; DESCRIPTION
              07091 ;
              07092 ; If a keyboard code has been collected during a keyboard IRQ in t
              07093 ; Interrupt Request handler IRQHNDLR ($A751), the idle counter is
              07094 ; PLAYER-PLAYFIELD priority arranges the PLAYERs in front of the P
              07095 ;
              07096 ; Then, the keyboard code is compared with keyboard codes of table
              07097 ; ($BABE). If no match is found the "WHAT'S WRONG" message is disp
              07098 ; title line and code execution returns.
              07099 ;
              07100 ; If one of the speed keys '0'..'9' has been pressed, a pending hy
              07101 ; aborted in subroutine ABORTWARP ($A980) and code execution retur
              07102 ; the Engines drain rate is adjusted as well as the new velocity o
              07103 ; starship. If the Engines are damaged, a maximum speed is possibl
              07104 ; to speed key '5'.
              07105 ;
              07106 ; If one of our starship's view keys 'F' (Front), 'A' (Aft), 'G' (
              07107 ; Chart), or 'L' (Long-Range Scan) have been pressed, the Display
              07108 ; modified accordingly in subroutine MODDLST ($ADF1) and a new sta
              07109 ; stars is created with the help of subroutine INITPOSVEC ($B764).
              07110 ; execution returns via subroutine UPDSCREEN ($B07B).
              07111 ;
              07112 ; If one of the 'T' (Tracking Computer), 'S' (Shields) or 'C' (Att
              07113 ; keys have been pressed, the corresponding status bits are toggle
              07114 ; title line is updated with the corresponding title phrase. The b
              07115 ; pattern ACKNOWLEDGE is played in subroutine BEEP ($B3A6). The tr
              07116 ; of the Control Panel Display is updated and the PLAYFIELD is cle
              07117 ; subroutine CLRPLAYFIELD ($AE0D). If the Attack Computer is on, t
              07118 ; Aft view cross hairs are drawn, depending on the current view of
              07119 ; via subroutine DRAWLINES ($A76F).
              07120 ;
              07121 ; If the 'H' (Hyperwarp) key has been pressed then the hyperwarp i
              07122 ; starship's velocity is set to the maximum value, the Engines dra
              07123 ; increased to the equivalent of speed key '7'. Star trails are pr
              07124 ; position vector of the Hyperwarp Target Marker (PLAYER3) is set
              07125 ; following values:
              07126 ;
              07127 ;     x-coordinate :=   +0 (+$0000) <KM>
              07128 ;     y-coordinate := +256 (+$0100) <KM>
              07129 ;     z-coordinate :=    + (+$****) <KM> (sign only)
              07130 ;
              07131 ; The velocity vector is set to the following values:
              07132 ;
              07133 ;     x-velocity   :=  (not initialized)
              07134 ;     y-velocity   :=  (not initialized)
              07135 ;     z-velocity   :=          +0 <KM/H>
```

```
                07136 ;
                07137 ; The temporary arrival hyperwarp marker column and row numbers ar
                07138 ; are not in a NOVICE mission, the maximum veer-off velocity of th
                07139 ; Target Marker during hyperwarp is picked from table VEERMASKTAB
                07140 ; value depends on the selected hyperwarp energy (and thus on the
                07141 ; hyperwarp). Finally, the title line displays the "HYPERWARP ENGA
                07142 ;
                07143 ; If the 'M' (Manual target selector) key has been pressed, the tr
                07144 ; space object is swapped and the corresponding digit of the Contr
                07145 ; Display is toggled between 0 and 1.
                07146 ;
                07147 ; If the 'P' (Pause) key has been pressed, an endless loop waits u
                07148 ; joystick is pushed.
                07149 ;
                07150 ; BUG (at $B103): The endless loop branches back one instruction t
                07151 ; Suggested fix: Branch to instruction LDA PORTA at $B0FE.
                07152 ;
                07153 ; If the 'INV' (Abort mission) key has been pressed, the mission i
                07154 ; setting the mission bonus offset, then displaying the "MISSION A
                07155 ; message in the title line. Code execution continues into subrout
                07156 ; ($B10A).
                07157 ;
                07158 ; NOTE: This subroutine has two additional entry points:
                07159 ;
                07160 ; (1)  SETVIEW ($B045), which is used to enforce the Front view. I
                07161 ;      from the game loop GAMELOOP ($A1F3) and subroutines INITSTA
                07162 ;      DECENERGY ($B86F).
                07163 ;
                07164 ; (2)  UPDSCREEN ($B07B), which draws the cross hairs and the Atta
                07165 ;      Display, and then sets the tracking letter of the Control P
                07166 ;      It is entered from subroutine DOCKING ($ACE6).
                07167
    =006A       07168 L.KEYCODE       = $6A                   ; Saves pressed keyboard c
                07169
AFFE A5CA       07170 KEYBOARD        LDA KEYCODE             ; Return if no keyboard co
B000 F03E       07171                 BEQ SKIP150             ;
                07172
B002 A214       07173                 LDX #20                 ; Prep keyboard code table
B004 856A       07174                 STA L.KEYCODE           ; Save keyboard code
                07175
B006 A900       07176                 LDA #0                  ; Reset idle counter
B008 8566       07177                 STA IDLECNTHI           ;
B00A 85CA       07178                 STA KEYCODE             ; Clear keyboard code
                07179
B00C A911       07180                 LDA #$11                ; GTIA: Enable PLAYER4, pr
B00E 8D1BD0     07181                 STA PRIOR               ; (PLAYERs in front of sta
                07182
                07183 ;*** Search keyboard code in lookup table **********************
                07184
B011 BDBEBA     07185 LOOP051         LDA KEYTAB,X            ; Loop over all valid keyb
B014 C56A       07186                 CMP L.KEYCODE           ;
B016 F008       07187                 BEQ SKIP147             ; Branch if matching entry
B018 CA         07188                 DEX                     ;
B019 10F6       07189                 BPL LOOP051             ; Next keyboard code
                07190
B01B A010       07191                 LDY #$10                ; No match found...
B01D 4C23B2     07192                 JMP SETTITLE            ; ...set title phrase "WHA
                07193
                07194 ;*** Handle '0'..'9' keyboard keys (speed) **********************
```

```
B020 E00A     07195 SKIP147         CPX #10                 ; Skip section if keyboard
B022 B01D     07196                 BCS SKIP151             ;
              07197
B024 A5C0     07198                 LDA WARPSTATE           ; Skip if hyperwarp diseng
B026 F003     07199                 BEQ SKIP148             ;
B028 4C80A9   07200                 JMP ABORTWARP           ; ...else abort hyperwarp
              07201
B02B 2C9309   07202 SKIP148         BIT GCSTATENG           ; Skip if Engines are OK o
B02E 5006     07203                 BVC SKIP149             ;
B030 E006     07204                 CPX #6                  ; Allow max velocity equiv
B032 9002     07205                 BCC SKIP149             ;
B034 A205     07206                 LDX #5                  ;
              07207
B036 BDD3BA   07208 SKIP149         LDA DRAINRATETAB,X      ; Set Engines energy drain
B039 8580     07209                 STA DRAINENGINES        ;
B03B BDB4BA   07210                 LDA VELOCITYTAB,X       ; Set new velocity
B03E 8571     07211                 STA NEWVELOCITY         ;
B040 60       07212 SKIP150         RTS                     ; Return
              07213
              07214 ;*** Handle 'F', 'A', 'L', 'G' keyboard keys (our starship's views
B041 E00E     07215 SKIP151         CPX #14                 ; Skip section if keyboard
B043 B01B     07216                 BCS SKIP152             ;
              07217
              07218 ;*** Entry to force Front view after game init and failed missions
B045 BD18BE   07219 SETVIEW         LDA VIEWMODETAB-10,X    ; Store our starship's vie
B048 85D0     07220                 STA SHIPVIEW            ;
              07221
B04A BC82BA   07222                 LDY DLSTFRAGOFFTAB-10,X ; Get DL fragment offset (
B04D A202     07223                 LDX #$02                ; Switch to corresponding
B04F A908     07224                 LDA #$08                ;
B051 20F1AD   07225                 JSR MODDLST             ;
              07226
B054 A210     07227                 LDX #NUMSPCOBJ.NORM-1   ; Create new star field of
B056 2064B7   07228 LOOP052         JSR INITPOSVEC          ;
B059 CA       07229                 DEX                     ;
B05A E005     07230                 CPX #NUMSPCOBJ.PL       ;
B05C B0F8     07231                 BCS LOOP052             ;
              07232
B05E 901B     07233                 BCC UPDSCREEN           ; Return via updating scre
              07234
              07235 ;*** Handle 'T', 'S', 'C' keyboard keys (Tracking, Shields, Attack
B060 E011     07236 SKIP152         CPX #17                 ; Skip section if keyboard
B062 B035     07237                 BCS SKIP156             ;
              07238
B064 BC18BE   07239                 LDY MSGOFFTAB-14,X      ; Prep title phrase offset
B067 B56E     07240                 LDA ISTRACKCOMPON-14,X  ; Toggle status bits (also
B069 5D1BBE   07241                 EOR MSGBITTAB-14,X      ;
B06C 956E     07242                 STA ISTRACKCOMPON-14,X  ;
B06E F003     07243                 BEQ SKIP153             ;
B070 BC1EBE   07244                 LDY MSGONTAB-14,X       ; Prep title phrase offset
B073 2023B2   07245 SKIP153         JSR SETTITLE            ; Set title phrase to "...
              07246
B076 A20C     07247                 LDX #$0C                ; Play beeper sound patter
B078 20A6B3   07248                 JSR BEEP                ;
              07249
              07250 ;*** Update PLAYFIELD (Cross hairs, Attack Computer, set tracking
B07B A216     07251 UPDSCREEN       LDX #CCS.T              ; Get custom char 'T' (ent
B07D A47C     07252                 LDY ISTRACKCOMPON       ;
B07F F001     07253                 BEQ SKIP154             ; Skip if Tracking Compute
```

```
             07254
B081 E8      07255                INX                    ; Get custom char 'C'
             07256
B082 8E5A09  07257 SKIP154        STX TRACKC1            ; Store tracking character
B085 200DAE  07258                JSR CLRPLAYFIELD       ; Clear PLAYFIELD
B088 A57E    07259                LDA DRAINATTCOMP       ; Return if Attack Compute
B08A F0B4    07260                BEQ SKIP150            ;
             07261
B08C A6D0    07262                LDX SHIPVIEW           ; If Aft view   -> Draw Af
B08E F006    07263                BEQ SKIP155            ; If Front view -> Draw Fr
B090 E001    07264                CPX #$01               ;                  ...Atta
B092 D0AC    07265                BNE SKIP150            ;
B094 A22A    07266                LDX #$2A               ;
B096 4C6FA7  07267 SKIP155        JMP DRAWLINES          ;
             07268
             07269 ;*** Handle 'H' keyboard key (Hyperwarp) ************************
B099 E011    07270 SKIP156        CPX #17                ; Skip if keyboard code do
B09B D050    07271                BNE SKIP158            ;
             07272
             07273 ;*** Engage Hyperwarp ***************************************
B09D A5C0    07274                LDA WARPSTATE          ; Return if hyperwarp enga
B09F D05A    07275                BNE SKIP159            ;
             07276
B0A1 A97F    07277                LDA #$7F               ; Engage hyperwarp
B0A3 85C0    07278                STA WARPSTATE          ;
B0A5 A9FF    07279                LDA #255               ; Set new velocity
B0A7 8571    07280                STA NEWVELOCITY        ;
B0A9 A91E    07281                LDA #30                ; Set Engines energy drain
B0AB 8580    07282                STA DRAINENGINES       ;
             07283
B0AD A930    07284                LDA #NUMSPCOBJ.ALL-1   ; Set space obj index of f
B0AF 85C3    07285                STA TRAILIND           ;
B0B1 A900    07286                LDA #0                 ; Clear star trail delay
B0B3 85C2    07287                STA TRAILDELAY         ;
             07288
B0B5 8D740A  07289                STA PL3XPOSHI          ; Init position vector and
B0B8 8D070B  07290                STA PL3XPOSLO          ; ... Hyperwarp Target Mar
B0BB 8D380B  07291                STA PL3YPOSLO          ; x-coordinate :=   +0 (+$
B0BE 8D690B  07292                STA PL3ZVEL            ; y-coordinate := +256 (+$
B0C1 A901    07293                LDA #1                 ; z-coordinate :=    + (+$
B0C3 8DB009  07294                STA PL3ZPOSSIGN        ; z-velocity := +0 <KM/H>
B0C6 8DE109  07295                STA PL3XPOSSIGN        ;
B0C9 8D120A  07296                STA PL3YPOSSIGN        ;
B0CC 8DA50A  07297                STA PL3YPOSHI          ;
             07298
B0CF A58F    07299                LDA WARPARRVCOLUMN     ; Store temp arrival hyper
B0D1 85C4    07300                STA WARPTEMPCOLUMN     ;
B0D3 A58E    07301                LDA WARPARRVROW        ; Store temp arrival hyper
B0D5 85C5    07302                STA WARPTEMPROW        ;
             07303
B0D7 A562    07304                LDA MISSIONLEVEL       ; Skip if NOVICE mission
B0D9 F00B    07305                BEQ SKIP157            ;
             07306
B0DB A591    07307                LDA WARPENERGY         ; Bits B0..1 of hyperwarp
B0DD 2A      07308                ROL A                  ; ...containing the maximu
B0DE 2A      07309                ROL A                  ; ...Hyperwarp Target Mark
B0DF 2A      07310                ROL A                  ; ...hyperwarp
B0E0 2903    07311                AND #$03               ;
B0E2 A8      07312                TAY                    ;
```

```
B0E3 B9D7BE   07313                LDA VEERMASKTAB,Y      ;
              07314
B0E6 85C6     07315 SKIP157        STA VEERMASK           ; Store veer-off velocity
              07316
B0E8 A011     07317                LDY #$11               ; Set title phrase "HYPERW
B0EA 4C23B2   07318                JMP SETTITLE           ;
              07319
              07320 ;*** Handle 'M' keyboard key (Manual Target Selector) key ********
B0ED E013     07321 SKIP158        CPX #19                ; Skip if keyboard code do
B0EF B00B     07322                BCS SKIP160            ;
              07323
B0F1 AD5C09   07324                LDA TRACKDIGIT         ; Toggle digit of tracked
B0F4 4901     07325                EOR #$01               ; ... Control Panel Displa
B0F6 2901     07326                AND #$01               ;
B0F8 8D5C09   07327                STA TRACKDIGIT         ;
B0FB 60       07328 SKIP159        RTS                    ; Return
              07329
              07330 ;*** Handle 'P' keyboard key (Pause) ****************************
B0FC D008     07331 SKIP160        BNE SKIP161            ; Skip if keyboard code do
              07332
B0FE AD00D3   07333                LDA PORTA              ; Push joystick to resume
B101 C9FF     07334                CMP #$FF               ;
B103 F0F7     07335                BEQ SKIP160            ; (!)
B105 60       07336                RTS                    ; Return
              07337
              07338 ;*** Handle 'INV' keyboard key (Abort Mission) ******************
B106 A076     07339 SKIP161        LDY #$76               ; Preload title phrase "MI
B108 A204     07340                LDX #$04               ; Set mission bonus offset
              07341
              07342 ;*********************************************************************
              07343 ;*
              07344 ;*                                GAMEOVER
              07345 ;*
              07346 ;*                             Handle game over
              07347 ;*
              07348 ;*********************************************************************
              07349
              07350 ; DESCRIPTION
              07351 ;
              07352 ; Handles game over, including calculating the scored rank and cla
              07353 ;
              07354 ; This subroutine has two entry points:
              07355 ;
              07356 ; (1)  GAMEOVER ($B10A) is entered at the end of a failed mission
              07357 ;      aborted, zero energy, or starship destroyed by Zylon fire),
              07358 ;      shutting down our starship. Code execution continues into G
              07359 ;      ($B121) below.
              07360 ;
              07361 ; (2)  GAMEOVER2 ($B121) is entered at the end of a successful mis
              07362 ;      Zylon ships destroyed). It puts the game in demo mode, enqu
              07363 ;      corresponding game over message, and calculates the scored
              07364 ;      class.
              07365 ;
              07366 ;      The scored rank and class are based on the total score. Thi
              07367 ;      accumulated during the game plus a mission bonus, which dep
              07368 ;      mission level and on how the mission ended (mission complet
              07369 ;      aborted, or starship destroyed by Zylon fire). The mission
              07370 ;      picked from table BONUSTAB ($BEDD).
              07371 ;
```

```
07372 ;      The scored rank index is taken from bits B8..4 of the total
07373 ;      limited to values of 0..18. It indexes table RANKTAB ($BEE9
07374 ;      string. The rank string is displayed in subroutine SETTITLE
07375 ;
07376 ;      The scored class index is taken from bits B3..0 (for rank i
07377 ;      11..14) and computed from bits B4..1 (for rank indices 1..1
07378 ;      It takes values of 0..15. It indexes table CLASSTAB ($BEFC)
07379 ;      digit. The class digit is displayed in subroutine SETTITLE
07380 ;
07381 ;      For quick lookup, the following table lists rank and class
07382 ;      score. Use the table as follows: Pick the cell with the clo
07383 ;      less or equal to your score then read the rank and class of
07384 ;      the top of the table, respectively.
07385 ;
07386 ;      For example: A score of 90 results in a ranking of "Novice
07387 ;      score of 161 results in a ranking of "Pilot Class 3".
07388 ;
07389 ; +-------------------------------+--------------------------------
07390 ; |     Minimum Total Score       |                          Class In
07391 ; |                               |  0   1   2   3   4   5   6   7
07392 ; +-------+-----------------------+--------------------------------
07393 ; | Rank  |                       |                              Class
07394 ; | Index |         Rank          |  5   5   5   4   4   4   4   3
07395 ; +-------+-----------------------+---+---+---+---+---+---+---+---+
07396 ; |     0 | Galactic Cook         |  0|  1|  2|  3|  4|  5|  6|  7|
07397 ; |     1 | Garbage Scow Captain  | 16| 18| 20| 22| 24| 26| 28| 30|
07398 ; |     2 | Garbage Scow Captain  |   |   |   |   |   |   |   |   |
07399 ; |     3 | Rookie                | 48| 50| 52| 54| 56| 58| 60| 62|
07400 ; |     4 | Rookie                |   |   |   |   |   |   |   |   |
07401 ; |     5 | Novice                | 80| 82| 84| 86| 88| 90| 92| 94|
07402 ; |     6 | Novice                |   |   |   |   |   |   |   |   |
07403 ; |     7 | Ensign                |112|114|116|118|120|122|124|126|
07404 ; |     8 | Ensign                |   |   |   |   |   |   |   |   |
07405 ; |     9 | Pilot                 |144|146|148|150|152|154|156|158|
07406 ; |    10 | Pilot                 |   |   |   |   |   |   |   |   |
07407 ; |    11 | Ace                   |176|177|178|179|180|181|182|183|
07408 ; |    12 | Lieutenant            |192|193|194|195|196|197|198|199|
07409 ; |    13 | Warrior               |208|209|210|211|212|213|214|215|
07410 ; |    14 | Captain               |224|225|226|227|228|229|230|231|
07411 ; |    15 | Commander             |240|242|244|246|248|250|252|254|
07412 ; |    16 | Commander             |   |   |   |   |   |   |   |   |
07413 ; |    17 | Star Commander        |272|274|276|278|280|282|284|286|
07414 ; |    18 | Star Commander        |   |   |   |   |   |   |   |   |
07415 ; +-------+-----------------------+---+---+---+---+---+---+---+---+
07416 ;
07417 ;      NOTE: This subroutine also clears the vertical and horizont
07418 ;      directions.
07419 ;
07420 ; INPUT
07421 ;
07422 ;   X = Offset to index table BONUSTAB ($BEDD) of mission bonus va
07423 ;       values are:
07424 ;     $00 -> Mission complete
07425 ;     $04 -> Mission was aborted due to zero energy
07426 ;     $08 -> Our starship was destroyed by Zylon fire
07427 ;
07428 ;   Y = Title phrase offset. Used values are:
07429 ;     $3F -> "MISSION COMPLETE"
07430 ;     $31 -> "MISSION ABORTED ZERO ENERGY"
```

```
                07431 ;     $23 -> "SHIP DESTROYED BY ZYLON FIRE"
                07432
                07433 ;*** Game over (Mission failed) **********************************
B10A A900       07434 GAMEOVER        LDA #0                  ;
B10C 85EC       07435                 STA PL3LIFE             ; PLAYER3 lifetime := 0 ga
B10E 85D6       07436                 STA BEEPPRIORITY        ; Mute beeper
B110 85D1       07437                 STA TITLEPHR            ; Clear title line
B112 858B       07438                 STA REDALERTLIFE        ; Red alert flash lifetime
B114 8D07D2     07439                 STA AUDC4               ; Mute audio channel 4
B117 8571       07440                 STA NEWVELOCITY         ; Shut down Engines
B119 8581       07441                 STA SHIELDSCOLOR        ; Set Shields color to {BL
B11B 857D       07442                 STA DRAINSHIELDS        ; Switch off Shields
B11D 85C0       07443                 STA WARPSTATE           ; Disengage hyperwarp
B11F 85C1       07444                 STA VELOCITYHI          ; Turn off hyperwarp veloc
                07445
                07446 ;*** Game over (Mission successful) ******************************
B121 A9FF       07447 GAMEOVER2       LDA #$FF                ; Enter demo mode
B123 8564       07448                 STA ISDEMOMODE          ;
                07449
B125 8465       07450                 STY NEWTITLEPHR         ; Enqueue title phrase
                07451
                07452 ;*** Calculate total score ***************************************
B127 8A         07453                 TXA                     ;
B128 0562       07454                 ORA MISSIONLEVEL        ;
B12A AA         07455                 TAX                     ;
B12B BDDDBE     07456                 LDA BONUSTAB,X          ; Retrieve mission bonus
B12E 18         07457                 CLC                     ; Add mission bonus and ga
B12F 65CB       07458                 ADC SCORE               ;
B131 AA         07459                 TAX                     ;
B132 A900       07460                 LDA #0                  ;
                07461
B134 85C9       07462                 STA JOYSTICKY           ; Clear vertical joystick
B136 85C8       07463                 STA JOYSTICKX           ; Clear horizontal joystic
                07464
B138 65CC       07465                 ADC SCORE+1             ;
B13A 3025       07466                 BMI SKIP165             ; Return if total score <
                07467
                07468 ;*** Calculate scored rank ***************************************
B13C 4A         07469                 LSR A                   ;
B13D 8A         07470                 TXA                     ;
B13E 6A         07471                 ROR A                   ;
B13F 4A         07472                 LSR A                   ;
B140 4A         07473                 LSR A                   ;
B141 4A         07474                 LSR A                   ; Use bits B8..4 of total
B142 C913       07475                 CMP #19                 ; Limit scored rank index
B144 9004       07476                 BCC SKIP162             ;
B146 A912       07477                 LDA #18                 ;
B148 A20F       07478                 LDX #15                 ; Prep class index of 15
B14A 85CD       07479 SKIP162         STA SCOREDRANKIND       ; Store scored rank index
                07480
                07481 ;*** Calculate scored class **************************************
B14C A8         07482                 TAY                     ;
B14D 8A         07483                 TXA                     ;
B14E C000       07484                 CPY #0                  ;
B150 F00B       07485                 BEQ SKIP164             ;
B152 C00B       07486                 CPY #11                 ;
B154 9004       07487                 BCC SKIP163             ;
B156 C00F       07488                 CPY #15                 ;
B158 9003       07489                 BCC SKIP164             ;
```

```
B15A 4A       07490 SKIP163         LSR A                   ;
B15B 4908     07491                 EOR #$08                ;
B15D 290F     07492 SKIP164         AND #$0F                ;
B15F 85CE     07493                 STA SCOREDCLASSIND      ; Store scored class index
              07494
B161 60       07495 SKIP165         RTS                     ; Return
              07496
              07497 ;*****************************************************************
              07498 ;*
              07499 ;*                                SELECTWARP
              07500 ;*
              07501 ;*             Select hyperwarp arrival location on Galactic Chart
              07502 ;*
              07503 ;*****************************************************************
              07504
              07505 ; DESCRIPTION
              07506 ;
              07507 ; This subroutine executes the following steps:
              07508 ;
              07509 ; (1)  Check if we are in Galactic Chart view and not in hyperwarp
              07510 ;
              07511 ; (2)  Update the Galactic Chart in subroutine DRAWGC ($B4B9) if t
              07512 ;      Radio is not damaged.
              07513 ;
              07514 ; (3)  Move the arrival hyperwarp marker (PLAYER4) across the Gala
              07515 ;      every other game loop iteration. The current location of ou
              07516 ;      indicated by the departure hyperwarp marker (PLAYER3).
              07517 ;
              07518 ; Code execution continues into subroutine CALCWARP ($B1A7) to cal
              07519 ; required hyperwarp energy to hyperwarp from the departure hyperw
              07520 ; position to the arrival hyperwarp marker position.
              07521 ;
              07522 ; NOTE: To calculate the horizontal position of PLAYER3..4 an offs
              07523 ; added (from left to right: 48 Player/Missile (PM) pixels to the
              07524 ; the screen + 16 PM pixels to the left border of the Galactic Cha
              07525 ; pixels relative offset of the PLAYER shape's horizontal center t
              07526 ; edge = 61 PM pixels).
              07527 ;
              07528 ; NOTE: To calculate the vertical position of PLAYER3..4 an offset
              07529 ; added (from top to bottom: 8 Player/Missile (PM) pixels to the s
              07530 ; Display List + 56 PM pixels to the first row of sectors - 1 PM p
              07531 ; offset of the PLAYER shape's vertical center to its top edge (?)
              07532 ; pixels).
              07533
B162 A5C0     07534 SELECTWARP      LDA WARPSTATE           ; Return if hyperwarp enga
B164 D004     07535                 BNE SKIP166             ;
              07536
B166 A5D0     07537                 LDA SHIPVIEW            ; Return if not in Galacti
B168 3001     07538                 BMI SKIP167             ;
B16A 60       07539 SKIP166         RTS                     ; Return
              07540
B16B 2C9709   07541 SKIP167         BIT GCSTATRAD           ; Skip if Subspace Radio i
B16E 3003     07542                 BMI SKIP168             ;
              07543
B170 20B9B4   07544                 JSR DRAWGC              ; Redraw Galactic Chart
              07545
B173 A572     07546 SKIP168         LDA COUNT8              ; Move hyperwarp markers o
B175 2901     07547                 AND #$01                ; (slowing down movement o
B177 D02E     07548                 BNE CALCWARP            ;
```

```
                07549
                07550 ;*** Calc arrival hyperwarp marker column and row numbers, update
B179 18         07551                 CLC                     ;
B17A A58F       07552                 LDA WARPARRVCOLUMN      ; Load arrival hyperwarp m
B17C 65C8       07553                 ADC JOYSTICKX           ; Add joystick x-delta
B17E 297F       07554                 AND #$7F                ; Limit value to 0..127
B180 858F       07555                 STA WARPARRVCOLUMN      ; Save new arrival hyperwa
B182 18         07556                 CLC                     ;
B183 693D       07557                 ADC #61                 ; Add offset of 61
B185 8D2E0C     07558                 STA PL4COLUMN           ; Store as PLAYER4 column
                07559
B188 18         07560                 CLC                     ;
B189 A58E       07561                 LDA WARPARRVROW         ; Load arrival hyperwarp m
B18B 65C9       07562                 ADC JOYSTICKY           ; Add joystick y-delta
B18D 297F       07563                 AND #$7F                ; Limit value to 0..127
B18F 858E       07564                 STA WARPARRVROW         ; Save new arrival hyperwa
B191 18         07565                 CLC                     ;
B192 693F       07566                 ADC #63                 ; Add offset of 63
B194 8DFD0B     07567                 STA PL4ROWNEW           ; Store as PLAYER4 row num
                07568
                07569 ;*** Calc departure hyperwarp marker column and row numbers, updat
B197 A58C       07570                 LDA WARPDEPRROW         ; Load departure hyperwarp
B199 18         07571                 CLC                     ;
B19A 693F       07572                 ADC #63                 ; Add offset of 63
B19C 8DFC0B     07573                 STA PL3ROWNEW           ; Store as PLAYER3 row num
                07574
B19F A58D       07575                 LDA WARPDEPRCOLUMN      ; Load departure hyperwarp
B1A1 18         07576                 CLC                     ;
B1A2 693D       07577                 ADC #61                 ; Add offset of 61
B1A4 8D2D0C     07578                 STA PL3COLUMN           ; Store as PLAYER3 column
                07579
                07580 ;*****************************************************************
                07581 ;*
                07582 ;*                                   CALCWARP
                07583 ;*
                07584 ;*                    Calculate and display hyperwarp energy
                07585 ;*
                07586 ;*****************************************************************
                07587
                07588 ; DESCRIPTION
                07589 ;
                07590 ; Calculates and displays the hyperwarp energy in the Galactic Cha
                07591 ;
                07592 ; This subroutine executes the following steps:
                07593 ;
                07594 ; (1)  Determine the arrival sector from the arrival hyperwarp mar
                07595 ;
                07596 ; (2)  If the Subspace Radio is not destroyed, update the target n
                07597 ;      the Galactic Chart Panel Display.
                07598 ;
                07599 ; (3)  Calculate the hyperwarp energy that is required to hyperwar
                07600 ;      departure hyperwarp marker to the arrival hyperwarp marker
                07601 ;      "block-distance":
                07602 ;
                07603 ;          DISTANCE := DELTAR / 2 + DELTAC
                07604 ;
                07605 ;          where
                07606 ;
                07607 ;          DELTAR := ABS(WARPARRVROW - WARPDEPRROW)
```

```
                07608 ;          DELTAC := ABS(WARPARRVCOLUMN - WARPDEPRCOLUMN)
                07609 ;
                07610 ;      NOTE: Dividing DELTAR by 2 compensates for PLAYERs at singl
                07611 ;      resolution having Player/Missile pixels that are half as hi
                07612 ;      wide.
                07613 ;
                07614 ;      The hyperwarp energy, divided by 10, is the sum of a value
                07615 ;      the hyperwarp energy table WARPENERGYTAB ($BADD) indexed by
                07616 ;      plus a remainder computed from Bits B1..0 of DISTANCE.
                07617 ;
                07618 ; (4)  Store the hyperwarp energy value in WARPENERGY ($91).
                07619 ;
                07620 ; (5)  Update the HYPERWARP ENERGY readout of the Galactic Chart P
                07621
    =006A       07622 L.WARPARRVCOL   = $6A                  ; Saves arrival sector col
    =006A       07623 L.DELTAC        = $6A                  ; Saves diff column value
                07624
                07625 ;*** Calculate arrival sector ************************************
B1A7 A58F       07626 CALCWARP        LDA WARPARRVCOLUMN     ;
B1A9 4A         07627                 LSR A                  ;
B1AA 4A         07628                 LSR A                  ;
B1AB 4A         07629                 LSR A                  ;
B1AC 856A       07630                 STA L.WARPARRVCOL      ; A := arrival sector colu
B1AE A58E       07631                 LDA WARPARRVROW        ;
B1B0 2970       07632                 AND #$70               ; A := arrival sector row
B1B2 056A       07633                 ORA L.WARPARRVCOL      ;
B1B4 8592       07634                 STA ARRVSECTOR         ; Save arrival sector (for
                07635
                07636 ;*** Update target number digit of Galactic Chart Panel Display **
B1B6 AA         07637                 TAX                    ;
B1B7 BDC908     07638                 LDA GCMEMMAP,X         ; Get number of Zylon ship
B1BA 1002       07639                 BPL SKIP169            ; Skip if no starbase in a
B1BC A900       07640                 LDA #0                 ; Clear number of Zylon sh
B1BE 0990       07641 SKIP169         ORA #CCS.COL2!ROM.0    ; Merge COLOR2 bits with n
B1C0 2C9709     07642                 BIT GCSTATRAD          ; Skip if Subspace Radio d
B1C3 7003       07643                 BVS SKIP170            ;
                07644
B1C5 8D8D09     07645                 STA GCTRGCNT           ; Set target number digit
                07646
                07647 ;*** Calculate energy to hyperwarp between hyperwarp markers *****
B1C8 38         07648 SKIP170         SEC                    ; A := DELTAC := ABS(WARPA
B1C9 A58F       07649                 LDA WARPARRVCOLUMN     ; (Column value difference
B1CB E58D       07650                 SBC WARPDEPRCOLUMN     ;
B1CD B004       07651                 BCS SKIP171            ;
B1CF 49FF       07652                 EOR #$FF               ;
B1D1 6901       07653                 ADC #1                 ;
B1D3 856A       07654 SKIP171         STA L.DELTAC           ;
                07655
B1D5 38         07656                 SEC                    ; A := DELTAR := ABS(WARPA
B1D6 A58E       07657                 LDA WARPARRVROW        ; (Row value difference)
B1D8 E58C       07658                 SBC WARPDEPRROW        ;
B1DA B004       07659                 BCS SKIP172            ;
B1DC 49FF       07660                 EOR #$FF               ;
B1DE 6901       07661                 ADC #1                 ;
                07662
B1E0 4A         07663 SKIP172         LSR A                  ; A := DISTANCE := DELTAR
B1E1 18         07664                 CLC                    ;
B1E2 656A       07665                 ADC L.DELTAC           ;
                07666
```

```
B1E4 A8       07667                TAY                    ; Save DISTANCE
B1E5 4A       07668                LSR A                  ; Calc index into hyperwar
B1E6 4A       07669                LSR A                  ;
B1E7 4A       07670                LSR A                  ;
B1E8 AA       07671                TAX                    ;
              07672
B1E9 98       07673                TYA                    ; Load DISTANCE value
B1EA 2903     07674                AND #$03               ; Get DISTANCE bits B1..0
B1EC 18       07675                CLC                    ;
B1ED 7DDDBA   07676                ADC WARPENERGYTAB,X    ; Add hyperwarp energy fro
B1F0 8591     07677                STA WARPENERGY         ; Save hyperwarp energy
              07678
              07679 ;*** Update HYPERWARP ENERGY readout of Galactic Chart Panel Displ
B1F2 A8       07680                TAY                    ; Prep with hyperwarp ener
              07681
B1F3 A910     07682                LDA #ROM.0             ; Set HYPERWARP ENERGY rea
B1F5 8D7D09   07683                STA GCWARPD1           ;
B1F8 8D7E09   07684                STA GCWARPD1+1         ;
B1FB 8D7F09   07685                STA GCWARPD1+2         ;
              07686
B1FE A202     07687 LOOP053        LDX #2                 ; Loop over HYPERWARP ENER
B200 FE7D09   07688 LOOP054        INC GCWARPD1,X         ; Increment digit value
B203 BD7D09   07689                LDA GCWARPD1,X         ;
B206 C91A     07690                CMP #ROM.9+1           ;
B208 9008     07691                BCC SKIP173            ; Skip if energy digit <=
              07692
B20A A910     07693                LDA #ROM.0             ; Replace energy digit wit
B20C 9D7D09   07694                STA GCWARPD1,X         ;
B20F CA       07695                DEX                    ;
B210 10EE     07696                BPL LOOP054            ; Next energy digit
              07697
B212 88       07698 SKIP173        DEY                    ; Decrement HYPERWARP ENER
B213 D0E9     07699                BNE LOOP053            ;
B215 60       07700                RTS                    ; Return
              07701
              07702 ;*****************************************************************
              07703 ;*
              07704 ;*                                UPDTITLE
              07705 ;*
              07706 ;*                             Update title line
              07707 ;*
              07708 ;*****************************************************************
              07709
              07710 ; DESCRIPTION
              07711 ;
              07712 ; Updates the title phrase displayed in the title line.
              07713 ;
              07714 ; If no title phrase has been set then fetch the offset of the nex
              07715 ; title phrase to be displayed. If one has been set then code exec
              07716 ; continues into subroutine SETTITLE ($B223), otherwise code execu
              07717 ;
              07718 ; If a title phrase has been set then decrement the lifetime of th
              07719 ; displayed title phrase segment. If its lifetime has reached a va
              07720 ; branch to subroutine SETTITLE ($B223) to display the next segmen
              07721
B216 A5D1     07722 UPDTITLE       LDA TITLEPHR           ; Skip if no title phrase
B218 F005     07723                BEQ SKIP175            ;
              07724
B21A C6CF     07725                DEC TITLELIFE          ; Decrement title phrase s
```

```
B21C F010     07726                BEQ SKIP176            ; If lifetime expired show
              07727
B21E 60       07728 SKIP174        RTS                    ; Return
              07729
B21F A465     07730 SKIP175        LDY NEWTITLEPHR        ; Prep enqueued new title
B221 F0FB     07731                BEQ SKIP174            ; Return if not set
              07732
              07733 ;*****************************************************************
              07734 ;*
              07735 ;*                                  SETTITLE
              07736 ;*
              07737 ;*                       Set title phrase in title line
              07738 ;*
              07739 ;*****************************************************************
              07740
              07741 ; DESCRIPTION
              07742 ;
              07743 ; Displays a title phrase in the title line.
              07744 ;
              07745 ; INTRODUCTION
              07746 ;
              07747 ; Title phrases are picked from the title phrase table PHRASETAB (
              07748 ; consist of one or more phrase tokens. Each token is a byte repre
              07749 ; in word table WORDTAB ($BC2B). Two special tokens are placeholde
              07750 ; scored class string ($FC) and scored rank string ($FD).
              07751 ;
              07752 ; A title phrase is split up into one or more title phrase segment
              07753 ; fitting into the title line. One title phrase segment is display
              07754 ; other after a delay called the "title segment lifetime".
              07755 ;
              07756 ; Phrase tokens, except the tokens for the scored class ($FC) and
              07757 ; rank ($FD), contain the number of a word in word table WORDTAB (
              07758 ; contain an end-of-segment or end-of-phrase marker bit.
              07759 ;
              07760 ; DETAILS
              07761 ;
              07762 ; The Display List is modified by subroutine MODDLST ($ADF1) to di
              07763 ; title line. Then, the title line is cleared and the words of the
              07764 ; are copied into it using the passed offset into title phrase tab
              07765 ; ($BBAA). If the offset has a value of $FF the title line is hidd
              07766 ; subroutine MODDLST ($ADF1).
              07767 ;
              07768 ; INPUT
              07769 ;
              07770 ;   Y = Offset into title phrase table PHRASETAB ($BBAA). Used val
              07771 ;     $FF  -> Hide title line
              07772 ;     else -> Offset into title phrase table PHRASETAB ($BBAA), wi
              07773 ;             used values:
              07774 ;
              07775 ;     $01 -> "COMPUTER ON"
              07776 ;     $04 -> "COMPUTER OFF"
              07777 ;     $07 -> "SHIELDS ON"
              07778 ;     $09 -> "SHIELDS OFF"
              07779 ;     $0B -> "COMPUTER TRACKING ON"
              07780 ;     $0E -> "TRACKING OFF"
              07781 ;     $13 -> "STARBASE SURROUNDED"
              07782 ;     $15 -> "STARBASE DESTROYED"
              07783 ;     $1F -> "DOCKING ABORTED"
              07784 ;     $21 -> "TRANSFER COMPLETE"
```

```
                07785 ;     $4A -> "NOVICE MISSION"
                07786 ;     $4C -> "PILOT MISSION"
                07787 ;     $4E -> "WARRIOR MISSION"
                07788 ;     $50 -> "COMMANDER MISSION"
                07789 ;     $52 -> "DAMAGE CONTROL..."
                07790 ;     $75 -> "RED ALERT"
                07791
    =006A       07792 L.WORD          = $6A                 ; Saves word number of WOR
                07793                                       ; are $00..$3F.
    =006B       07794 L.COLUMNPOS     = $6B                 ; Saves cursor column posi
                07795                                       ; into title line
    =006C       07796 L.TOKEN         = $6C                 ; Saves title phrase token
                07797                                       ; contains bit-encoded inf
                07798                                       ; the title phrase:
                07799                                       ; B7..6 = %00 -> Copy next
                07800                                       ; B7..6 = %01 -> End-of-ph
                07801                                       ;                delay, th
                07802                                       ;                segment l
                07803                                       ; B7..6 = %10 -> End-of-se
                07804                                       ;                segment l
                07805                                       ; B7..6 = %11 -> End-of-ph
                07806                                       ;                delay, th
                07807                                       ;                segment l
                07808                                       ;                Used with
                07809                                       ;                  "STARBA
                07810                                       ;                  "STARBA
                07811                                       ;                  "HYPERS
                07812                                       ;                  "RED AL
                07813                                       ; B5..0       -> Word numb
                07814
B223 84D1       07815 SETTITLE        STY TITLEPHR          ; Save title phrase offset
                07816
B225 A023       07817                 LDY #$23              ; Show title line
B227 A20F       07818                 LDX #$0F              ;
B229 A907       07819                 LDA #$07              ;
B22B 20F1AD     07820                 JSR MODDLST           ;
                07821
                07822 ;*** Init cursor column position and clear title line ************
B22E A213       07823 SKIP176         LDX #19               ; There are 19(+1) charact
B230 A900       07824                 LDA #0                ;
B232 856B       07825                 STA L.COLUMNPOS       ; Init cursor column posit
                07826
B234 9D1F0D     07827 LOOP055         STA TITLETXT,X        ; Clear character in title
B237 CA         07828                 DEX                   ;
B238 10FA       07829                 BPL LOOP055           ;
                07830
                07831 ;*** If title phrase offset = $FF then hide title line ***********
B23A A6D1       07832 SKIP177         LDX TITLEPHR          ; Load title phrase offset
B23C E6D1       07833                 INC TITLEPHR          ; Prepare title phrase off
B23E D009       07834                 BNE SKIP178           ; ...skip if it turned 0
                07835
B240 A20F       07836                 LDX #$0F              ; Remove title line and re
B242 A080       07837                 LDY #$80              ;
B244 A907       07838                 LDA #$07              ;
B246 4CF1AD     07839                 JMP MODDLST           ;
                07840
B249 BDAABB     07841 SKIP178         LDA PHRASETAB,X       ; Get phrase token
                07842
                07843 ;*** Display scored class? ************************************
```

```
B24C C9FC     07844                CMP #$FC               ; Skip if not "scored clas
B24E D00F     07845                BNE SKIP179            ;
              07846
B250 A4CE     07847                LDY SCOREDCLASSIND     ; Get scored class index,
B252 B9FCBE   07848                LDA CLASSTAB,Y         ; Load scored class number
B255 A66B     07849                LDX L.COLUMNPOS        ; Load cursor position
B257 9D1F0D   07850                STA TITLETXT,X         ; Store class in title lin
B25A A93C     07851                LDA #60                ; Title segment lifetime :
B25C 85CF     07852                STA TITLELIFE          ;
B25E 60       07853                RTS                    ; Return
              07854
              07855 ;*** Display scored rank? ***************************************
B25F C9FD     07856 SKIP179        CMP #$FD               ; Skip if not "scored rank
B261 D005     07857                BNE SKIP180            ;
              07858
B263 A4CD     07859                LDY SCOREDRANKIND      ; Get scored rank index, i
B265 B9E9BE   07860                LDA RANKTAB,Y          ; Load rank word number
              07861
              07862 ;*** Search word of token in word table *************************
B268 856C     07863 SKIP180        STA L.TOKEN            ; Save phrase token
B26A 293F     07864                AND #$3F               ; Strip bits B6..7 from ph
B26C 856A     07865                STA L.WORD             ; Store word number (bits
              07866
B26E A92A     07867                LDA #<[WORDTAB-1]      ; Point MEMPTR to WORDTAB-
B270 8568     07868                STA MEMPTR             ;
B272 A9BC     07869                LDA #>[WORDTAB-1]      ;
B274 8569     07870                STA MEMPTR+1           ;
              07871
B276 E668     07872 LOOP056        INC MEMPTR             ; Increment MEMPTR
B278 D002     07873                BNE SKIP181            ;
B27A E669     07874                INC MEMPTR+1           ;
              07875
B27C A000     07876 SKIP181        LDY #0                 ;
B27E B168     07877                LDA (MEMPTR),Y         ; Load character of word
B280 10F4     07878                BPL LOOP056            ; Loop until end-of-word m
B282 C66A     07879                DEC L.WORD             ;
B284 D0F0     07880                BNE LOOP056            ; Loop until word found
              07881
              07882 ;*** Copy word to title line, add space ************************
B286 293F     07883 LOOP057        AND #$3F               ; Strip color bits B6..7 f
B288 49A0     07884                EOR #CCS.COL2!$20      ; Merge COLOR2 bits and co
B28A A66B     07885                LDX L.COLUMNPOS        ; Copy character to title
B28C E66B     07886                INC L.COLUMNPOS        ; Increment cursor column
B28E 9D1F0D   07887                STA TITLETXT,X         ;
B291 C8       07888                INY                    ;
B292 B168     07889                LDA (MEMPTR),Y         ; Load next character of w
B294 10F0     07890                BPL LOOP057            ; Next character of word i
B296 E66B     07891                INC L.COLUMNPOS        ; Word was copied. Add spa
              07892
              07893 ;*** Decide to copy another word, etc. *************************
B298 A93C     07894                LDA #60                ; SUMMARY:
B29A 246C     07895                BIT L.TOKEN            ; If bits B7..6 of phrase
B29C 1004     07896                BPL SKIP182            ; %00 -> Copy next word to
B29E 5008     07897                BVC SKIP183            ; %01 -> End-of-phrase, sh
B2A0 A9FE     07898                LDA #254               ;        Title segment lif
B2A2 5096     07899 SKIP182        BVC SKIP177            ; %10 -> End-of-segment.
B2A4 A0FF     07900                LDY #$FF               ;        Title segment lif
B2A6 84D1     07901                STY TITLEPHR           ; %11 -> End-of-phrase, lo
B2A8 85CF     07902 SKIP183        STA TITLELIFE          ;        Title segment lif
```

```
B2AA 60       07903                 RTS                     ; Return
              07904
              07905 ;*****************************************************************
              07906 ;*
              07907 ;*                                  SOUND
              07908 ;*
              07909 ;*                            Handle sound effects
              07910 ;*
              07911 ;*****************************************************************
              07912
              07913 ; DESCRIPTION
              07914 ;
              07915 ; This subroutine handles the sound effects. It is called every ve
              07916 ; phase, that is, every TICK (1/60 s on an NTSC Atari 8-bit Home C
              07917 ; system, 1/50 s on a PAL Atari 8-bit Home Computer system) from t
              07918 ; Blank Interrupt handler VBIHNDLR ($A6D1).
              07919 ;
              07920 ; The game uses all of the available 4 audio channels: Audio chann
              07921 ; 3 are shared among the Engines sound effects and the "noise soun
              07922 ; (explosion and photon torpedo sound effects), while audio channe
              07923 ; for "beeper sound patterns" (status report sound effects). The f
              07924 ; sections explain the beeper sound patterns and the noise sound p
              07925 ;
              07926 ; o   BEEPER SOUND PATTERNS
              07927 ;
              07928 ;     There are the following beeper sound patterns:
              07929 ;
              07930 ;     (1)  HYPERWARP TRANSIT
              07931 ;     (2)  RED ALERT
              07932 ;     (3)  ACKNOWLEDGE
              07933 ;     (4)  DAMAGE REPORT
              07934 ;     (5)  MESSAGE FROM STARBASE
              07935 ;
              07936 ;     They are encoded in table BEEPPATTAB ($BF3E) in 6-byte long
              07937 ;     patterns".
              07938 ;
              07939 ;     Another table, BEEPFRQTAB ($BF5C), stores the frequencies fo
              07940 ;     of each beeper sound pattern, terminated by a marker byte ($
              07941 ;
              07942 ;     BUG (at $BF5C): The pattern frequencies in table BEEPFRQTAB
              07943 ;     offset $00 are unused. Suggested Fix: Remove from code.
              07944 ;
              07945 ;     Whenever the game calls subroutine BEEP ($B3A6), that subrou
              07946 ;     beeper sound pattern for playing by copying 6 bytes from the
              07947 ;     BEEPPATTAB ($BF3E) to BEEPFRQIND ($D2)..BEEPFRQSTART ($D7).
              07948 ;     SOUND ($B2AB) detects the copied beeper sound pattern and pl
              07949 ;     encoded tones and pauses.
              07950 ;
              07951 ;     The relevant variables for playing a beeper sound pattern ar
              07952 ;     following (see also figures at BEEPPATTAB ($BF3E)):
              07953 ;
              07954 ;     BEEPFRQIND    ($D2)   = Running index into table BEEPFRQTAB
              07955 ;     BEEPREPEAT    ($D3)   = Number of times that the beeper soun
              07956 ;                             repeated - 1
              07957 ;     BEEPTONELIFE  ($D4)   = Lifetime of tone in TICKs - 1
              07958 ;     BEEPPAUSELIFE ($D5)   = Lifetime of pause in TICKs - 1 ($FF
              07959 ;     BEEPPRIORITY  ($D6)   = Beeper sound pattern priority. A pla
              07960 ;                             sound pattern is stopped if a beeper
              07961 ;                             of higher priority is about to be pl
```

```
07962 ;                             of 0 indicates that no beeper sound
07963 ;                             playing at the moment.
07964 ;     BEEPFRQSTART  ($D7)   = Index to first byte of the beeper so
07965 ;                             table BEEPFRQTAB ($BF5C)
07966 ;
07967 ;     BEEPLIFE      ($D8)   = Lifetime of the current tone or paus
07968 ;     BEEPTOGGLE    ($D9)   = Indicates that either a tone (0) or
07969 ;                             0) is currently playing.
07970 ;
07971 ; o   NOISE SOUND PATTERNS
07972 ;
07973 ;     There are the following noise sound patterns:
07974 ;
07975 ;     (1)  PHOTON TORPEDO LAUNCHED
07976 ;     (2)  SHIELD EXPLOSION
07977 ;     (3)  ZYLON EXPLOSION
07978 ;
07979 ;     They are encoded in table NOISEPATTAB ($BF20) in 10-byte lon
07980 ;     patterns".
07981 ;
07982 ;     Whenever the game calls subroutine NOISE ($AEA8), that subro
07983 ;     a noise sound pattern for being played by copying 10 bytes f
07984 ;     pattern table NOISEPATTAB ($BF20) to NOISETORPTIM ($DA)..NOI
07985 ;     and hardware sound registers AUDCTL ($D208) and AUDF3 ($D204
07986 ;
07987 ;     The relevant variables for playing a noise sound pattern are
07988 ;     following:
07989 ;
07990 ;     NOISETORPTIM  ($DA)   = Delay timer for PHOTON TORPEDO LAUNC
07991 ;                             sound pattern
07992 ;     NOISEEXPLTIM  ($DB)   = Delay timer for SHIELD EXPLOSION and
07993 ;                             EXPLOSION noise sound patterns
07994 ;     NOISEAUDC2    ($DC)   = Audio channel 1/2 control shadow reg
07995 ;     NOISEAUDC3    ($DD)   = Audio channel 3   control shadow reg
07996 ;     NOISEAUDF1    ($DE)   = Audio channel 1 frequency shadow reg
07997 ;     NOISEAUDF2    ($DF)   = Audio channel 2 frequency shadow reg
07998 ;     NOISEFRQINC   ($E0)   = Audio channel 1/2 frequency incremen
07999 ;     NOISELIFE     ($E1)   = Noise sound pattern lifetime
08000 ;
08001 ;     AUDCTL        ($D208) = POKEY: Audio control
08002 ;     AUDF3         ($D204) = POKEY: Audio channel 3 frequency aud
08003 ;
08004 ;     There are two more variables that trigger noise effects. The
08005 ;     of the noise sound pattern table:
08006 ;
08007 ;     NOISEZYLONTIM ($E2)   = Delay timer to trigger the ZYLON EXP
08008 ;                             sound pattern. It is set in subrouti
08009 ;                             ($AF3D) when the impact of one of ou
08010 ;                             photon torpedoes with a target is im
08011 ;                             timer is decremented every TICK. Whe
08012 ;                             value of 0 the ZYLON EXPLOSION noise
08013 ;                             is played in subroutine SOUND ($B2AB
08014 ;     NOISEHITLIFE ($E3)    = Lifetime of the STARSHIP EXPLOSION n
08015 ;                             starship was destroyed by a Zylon ph
08016 ;                             It is set in GAMELOOP ($A1F3) to a v
08017 ;                             TICKs. When it reaches a value of 0
08018 ;                             EXPLOSION noise is played in subrout
08019 ;                             ($B2AB).
08020 ;
```

```
08021 ; SUBROUTINE DETAILS
08022 ;
08023 ; This subroutine executes the following steps:
08024 ;
08025 ; (1)  Play beeper sound pattern
08026 ;
08027 ;      The playing of a beeper sound pattern is started, continued
08028 ;
08029 ; (2)  Play ZYLON EXPLOSION noise sound pattern
08030 ;
08031 ;      If the explosion of a target space object is imminent (subr
08032 ;      COLLISION ($AF3D) has set NOISEZYLONTIM ($E2) to the number
08033 ;      iterations that will pass until our starship's photon torpe
08034 ;      the target), the timer NOISEZYLONTIM ($E2) is decremented e
08035 ;      it reaches a value of 0, then the noise sound pattern ZYLON
08036 ;      played.
08037 ;
08038 ; (3)  Play starship's Engines sound
08039 ;
08040 ;      If the Engines are louder than the current noise sound patt
08041 ;      noise sound pattern is terminated and the values for the au
08042 ;      1..3 are updated:
08043 ;
08044 ;      The velocity of our starship determines the pitch and the v
08045 ;      Engines: the higher the velocity, the higher the pitch and
08046 ;      the Engines. The incremented value of VELOCITYLO ($70) is u
08047 ;      value" %abcdefgh.
08048 ;
08049 ;      Audio channels 1 and 2 are combined to a 16-bit audio chann
08050 ;      clocked at 1.79 MHz. The inverted bits (represented by an o
08051 ;      B7..0 of the base value form bits B12..5 of the 16-bit freq
08052 ;      audio channel 1/2. Bits B7..4 of the base value form bits B
08053 ;      volume of audio channel 1/2, with noise distortion bit B7 s
08054 ;                                        ________
08055 ;      AUDF1/2 ($D202..3) := %000abcdefgh00000
08056 ;      AUDC2   ($D203)    := %1000abcd
08057 ;
08058 ;      Audio channel 3 is also clocked at 1.79 MHz. The inverted b
08059 ;      the base value form bits B7..0 of the frequency value of au
08060 ;      Bits B6..4 of the base value form bits B3..0 of the volume
08061 ;      channel 3, with noise distortion bit B7 set:
08062 ;                                 ________
08063 ;      AUDF3   ($D204)    := %abcdefgh
08064 ;      AUDC3   ($D205)    := %10000bcd
08065 ;
08066 ;      Code execution returns at this point.
08067 ;
08068 ; (4)  Play ZYLON EXPLOSION or SHIELD EXPLOSION noise sound patter
08069 ;
08070 ;      If the ZYLON EXPLOSION or SHIELD EXPLOSION noise sound patt
08071 ;      up, the explosion noise timer NOISEEXPLTIM ($DB) is decreme
08072 ;      TICK. It starts either with a value of 4 TICKs with a ZYLON
08073 ;      noise sound pattern or with a value of 8 TICKs with a SHIEL
08074 ;      noise sound pattern, set up in subroutine NOISE ($AEA8). If
08075 ;      value of 0, then the shadow control register of audio chann
08076 ;      switches to "noise distortion" at maximum volume.
08077 ;
08078 ; (5)  Play PHOTON TORPEDO LAUNCHED noise sound pattern
08079 ;
```

```
                    08080 ;      If the PHOTON TORPEDO LAUNCHED noise sound pattern was set
                    08081 ;      torpedo noise timer NOISETORPTIM ($DA) is decremented every
                    08082 ;      starts with a value of 8 TICKs, set in subroutine TRIGGER (
                    08083 ;      noise distortion and volume for the shadow control register
                    08084 ;      channel 3 is picked from table NOISETORPVOLTAB ($BFEB), the
                    08085 ;      frequency for audio channel 3 is picked from table NOISETOR
                    08086 ;      ($BFF3). If the photon torpedo noise timer reaches a value
                    08087 ;      shadow control registers of audio channel 1/2 switch to "to
                    08088 ;      at maximum volume and a frequency of $0202.
                    08089 ;
                    08090 ;      NOTE: Using a real-time volume envelope stored in table NOI
                    08091 ;      ($BFEB) for a launched photon torpedo results in producing
                    08092 ;      distinctive "whooshing" photon torpedo sound.
                    08093 ;
                    08094 ; (6)  Play STARSHIP EXPLOSION noise
                    08095 ;
                    08096 ;      If our starship was hit by a Zylon photon torpedo then NOIS
                    08097 ;      was set to 64 TICKs in routine GAMELOOP ($A1F3). While this
                    08098 ;      decremented every TICK, a random frequency value is stored
                    08099 ;      channel 3 and the distortion bit of the shadow control regi
                    08100 ;      channel 3 is randomly toggled.
                    08101 ;
                    08102 ; (7)  Increase audio channels 1/2 frequency
                    08103 ;
                    08104 ;      The 16-bit frequency value of audio channels 1/2 (both shad
                    08105 ;      and audio registers) is increased every TICK by an incremen
                    08106 ;      the currently playing noise sound pattern.
                    08107 ;
                    08108 ; (8)  Mute audio channels gradually
                    08109 ;
                    08110 ;      Toward the end of a noise sound pattern's lifetime all audi
                    08111 ;      gradually mute their volume every other TICK until complete
                    08112
                    08113 ;*** Play beeper sound pattern ********************************
B2AB A5D6           08114 SOUND           LDA BEEPPRIORITY       ; Skip if beeper sound pat
B2AD F037           08115                 BEQ SKIP185            ;
                    08116
B2AF C6D8           08117                 DEC BEEPLIFE           ; Decrement beeper lifetim
B2B1 1033           08118                 BPL SKIP185            ; Skip if beeper lifetime
                    08119
B2B3 A5D9           08120                 LDA BEEPTOGGLE         ; Load tone/pause toggle
B2B5 F00A           08121                 BEQ LOOP058            ; Skip if a tone is playin
                    08122
B2B7 A5D5           08123                 LDA BEEPPAUSELIFE      ; Load pause lifetime
B2B9 3006           08124                 BMI LOOP058            ; Skip if duration = $FF (
B2BB 85D8           08125                 STA BEEPLIFE           ; Store pause lifetime as
B2BD A000           08126                 LDY #0                 ; Prep AUDC4 (zero volume)
B2BF F020           08127                 BEQ SKIP184            ; Skip unconditionally
                    08128
B2C1 A5D4           08129 LOOP058         LDA BEEPTONELIFE       ; Load tone lifetime
B2C3 85D8           08130                 STA BEEPLIFE           ; Store tone lifetime as b
B2C5 A6D2           08131                 LDX BEEPFRQIND         ; Load frequency index
B2C7 E6D2           08132                 INC BEEPFRQIND         ; Increment frequency inde
B2C9 BD5CBF         08133                 LDA BEEPFRQTAB,X       ; Store tone frequency fro
B2CC 8D06D2         08134                 STA AUDF4              ;
B2CF A0A8           08135                 LDY #$A8               ; Prep AUDC4 (tone distort
B2D1 C9FF           08136                 CMP #$FF               ; Skip if frequency not $F
B2D3 D00C           08137                 BNE SKIP184            ;
                    08138
```

```
B2D5 A5D7     08139                LDA BEEPFRQSTART        ; Rewind pattern frequency
B2D7 85D2     08140                STA BEEPFRQIND          ;
B2D9 C6D3     08141                DEC BEEPREPEAT          ; Decrement sequence count
B2DB 10E4     08142                BPL LOOP058             ; Keep playing until seque
              08143
B2DD A000     08144                LDY #0                  ; Prep AUDC4 with zero vol
B2DF 84D6     08145                STY BEEPPRIORITY        ; Stop playing beeper soun
              08146
B2E1 8C07D2   08147 SKIP184        STY AUDC4               ; Store in AUDC4
B2E4 84D9     08148                STY BEEPTOGGLE          ; Store in BEEPTOGGLE
              08149
              08150 ;*** Play ZYLON EXPLOSION noise sound pattern ********************
B2E6 A5E2     08151 SKIP185        LDA NOISEZYLONTIM       ; Skip if ZYLON EXPLOSION
B2E8 F009     08152                BEQ SKIP186             ;
              08153
B2EA C6E2     08154                DEC NOISEZYLONTIM       ; Decrement ZYLON EXPLOSIO
B2EC D005     08155                BNE SKIP186             ; Skip if ZYLON EXPLOSION
              08156
B2EE A214     08157                LDX #$14                ; Play noise sound pattern
B2F0 20A8AE   08158                JSR NOISE               ;
              08159
              08160 ;*** Play our starship's Engines sound ***************************
B2F3 A670     08161 SKIP186        LDX VELOCITYLO          ; Skip if Engines softer t
B2F5 8A       08162                TXA                     ;
B2F6 4A       08163                LSR A                   ;
B2F7 4A       08164                LSR A                   ;
B2F8 4A       08165                LSR A                   ;
B2F9 4A       08166                LSR A                   ;
B2FA 4A       08167                LSR A                   ;
B2FB C5E1     08168                CMP NOISELIFE           ;
B2FD 902C     08169                BCC SKIP187             ;
              08170
B2FF A900     08171                LDA #0                  ; Terminate noise sound pa
B301 85E1     08172                STA NOISELIFE           ;
              08173
B303 E8       08174                INX                     ;
B304 8A       08175                TXA                     ; A := %abcdefgh = VELOCIT
B305 49FF     08176                EOR #$FF                ;        ________
B307 8D04D2   08177                STA AUDF3               ; AUDF3 := %abcdefgh
              08178
B30A AA       08179                TAX                     ;                 ________
B30B 0A       08180                ASL A                   ; AUDF2/1 := %000abcdefgh0
B30C 0A       08181                ASL A                   ;
B30D 0A       08182                ASL A                   ;
B30E 0A       08183                ASL A                   ;
B30F 0A       08184                ASL A                   ;
B310 8D00D2   08185                STA AUDF1               ;
B313 8A       08186                TXA                     ;
B314 4A       08187                LSR A                   ;
B315 4A       08188                LSR A                   ;
B316 4A       08189                LSR A                   ;
B317 8D02D2   08190                STA AUDF2               ;
              08191
B31A 4A       08192                LSR A                   ; AUDC2 := %1000abcd
B31B 498F     08193                EOR #$8F                ; (noise distortion + B7..
B31D 8D03D2   08194                STA AUDC2               ;
              08195
B320 2987     08196                AND #$87                ; AUDC3 := %10000bcd
B322 8D05D2   08197                STA AUDC3               ; (noise distortion + B6..
```

```
                08198
B325 A970       08199                   LDA #$70                ; Clock audio channel 1 an
B327 8D08D2     08200                   STA AUDCTL              ; ...combine audio channel
                08201
B32A 60         08202                   RTS                     ; Return
                08203
                08204 ;*** Play ZYLON EXPLOSION or SHIELD EXPLOSION noise **************
B32B A5DB       08205 SKIP187           LDA NOISEEXPLTIM        ; Skip if explosion noise
B32D F008       08206                   BEQ SKIP188             ;
                08207
B32F C6DB       08208                   DEC NOISEEXPLTIM        ; Decrement explosion nois
B331 D004       08209                   BNE SKIP188             ; Skip if explosion noise
                08210
B333 A98F       08211                   LDA #$8F                ; Shadow register AUDC2 :=
B335 85DC       08212                   STA NOISEAUDC2          ;
                08213
                08214 ;*** Play PHOTON TORPEDO LAUNCHED noise sound ********************
B337 A6DA       08215 SKIP188           LDX NOISETORPTIM        ; Skip if photon torpedo n
B339 F01C       08216                   BEQ SKIP190             ;
                08217
B33B C6DA       08218                   DEC NOISETORPTIM        ; Decrement photon torpedo
B33D D00A       08219                   BNE SKIP189             ; Skip if torpedo noise ti
                08220
B33F A9AF       08221                   LDA #$AF                ; Shadow register AUDC2 :=
B341 85DC       08222                   STA NOISEAUDC2          ;
B343 A902       08223                   LDA #$02                ; Set frequency $0202 to A
B345 85DE       08224                   STA NOISEAUDF1          ; ...registers
B347 85DF       08225                   STA NOISEAUDF2          ;
                08226
B349 BDEABF     08227 SKIP189           LDA NOISETORPVOLTAB-1,X ; Pick torpedo noise + vol
B34C 85DD       08228                   STA NOISEAUDC3          ; ...and store it in AUDC3
B34E BDF2BF     08229                   LDA NOISETORPFRQTAB-1,X ; Pick photon torpedo nois
B351 8D04D2     08230                   STA AUDF3               ; ...and store it in AUDF3
B354 8D09D2     08231                   STA STIMER              ; Reset POKEY audio timers
                08232
                08233 ;*** Play STARSHIP EXPLOSION noise when our starship is hit ******
B357 A5E3       08234 SKIP190           LDA NOISEHITLIFE        ; Skip if STARSHIP EXPLOSI
B359 F00E       08235                   BEQ SKIP191             ;
                08236
B35B C6E3       08237                   DEC NOISEHITLIFE        ; Decrement STARSHIP EXPLO
B35D AD0AD2     08238                   LDA RANDOM              ; Set random frequency to
B360 8D04D2     08239                   STA AUDF3               ;
B363 2920       08240                   AND #$20                ; Toggle noise/tone dist.
B365 45DD       08241                   EOR NOISEAUDC3          ; ...randomly
B367 85DD       08242                   STA NOISEAUDC3          ;
                08243
                08244 ;*** Increase 16-bit frequency of audio channels 1/2 (shadow regis
B369 18         08245 SKIP191           CLC                     ; Increase 16-bit frequenc
B36A A5DE       08246                   LDA NOISEAUDF1          ; ...and its shadow regist
B36C 65E0       08247                   ADC NOISEFRQINC         ; ...noise sound pattern f
B36E 85DE       08248                   STA NOISEAUDF1          ; AUDF1/2 := NOISEAUDF1/2
B370 8D00D2     08249                   STA AUDF1               ; ...NOISEAUDF1/2 + NOISEF
B373 A5DF       08250                   LDA NOISEAUDF2          ;
B375 6900       08251                   ADC #0                  ;
B377 85DF       08252                   STA NOISEAUDF2          ;
B379 8D02D2     08253                   STA AUDF2               ;
                08254
                08255 ;*** Gradually mute audio channels while noise sound pattern expir
B37C A6DC       08256                   LDX NOISEAUDC2          ; Prep AUDC2's shadow regi
```

```
B37E A4DD     08257                LDY NOISEAUDC3         ; Prep AUDC3's shadow regi
              08258
B380 A572     08259                LDA COUNT8             ; Decrement volumes every
B382 4A       08260                LSR A                  ;
B383 901A     08261                BCC SKIP193            ;
              08262
B385 A5E1     08263                LDA NOISELIFE          ; Skip if noise sound patt
B387 F016     08264                BEQ SKIP193            ;
              08265
B389 C6E1     08266                DEC NOISELIFE          ; Decrement noise sound pa
              08267
B38B C911     08268                CMP #17                ; Mute noise sound pattern
B38D B010     08269                BCS SKIP193            ; ...the last 16 TICKs of
              08270
B38F 8A       08271                TXA                    ; Decrement volume of AUDC
B390 290F     08272                AND #$0F               ;
B392 F003     08273                BEQ SKIP192            ;
B394 CA       08274                DEX                    ;
B395 86DC     08275                STX NOISEAUDC2         ;
              08276
B397 98       08277 SKIP192        TYA                    ; Decrement volume of AUDC
B398 290F     08278                AND #$0F               ;
B39A F003     08279                BEQ SKIP193            ;
B39C 88       08280                DEY                    ;
B39D 84DD     08281                STY NOISEAUDC3         ;
              08282
B39F 8E03D2   08283 SKIP193        STX AUDC2              ; Store shadow register va
B3A2 8C05D2   08284                STY AUDC3              ;
              08285
B3A5 60       08286                RTS                    ; Return
              08287
              08288 ;*****************************************************************
              08289 ;*
              08290 ;*                                  BEEP
              08291 ;*
              08292 ;*                          Copy beeper sound pattern
              08293 ;*
              08294 ;*****************************************************************
              08295
              08296 ; DESCRIPTION
              08297 ;
              08298 ; Copies a 6-byte beeper sound pattern from beeper sound pattern t
              08299 ; BEEPPATTAB ($BF3E) to BEEPFRQIND ($D2)..BEEPFRQSTART ($D7), prov
              08300 ; beeper sound pattern with higher priority is currently playing.
              08301 ; sound pattern will then be automatically played in subroutine SO
              08302 ; See subroutine SOUND ($B2AB) for more information on beeper soun
              08303 ;
              08304 ; NOTE: The bytes from table BEEPPATTAB ($BF3E) are copied in reve
              08305 ;
              08306 ; INPUT
              08307 ;
              08308 ;   X = Offset to beeper sound pattern in table BEEPPATTAB ($BF3E)
              08309 ;       are:
              08310 ;     $00 -> HYPERWARP TRANSIT
              08311 ;     $06 -> RED ALERT
              08312 ;     $0C -> ACKNOWLEDGE
              08313 ;     $12 -> DAMAGE REPORT
              08314 ;     $18 -> MESSAGE FROM STARBASE
              08315
```

```
B3A6 BD3EBF   08316 BEEP           LDA BEEPPATTAB,X       ; Return if beeper sound p
B3A9 C5D6     08317                CMP BEEPPRIORITY       ; ...higher priority is pl
B3AB 900C     08318                BCC SKIP194            ;
              08319
B3AD A005     08320                LDY #5                 ; Copy 6-byte beeper sound
B3AF BD3EBF   08321 LOOP059        LDA BEEPPATTAB,X       ;
B3B2 99D200   08322                STA BEEPFRQIND,Y       ;
B3B5 E8       08323                INX                    ;
B3B6 88       08324                DEY                    ;
B3B7 10F6     08325                BPL LOOP059            ;
              08326
B3B9 60       08327 SKIP194        RTS                    ; Return
              08328
              08329 ;*****************************************************************
              08330 ;*
              08331 ;*                                INITIALIZE
              08332 ;*
              08333 ;*                          More game initialization
              08334 ;*
              08335 ;*****************************************************************
              08336
              08337 ; DESCRIPTION
              08338 ;
              08339 ; This subroutine executes the following initialization steps:
              08340 ;
              08341 ; (1)  Set up Display List
              08342 ;
              08343 ;      A Display List is created at DSPLST ($0280). It starts with
              08344 ;      blank video lines, followed by 90 GRAPHICS7 rows. After a d
              08345 ;      in Display List instructions, which will be filled dynamica
              08346 ;      game by calls to subroutine MODDLST ($ADF1), it ends with a
              08347 ;      wait-and-jump-back instruction to the start of the Display
              08348 ;      ($0280).
              08349 ;
              08350 ;      NOTE: The PLAYFIELD color table PFCOLORTAB ($BFA9) is copie
              08351 ;      table PF0COLOR ($F2) by loop jamming.
              08352 ;
              08353 ; (2)  Create lookup tables
              08354 ;
              08355 ;      The first lookup table MAPTO80 ($0DE9) maps a byte value of
              08356 ;      0..80. This table is used to map unsigned (absolute) positi
              08357 ;      components (coordinates) to pixel (or PLAYER) row and colum
              08358 ;
              08359 ;      NOTE: The PLAYFIELD is 160 pixels wide. Pixel column number
              08360 ;      horizontal PLAYFIELD center are in -80..79, hence the range
              08361 ;      lookup table. Pixel row numbers relative the vertical PLAYF
              08362 ;      are in -50..49, thus they also fit in the range of this loo
              08363 ;
              08364 ;      The second lookup table MAPTOBCD99 ($0EE9) maps a byte valu
              08365 ;      a BCD-encoded value in 00..99. This table is used to conver
              08366 ;      into decimal 2-digit values displayed by the THETA (in "gra
              08367 ;      (in "gradons"), RANGE (in "centrons"), and VELOCITY (in "me
              08368 ;      second") readouts of the Console Panel Display.
              08369 ;
              08370 ;      The third and fourth lookup tables accelerate drawing of PL
              08371 ;      objects: In combination they contain the 16-bit start addre
              08372 ;      of the 100 rows of PLAYFIELD memory. The low bytes of the 1
              08373 ;      addresses are stored in table PFMEMROWLO ($0800). The high
              08374 ;      stored in table PFMEMROWHI ($0864).
```

```
                08375 ;
                08376 ;       NOTE: The address increment is 40 (40 bytes = 160 pixels in
                08377 ;       mode = 1 PLAYFIELD row of pixels).
                08378 ;
                08379 ;       NOTE: The PLAYFIELD consists of 90 GRAPHICS7 rows when the
                08380 ;       Display is displayed at the bottom. When the Control Panel
                08381 ;       displayed, for example in demo mode, the PLAYFIELD contains
                08382 ;       GRAPHICS7 rows.
                08383 ;
                08384 ; (3)  Copy Control Panel Display and Galactic Chart Panel Display
                08385 ;
                08386 ;       The texts of the Control Panel Display and the Galactic Cha
                08387 ;       Display are copied to their respective locations in memory
                08388 ;       jamming.
                08389 ;
                08390 ; (4)  Initialize Zylon unit movement timer
                08391 ;
                08392 ;       The timer that triggers the movement of Zylon units in the
                08393 ;       is initialized to a value of 99. See subroutine FLUSHGAMELO
                08394 ;       more information on Zylon unit movement.
                08395 ;
                08396 ; (5)  Create Galactic Chart
                08397 ;
                08398 ;       The Galactic Chart memory map GCMEMMAP ($08C9) is initializ
                08399 ;       represents 16 columns x 8 rows of sectors. Each sector cont
                08400 ;       the 4 sector types stored in table SECTORTYPETAB ($BBA6) (s
                08401 ;       Zylon ships, 3 Zylon ships, and 2 or 1 Zylon ships), and em
                08402 ;       Before distributing the sector types, the initial position
                08403 ;       starship is blocked on the Galactic Chart at sector row 4,
                08404 ;       8, so that it will not be inadvertently occupied while othe
                08405 ;       are distributed. The number of each of the sector types to
                08406 ;       is the mission level plus 2. While Zylon units can be place
                08407 ;       starbases are placed neither at the borders of the Galactic
                08408 ;       a sector adjacent to an occupied sector.
                08409 ;
                08410 ;       After having initialized the Galactic Chart memory map, the
                08411 ;       the Galactic Chart is drawn with characters from the custom
                08412 ;       set.
                08413 ;
                08414 ;       Finally, the sector in which our starship is located and th
                08415 ;       departure hyperwarp marker column and row numbers are initi
                08416 ;
                08417 ; (6)  Apply a final tweak
                08418 ;
                08419 ;       The last entry of lookup table MAPTOBCD99 ($0EE9) is tweake
                08420 ;       of CCS.INF * 16 + CCS.SPC. It is used to display an infinit
                08421 ;       the RANGE readout of the Control Panel Display in subroutin
                08422 ;       ($B8A7).
                08423 ;
                08424 ; Code execution continues into subroutine DRAWGC ($B4B9), which d
                08425 ; content of the Galactic Chart with characters from the custom ch
                08426
=0068           08427 L.MEMPTR1       = $68                 ; 16-bit memory pointer
=006A           08428 L.MEMPTR2       = $6A                 ; 16-bit memory pointer
=006A           08429 L.SECTORTYPE    = $6A                 ; Saves sector type. Used
                08430                                       ;   $CF -> Sector contains
                08431                                       ;   $04 -> Sector contains
                08432                                       ;   $03 -> Sector contains
                08433                                       ;   $02 -> Sector contains
```

```
    =006B     08434 L.SECTORCNT     = $6B                   ; Saves number of sectors
              08435
              08436 ;*** Initialize Display List and copy color table ***************
B3BA A259     08437 INITIALIZE      LDX #89                 ; Set 89(+1) GRAPHICS7 row
B3BC A90D     08438 LOOP060         LDA #$0D                ; Prep DL instruction $0D
B3BE 9D8502   08439                 STA DSPLST+5,X          ; DSPLST+5,X := one row of
B3C1 E00A     08440                 CPX #10                 ;
B3C3 B005     08441                 BCS SKIP195             ;
B3C5 BDA9BF   08442                 LDA PFCOLORTAB,X        ; Copy PLAYFIELD color tab
B3C8 95F2     08443                 STA PF0COLOR,X          ; (loop jamming)
B3CA CA       08444 SKIP195         DEX                     ;
B3CB 10EF     08445                 BPL LOOP060             ;
              08446
B3CD A970     08447                 LDA #$70                ; DSPLST     := BLK8
B3CF 8D8002   08448                 STA DSPLST              ; DSPLST+1   := BLK8
B3D2 8D8102   08449                 STA DSPLST+1            ;
B3D5 A941     08450                 LDA #$41                ; DSPLST+103 := WAITJMP @
B3D7 8DE702   08451                 STA DSPLST+103          ;
B3DA A980     08452                 LDA #<DSPLST            ;
B3DC 8DE802   08453                 STA DSPLST+104          ;
B3DF A902     08454                 LDA #>DSPLST            ;
B3E1 8DE902   08455                 STA DSPLST+105          ;
              08456
              08457 ;*** Calculate lookup tables ************************************
B3E4 A200     08458                 LDX #0                  ; Clear both 16-bit memory
B3E6 8668     08459                 STX L.MEMPTR1           ;
B3E8 8669     08460                 STX L.MEMPTR1+1         ;
B3EA 866A     08461                 STX L.MEMPTR2           ;
B3EC 866B     08462                 STX L.MEMPTR2+1         ;
              08463
              08464 ;*** Calc MAPTO80 map (converts value of $00..$FF to value in 0..8
B3EE 18       08465 LOOP061         CLC                     ;
B3EF A568     08466                 LDA L.MEMPTR1           ;
B3F1 6951     08467                 ADC #81                 ;
B3F3 8568     08468                 STA L.MEMPTR1           ;
B3F5 A569     08469                 LDA L.MEMPTR1+1         ;
B3F7 9DE90D   08470                 STA MAPTO80,X           ;
B3FA 6900     08471                 ADC #0                  ;
B3FC 8569     08472                 STA L.MEMPTR1+1         ;
              08473
              08474 ;*** Calc MAPTOBCD99 map (converts value of $00..$FF to BCD-value
B3FE 18       08475                 CLC                     ;
B3FF A56A     08476                 LDA L.MEMPTR2           ;
B401 6964     08477                 ADC #100                ;
B403 856A     08478                 STA L.MEMPTR2           ;
B405 A56B     08479                 LDA L.MEMPTR2+1         ;
B407 9DE90E   08480                 STA MAPTOBCD99,X        ;
B40A F8       08481                 SED                     ;
B40B 6900     08482                 ADC #0                  ;
B40D D8       08483                 CLD                     ;
B40E 856B     08484                 STA L.MEMPTR2+1         ;
B410 E8       08485                 INX                     ;
B411 D0DB     08486                 BNE LOOP061             ;
              08487
              08488 ;*** Calculate PLAYFIELD memory row addresses, copy Panel Display
B413 A200     08489                 LDX #<PFMEM             ; Point L.MEMPTR1 to start
B415 8668     08490                 STX L.MEMPTR1           ; (X = 0, because PFMEM is
B417 A910     08491                 LDA #>PFMEM             ;
B419 8569     08492                 STA L.MEMPTR1+1         ;
```

```
            08493
B41B 18     08494 LOOP062       CLC                 ;
B41C A568   08495               LDA L.MEMPTR1       ;
B41E 9D0008 08496               STA PFMEMROWLO,X    ; Store 16-bit value of L.
B421 6928   08497               ADC #40             ; Add 40 to L.MEMPTR
B423 8568   08498               STA L.MEMPTR1       ; (40 bytes = 160 pixels =
B425 A569   08499               LDA L.MEMPTR1+1     ;
B427 9D6408 08500               STA PFMEMROWHI,X    ;
B42A 6900   08501               ADC #0              ;
B42C 8569   08502               STA L.MEMPTR1+1     ;
            08503
B42E BD42BB 08504               LDA PANELTXTTAB,X   ; Copy Control and Galacti
B431 9D4909 08505               STA PANELTXT,X      ; (loop jamming)
            08506
B434 E8     08507               INX                 ;
B435 E064   08508               CPX #100            ;
B437 90E2   08509               BCC LOOP062         ; Loop 100 times
            08510
            08511 ;*** Set Zylon unit movement timer ****************************
B439 CA     08512               DEX                 ;
B43A 8678   08513               STX ZYLONUNITTIM    ; Init Zylon unit movement
            08514
            08515 ;*** Create memory map of the Galactic Chart ********************
B43C A203   08516               LDX #3              ; Loop over all 3(+1) sect
B43E 8E1109 08517               STX GCMEMMAP+4*16+8 ; Block our starship's ini
            08518                                   ; ...Galactic Chart (secto
            08519
B441 BDA6BB 08520 LOOP063       LDA SECTORTYPETAB,X ; Prep sector type
B444 856A   08521               STA L.SECTORTYPE    ;
            08522
B446 A462   08523               LDY MISSIONLEVEL    ; Number sectors of curren
B448 C8     08524               INY                 ;
B449 C8     08525               INY                 ;
B44A 846B   08526               STY L.SECTORCNT     ;
            08527
B44C AD0AD2 08528 LOOP064       LDA RANDOM          ; Load random sector 0..12
B44F 297F   08529               AND #$7F            ;
B451 A8     08530               TAY                 ;
B452 B9C908 08531               LDA GCMEMMAP,Y      ;
B455 D0F5   08532               BNE LOOP064         ; If sector already occupi
            08533
B457 A56A   08534               LDA L.SECTORTYPE    ; Reload sector type
B459 1021   08535               BPL SKIP196         ; Skip if sector not to be
            08536
B45B C010   08537               CPY #$10            ; Place starbase...
B45D 90ED   08538               BCC LOOP064         ; ...not in first sector r
B45F C070   08539               CPY #$70            ;
B461 B0E9   08540               BCS LOOP064         ; ...not in last sector ro
B463 98     08541               TYA                 ;
B464 290F   08542               AND #$0F            ;
B466 F0E4   08543               BEQ LOOP064         ; ...not in first sector c
B468 C90F   08544               CMP #15             ;
B46A F0E0   08545               BEQ LOOP064         ; ...not in last sector co
B46C B9C808 08546               LDA GCMEMMAP-1,Y    ; ...not east  of an occup
B46F 19CA08 08547               ORA GCMEMMAP+1,Y    ; ...not west  of an occup
B472 19D908 08548               ORA GCMEMMAP+16,Y   ; ...not south of an occup
B475 19B908 08549               ORA GCMEMMAP-16,Y   ; ...not north of an occup
B478 D0D2   08550               BNE LOOP064         ;
            08551
```

```
B47A A56A     08552                LDA L.SECTORTYPE        ; Reload sector type
              08553
B47C 99C908   08554 SKIP196        STA GCMEMMAP,Y          ; Store sector type in Gal
B47F C66B     08555                DEC L.SECTORCNT         ;
B481 10C9     08556                BPL LOOP064             ; Next sector
B483 CA       08557                DEX                     ;
B484 10BB     08558                BPL LOOP063             ; Next sector type
              08559
              08560 ;*** Clear Galactic Chart and draw top border ******************
B486 A2B4     08561                LDX #180                ; Clear Galactic Chart PLA
B488 A90A     08562 LOOP065        LDA #CCS.SPC            ;
B48A 9D340D   08563                STA GCPFMEM-1,X         ;
B48D CA       08564                DEX                     ;
B48E D0F8     08565                BNE LOOP065             ;
              08566
B490 A20F     08567                LDX #15                 ; Draw top border (15(+1)
B492 A918     08568 LOOP066        LDA #CCS.BORDERS        ;
B494 9D370D   08569                STA GCPFMEM+2,X         ;
B497 CA       08570                DEX                     ;
B498 10F8     08571                BPL LOOP066             ;
              08572
B49A A91A     08573                LDA #CCS.CORNERSW       ; Draw NORTHEAST corner (1
B49C 8D470D   08574                STA GCPFMEM+18          ;
              08575
B49F A900     08576                LDA #0                  ; Release starship's posit
B4A1 8D1109   08577                STA GCMEMMAP+4*16+8     ; ...Chart (sector row 4,
              08578
              08579 ;*** Initialize current sector and hyperwarp marker column and row
B4A4 A948     08580                LDA #$48                ; Place our starship's cur
B4A6 8590     08581                STA CURRSECTOR          ; ...sector row 4, sector
B4A8 A943     08582                LDA #$43                ; Init departure & arrival
B4AA 858D     08583                STA WARPDEPRCOLUMN      ;
B4AC 858F     08584                STA WARPARRVCOLUMN      ;
B4AE A947     08585                LDA #$47                ; Init departure & arrival
B4B0 858E     08586                STA WARPARRVROW         ;
B4B2 858C     08587                STA WARPDEPRROW         ;
              08588
              08589 ;*** Tweak last entry of MAPTOBCD99 ****************************
B4B4 A9EA     08590                LDA #CCS.INF*16+CCS.SPC ; Last entry of MAPTOBCD99
B4B6 8DE80F   08591                STA MAPTOBCD99+255      ;
              08592
              08593 ;*****************************************************************
              08594 ;*
              08595 ;*                                  DRAWGC
              08596 ;*
              08597 ;*                             Draw Galactic Chart
              08598 ;*
              08599 ;*****************************************************************
              08600
              08601 ; DESCRIPTION
              08602 ;
              08603 ; Draws the content of the Galactic Chart memory map in GCMEMMAP (
              08604 ; Galactic Chart PLAYFIELD memory at GCPFMEM ($0D35).
              08605 ;
              08606 ; NOTE: CPU register X indexes the Galactic Chart memory map GCMEM
              08607 ; (16 x 8 bytes). CPU register Y indexes the Galactic Chart PLAYFI
              08608 ; GCPFMEM ($0D35) (20 x 9 bytes).
              08609 ;
              08610 ; NOTE: Sectors with 1 or 2 Zylon ships display the same symbol in
```

```
                08611 ; Chart.
                08612
    =006A       08613 L.GCMEMMAPIND   = $6A                   ; Saves Galactic Chart mem
                08614
B4B9 A000       08615 DRAWGC          LDY #0                  ; Clear Galactic Chart PLA
B4BB 846A       08616                 STY L.GCMEMMAPIND       ; Clear Galactic Chart mem
                08617
B4BD A66A       08618 LOOP067         LDX L.GCMEMMAPIND       ; Load sector of Galactic
B4BF BDC908     08619                 LDA GCMEMMAP,X          ;
B4C2 1002       08620                 BPL SKIP197             ; Skip if not a starbase s
B4C4 A905       08621                 LDA #5                  ; Prep sector character in
                08622
B4C6 AA         08623 SKIP197         TAX                     ; Load sector character in
B4C7 BDD1BE     08624                 LDA SECTORCHARTAB,X     ; Load custom character se
B4CA 994B0D     08625                 STA GCPFMEM+22,Y        ; ...and store it in Galac
B4CD C8         08626                 INY                     ; Increment Galactic Chart
B4CE E66A       08627                 INC L.GCMEMMAPIND       ; Increment Galactic Chart
B4D0 A56A       08628                 LDA L.GCMEMMAPIND       ;
B4D2 290F       08629                 AND #$0F                ;
B4D4 D0E7       08630                 BNE LOOP067             ; Next sector column until
                08631
B4D6 A919       08632                 LDA #CCS.BORDERW        ; Draw right border
B4D8 994B0D     08633                 STA GCPFMEM+22,Y        ;
                08634
B4DB C8         08635                 INY                     ; Adjust Galactic Chart PL
B4DC C8         08636                 INY                     ;
B4DD C8         08637                 INY                     ;
B4DE C8         08638                 INY                     ;
B4DF C0A0       08639                 CPY #$A0                ;
B4E1 90DA       08640                 BCC LOOP067             ; Next sector until bottom
                08641
B4E3 60         08642                 RTS                     ; Return
                08643
                08644 ;*****************************************************************
                08645 ;*
                08646 ;*                                FLUSHGAMELOOP
                08647 ;*
                08648 ;*         Handle remaining tasks at the end of a game loop iterat
                08649 ;*
                08650 ;*****************************************************************
                08651
                08652 ; DESCRIPTION
                08653 ;
                08654 ; This subroutine handles at the end of a game loop iteration the
                08655 ; tasks:
                08656 ;
                08657 ; (1)  Increment counters COUNT256 ($76) and COUNT8 ($72).
                08658 ;
                08659 ; (2)  If our starship's energy has dropped below 1000 units then
                08660 ;      alert that changes to {DARK GREY BLUE} and back to {PINK} e
                08661 ;      loop iterations.
                08662 ;
                08663 ; (3)  Set the Shields color and the Control Panel background colo
                08664 ;      loop iterations:
                08665 ;
                08666 ;      o   If the Shields are up and OK then set the Shields color
                08667 ;          GREEN} and the Control Panel background color to {DARK
                08668 ;
                08669 ;      o   If the Shields are up and damaged there is a probabilit
```

```
08670 ;          (200:256) that the Shield color is not changed.
08671 ;
08672 ;      o   If the Shields are down, damaged, or destroyed then set
08673 ;          color to {BLACK}.
08674 ;
08675 ;      o   If the Shields are destroyed then set the Control Panel
08676 ;          color to {ORANGE}.
08677 ;
08678 ; (4)  Decrement the lifetime of our starship's and Zylon photon t
08679 ;
08680 ; (5)  Decrement the lifetime of an explosion. If the explosion li
08681 ;      than 112 game loop iterations, clear HITBADNESS ($8A) (thus
08682 ;      cannot destroy our starship). If the explosion lifetime is
08683 ;      (?) game loops decrement the number of explosion fragments.
08684 ;      explosion fragments disappear gradually toward the end of a
08685 ;
08686 ; (6)  Increment every 40 game loop iterations the stardate clock
08687 ;      Galactic Chart Panel Display.
08688 ;
08689 ; (7)  Move Zylon units in the Galactic Chart.
08690 ;
08691 ;      Every 50 game loop iterations (or 100 game loop iterations
08692 ;      starbase is surrounded by Zylon units) decrement the score.
08693 ;
08694 ; Code execution continues if the game is not in demo mode with th
08695 ; steps:
08696 ;
08697 ; (1)  Search the Galactic Chart for starbases surrounded by Zylon
08698 ;      Destroy any such starbase: Replace it with a 2-Zylon sector
08699 ;      18 points from the score. If the Subspace Radio was not des
08700 ;      flash the title phrase "STARBASE DESTROYED" and play the be
08701 ;      pattern MESSAGE FROM STARBASE in subroutine BEEP ($B3A6).
08702 ;
08703 ; (2)  Every 8 game loop iterations the Zylon units decide, which
08704 ;      hunt: First, 128 randomly picked sectors are searched for a
08705 ;      no starbase was found in this way, the sectors of the Galac
08706 ;      scanned systematically left-to-right, top-to-bottom. If a s
08707 ;      found then its sector, sector column, and sector row are sa
08708 ;      code execution returns.
08709 ;
08710 ; (3)  Now the Zylon units converge toward the sector of the hunte
08711 ;      All sectors of the Galactic Chart are scanned. For any sect
08712 ;      Zylon unit that was not moved yet (its sector does not have
08713 ;      "already-moved" bit B5 set) its movement probability value
08714 ;      table MOVEPROBTAB ($BFBB):
08715 ;
08716 ;      +---------------+-------------+----------------+
08717 ;      |  Sector Type  |  Movement   |   Movement     |
08718 ;      |               | Probability |  Probability   |
08719 ;      |               |    Value    |                |
08720 ;      +---------------+-------------+----------------+
08721 ;      | Empty sector  |       0     |   0% (  0:256) |
08722 ;      | 1 Zylon ship  |     255     | 100% (255:256) |
08723 ;      | 2 Zylon ships |     255     | 100% (255:256) |
08724 ;      | 3 Zylon ships |     192     |  75% (192:256) |
08725 ;      | 4 Zylon ships |      32     |  13% ( 32:256) |
08726 ;      +---------------+-------------+----------------+
08727 ;
08728 ;      If this value is less or equal than a random number in 0..2
```

```
          08729 ;      Zylon unit is moved to another sector. A Zylon unit that cu
          08730 ;      occupies the sector of our starship is not moved.
          08731 ;
          08732 ;      BUG (at $B620): The instruction to check the marker bit B5
          08733 ;      is CPY #$0A. This works, as sectors containing Zylon units
          08734 ;      be moved have values of 2..4, see table SECTORTYPETAB ($BBA
          08735 ;      fix: Replace CPY #$0A with CPY #$20, which may make the cod
          08736 ;
          08737 ; (4)  For every Zylon unit that is about to be moved, 9 distances
          08738 ;      distances") between the Zylon unit and the starbase are cal
          08739 ;      tentatively moving the Zylon unit into each of its 8 adjace
          08740 ;      and by moving it not at all. The sector offsets are taken f
          08741 ;      COMPASSOFFTAB ($BFC0) which stores direction offsets in the
          08742 ;      order: NORTH, NORTHWEST, WEST, SOUTHWEST, SOUTH, SOUTHEAST,
          08743 ;      NORTHEAST, CENTER. All 9 distances are stored in 9 consecut
          08744 ;      NEWZYLONDIST ($96).
          08745 ;
          08746 ;      NOTE: The last calculated distance is the current distance
          08747 ;      unit and starbase.
          08748 ;
          08749 ;      The Zylon unit moves to the first of the 8 adjacent sectors
          08750 ;      the following conditions:
          08751 ;
          08752 ;      (1)  It is closer to the starbase than the Zylon unit's cur
          08753 ;
          08754 ;      (2)  It is located inside the Galactic Chart.
          08755 ;
          08756 ;      (3)  It is empty.
          08757 ;
          08758 ;      (4)  It is not the sector containing our starship.
          08759 ;
          08760 ;      If a suitable new sector was found then the Zylon unit is m
          08761 ;      sector, which is marked with the "already-moved" marker bit
          08762 ;      Galactic Chart memory map. This marker bit prevents a Zylon
          08763 ;      been already moved from being moved again. The old Zylon un
          08764 ;      cleared.
          08765 ;
          08766 ;      If no suitable new sector was found then the above distance
          08767 ;      are repeated once again by adding 1 to the current distance
          08768 ;      Zylon unit and the starbase. This may provoke a Zylon unit
          08769 ;      would not have moved in the previous set of distance calcul
          08770 ;
          08771 ;      After having moved all Zylon units the sectors are stripped
          08772 ;      "already-moved" marker bit B5.
          08773 ;
          08774 ; (5)  If a starbase has been surrounded then the Zylon unit movem
          08775 ;      reset to 99, buying our starship some time to destroy one o
          08776 ;      surrounding Zylon units. If the Subspace Radio is not destr
          08777 ;      message "STARBASE SURROUNDED" is flashed in the title line
          08778 ;      sound pattern MESSAGE FROM STARBASE is played in subroutine
          08779
=006A     08780 L.ISDESTROYED   = $6A                ; Flags the destruction of
          08781                                      ; Used values are:
          08782                                      ;   $00 -> Starbase not de
          08783                                      ;   $02 -> Starbase has be
=006A     08784 L.NEWSECTOR     = $6A                ; Sector to which the Zylo
=006B     08785 L.ABSDIFFCOLUMN = $6B                ; Absolute difference betw
          08786                                      ;   column on Galactic Cha
=006B     08787 L.LOOPCNT2      = $6B                ; Loop counter. Used value
```

```
    =006A     08788 L.DIRECTIONIND  = $6A                ; Compass rose direction i
              08789                                      ; Used values are: 0..7.
              08790
              08791 ;*** Increment counters and flash low-energy alert **************
B4E4 E676     08792 FLUSHGAMELOOP   INC COUNT256         ; Increment COUNT256 count
              08793
B4E6 A290     08794                 LDX #$90             ; Prep DLI background colo
B4E8 A576     08795                 LDA COUNT256         ;
B4EA 1009     08796                 BPL SKIP198          ; Skip if counter < 128.
              08797
B4EC AC5509   08798                 LDY ENERGYD1         ; When energy drops below
B4EF C080     08799                 CPY #CCS.COL2!CCS.0  ;
B4F1 D002     08800                 BNE SKIP198          ;
B4F3 A244     08801                 LDX #$44             ; ...prep new DLI backgrou
              08802
B4F5 2903     08803 SKIP198         AND #$03             ; Increment COUNT8
B4F7 8572     08804                 STA COUNT8           ;
B4F9 D01F     08805                 BNE SKIP202          ; Skip setting colors but
              08806
              08807 ;*** Set Shields and Control Panel background color **************
B4FB A47D     08808                 LDY DRAINSHIELDS     ; Skip if Shields are off
B4FD F017     08809                 BEQ SKIP201          ;
              08810
B4FF A0A0     08811                 LDY #$A0             ; Prep Shields color {DARK
B501 2C9409   08812                 BIT GCSTATSHL        ; Skip if Shields are OK
B504 100B     08813                 BPL SKIP200          ;
B506 7007     08814                 BVS SKIP199          ; Skip if Shields are dest
B508 AD0AD2   08815                 LDA RANDOM           ; If Shields are damaged,
B50B C9C8     08816                 CMP #200             ; ...unchanged with probab
B50D 9007     08817                 BCC SKIP201          ;
              08818
B50F A000     08819 SKIP199         LDY #$00             ; Prep Shields color {BLAC
B511 98       08820 SKIP200         TYA                  ;
B512 D002     08821                 BNE SKIP201          ;
              08822
B514 A226     08823                 LDX #$26             ; Prep Control Panel backg
              08824
B516 8481     08825 SKIP201         STY SHIELDSCOLOR     ; Store Shields color
B518 86FB     08826                 STX BGRCOLORDLI      ; Store Control Panel back
              08827
              08828 ;*** Decrement lifetime of our starship's and Zylon photon torpedo
B51A A202     08829 SKIP202         LDX #2               ; Loop over PLAYER2..4
              08830
B51C BD8E0C   08831 LOOP068         LDA PL2SHAPTYPE,X    ; Next PLAYER if not PHOTO
B51F D006     08832                 BNE SKIP203          ;
              08833
B521 B5EB     08834                 LDA PL2LIFE,X        ; Next PLAYER if this PLAY
B523 F002     08835                 BEQ SKIP203          ;
              08836
B525 D6EB     08837                 DEC PL2LIFE,X        ; Decrement photon torpedo
              08838
B527 CA       08839 SKIP203         DEX                  ;
B528 10F2     08840                 BPL LOOP068          ; Next PLAYER
              08841
              08842 ;*** Decrement lifetime of explosion ***************************
B52A A573     08843                 LDA EXPLLIFE         ; Skip if explosion lifeti
B52C F016     08844                 BEQ SKIP206          ;
              08845
B52E C673     08846                 DEC EXPLLIFE         ; Decrement explosion life
```

```
B530 D004     08847                 BNE SKIP204             ; Skip if explosion lifeti
              08848
B532 A211     08849                 LDX #NUMSPCOBJ.NORM     ; Explosion finished,...
B534 8679     08850                 STX MAXSPCOBJIND        ; ...restore normal number
              08851
B536 C970     08852 SKIP204         CMP #112                ; Skip if explosion lifeti
B538 B004     08853                 BCS SKIP205             ;
              08854
B53A A200     08855                 LDX #0                  ; HITBADNESS := NO HIT
B53C 868A     08856                 STX HITBADNESS          ;
              08857
B53E C918     08858 SKIP205         CMP #24                 ; Skip if explosion lifeti
B540 B002     08859                 BCS SKIP206             ;
              08860
B542 C679     08861                 DEC MAXSPCOBJIND        ; Decrement number of expl
              08862
              08863 ;*** Increment stardate clock ***********************************
B544 C674     08864 SKIP206         DEC CLOCKTIM            ; Decrement stardate clock
B546 1021     08865                 BPL SKIP209             ; Return if timer is still
              08866
B548 A928     08867                 LDA #40                 ; Reset stardate clock tim
B54A 8574     08868                 STA CLOCKTIM            ;
              08869
B54C A204     08870                 LDX #4                  ; Increment stardate clock
B54E FEA309   08871 LOOP069         INC GCSTARDAT,X         ;
B551 BDA309   08872                 LDA GCSTARDAT,X         ;
B554 C9DA     08873                 CMP #[CCS.COL3!ROM.9]+1 ;
B556 900D     08874                 BCC SKIP208             ;
B558 A9D0     08875                 LDA #[CCS.COL3!ROM.0]   ;
B55A 9DA309   08876                 STA GCSTARDAT,X         ;
B55D E003     08877                 CPX #3                  ;
B55F D001     08878                 BNE SKIP207             ;
B561 CA       08879                 DEX                     ;
B562 CA       08880 SKIP207         DEX                     ;
B563 10E9     08881                 BPL LOOP069             ;
              08882
              08883 ;*** Decrement Zylon unit movement timer ************************
B565 C678     08884 SKIP208         DEC ZYLONUNITTIM        ; Decrement Zylon unit mov
B567 3001     08885                 BMI SKIP210             ; If timer < 0 move Zylon
              08886
B569 60       08887 SKIP209         RTS                     ; Return
              08888
              08889 ;*** Restore Zylon unit movement timer and decrement score *******
B56A A931     08890 SKIP210         LDA #49                 ; Reset Zylon unit movemen
B56C 8578     08891                 STA ZYLONUNITTIM        ;
              08892
B56E A5CB     08893                 LDA SCORE               ; SCORE := SCORE - 1
B570 D002     08894                 BNE SKIP211             ;
B572 C6CC     08895                 DEC SCORE+1             ;
B574 C6CB     08896 SKIP211         DEC SCORE               ;
              08897
B576 A664     08898                 LDX ISDEMOMODE          ; Return if in demo mode
B578 D0EF     08899                 BNE SKIP209             ;
              08900
              08901 ;*** Is starbase surrounded? ************************************
B57A 866A     08902                 STX L.ISDESTROYED       ; Init L.ISDESTROYED with
B57C BDC908   08903 LOOP070         LDA GCMEMMAP,X          ; Loop over all sectors, l
B57F 1019     08904                 BPL SKIP212             ; Skip if not a starbase s
              08905
```

```
B581 20F1B7   08906                JSR ISSURROUNDED        ; Skip if starbase sector
B584 F014     08907                BEQ SKIP212             ;
              08908
              08909 ;*** Starbase is surrounded, destroy starbase ********************
B586 A902     08910                LDA #2                  ; Replace starbase sector
B588 9DC908   08911                STA GCMEMMAP,X          ;
B58B 856A     08912                STA L.ISDESTROYED       ; Flag destruction of star
              08913
B58D 38       08914                SEC                     ; SCORE := SCORE - 18
B58E A5CB     08915                LDA SCORE               ;
B590 E912     08916                SBC #18                 ;
B592 85CB     08917                STA SCORE               ;
B594 A5CC     08918                LDA SCORE+1             ;
B596 E900     08919                SBC #0                  ;
B598 85CC     08920                STA SCORE+1             ;
              08921
B59A E8       08922 SKIP212        INX                     ;
B59B 10DF     08923                BPL LOOP070             ; Next sector
              08924
              08925 ;*** Report starbase destruction *********************************
B59D A56A     08926                LDA L.ISDESTROYED       ; Skip if no starbase has
B59F F00F     08927                BEQ SKIP213             ;
              08928
B5A1 2C9709   08929                BIT GCSTATRAD           ; Skip notification if Sub
B5A4 700A     08930                BVS SKIP213             ;
              08931
B5A6 A015     08932                LDY #$15                ; Set title phrase "STARBA
B5A8 2023B2   08933                JSR SETTITLE            ;
              08934
B5AB A218     08935                LDX #$18                ; Play beeper sound patter
B5AD 20A6B3   08936                JSR BEEP                ;
              08937
              08938 ;*** Pick new starbase to be hunted by Zylon units ***************
B5B0 C69F     08939 SKIP213        DEC HUNTTIM             ; Decrement hunting timer
B5B2 3007     08940                BMI SKIP214             ; If timer < 0 decide whic
              08941
B5B4 A693     08942                LDX HUNTSECTOR          ; Skip if Zylon units alre
B5B6 BDC908   08943                LDA GCMEMMAP,X          ;
B5B9 301F     08944                BMI SKIP215             ;
              08945
B5BB A907     08946 SKIP214        LDA #7                  ; Reset hunting timer
B5BD 859F     08947                STA HUNTTIM             ;
              08948
B5BF A07F     08949                LDY #127                ; Loop over 127(+1) random
B5C1 AD0AD2   08950 LOOP071        LDA RANDOM              ;
B5C4 297F     08951                AND #$7F                ;
B5C6 AA       08952                TAX                     ;
B5C7 BDC908   08953                LDA GCMEMMAP,X          ; Skip if starbase sector
B5CA 300E     08954                BMI SKIP215             ;
B5CC 88       08955                DEY                     ;
B5CD 10F2     08956                BPL LOOP071             ; Next sector
              08957
B5CF A27F     08958                LDX #127                ; Loop over all sectors of
B5D1 BDC908   08959 LOOP072        LDA GCMEMMAP,X          ;
B5D4 3004     08960                BMI SKIP215             ; Skip if starbase sector
B5D6 CA       08961                DEX                     ;
B5D7 10F8     08962                BPL LOOP072             ; Next sector
              08963
B5D9 60       08964                RTS                     ; Return (no starbase sect
```

```
                08965
                08966 ;*** Store coordinates of starbase to be hunted ******************
B5DA 8693       08967 SKIP215         STX HUNTSECTOR          ; Store hunted starbase se
B5DC 8A         08968                 TXA                     ;
B5DD 290F       08969                 AND #$0F                ;
B5DF 8594       08970                 STA HUNTSECTCOLUMN      ;
B5E1 8A         08971                 TXA                     ;
B5E2 4A         08972                 LSR A                   ;
B5E3 4A         08973                 LSR A                   ;
B5E4 4A         08974                 LSR A                   ;
B5E5 4A         08975                 LSR A                   ;
B5E6 8595       08976                 STA HUNTSECTROW         ;
                08977
                08978 ;*** Move all Zylon units toward hunted starbase *****************
B5E8 A2FF       08979                 LDX #$FF                ;
B5EA E8         08980 LOOP073         INX                     ; Loop over all sectors to
B5EB 1030       08981                 BPL SKIP218             ; Jump into loop body belo
                08982
                08983 ;*** Strip marker bits from moved Zylon units *******************
B5ED A200       08984                 LDX #0                  ;
B5EF BDC908     08985 LOOP074         LDA GCMEMMAP,X          ; Loop over all sectors
B5F2 29DF       08986                 AND #$DF                ;
B5F4 9DC908     08987                 STA GCMEMMAP,X          ; Strip marker bit B5 from
B5F7 E8         08988                 INX                     ;
B5F8 10F5       08989                 BPL LOOP074             ; Next sector
                08990
                08991 ;*** Handle surrounded starbase *********************************
B5FA 2C9709     08992                 BIT GCSTATRAD           ; Return if Subspace Radio
B5FD 701D       08993                 BVS SKIP217             ;
                08994
B5FF A200       08995                 LDX #0                  ; Loop over all sectors
B601 BDC908     08996 LOOP075         LDA GCMEMMAP,X          ;
B604 1013       08997                 BPL SKIP216             ; Skip if not a starbase s
B606 20F1B7     08998                 JSR ISSURROUNDED        ; Skip if starbase not sur
B609 F00E       08999                 BEQ SKIP216             ;
                09000
B60B A963       09001                 LDA #99                 ; Yes, starbase surrounded
B60D 8578       09002                 STA ZYLONUNITTIM        ; ...set Zylon unit moveme
                09003
B60F A013       09004                 LDY #$13                ; Set title phrase "STARBA
B611 2023B2     09005                 JSR SETTITLE            ;
                09006
B614 A218       09007                 LDX #$18                ; Play beeper sound patter
B616 4CA6B3     09008                 JMP BEEP                ; ...and return
                09009
B619 E8         09010 SKIP216         INX                     ;
B61A 10E5       09011                 BPL LOOP075             ; Next sector
                09012
B61C 60         09013 SKIP217         RTS                     ; Return
                09014
                09015 ;*** Move single Zylon unit *************************************
B61D BCC908     09016 SKIP218         LDY GCMEMMAP,X          ; X contains current secto
B620 C00A       09017                 CPY #$0A                ; Next sector if it has ma
B622 B0C6       09018                 BCS LOOP073             ;
                09019
B624 AD0AD2     09020                 LDA RANDOM              ; Get random number
B627 D9BBBF     09021                 CMP MOVEPROBTAB,Y       ; Get movement probability
B62A B0BE       09022                 BCS LOOP073             ; Next sector if movement
                09023
```

```
B62C E490     09024                 CPX CURRSECTOR          ; Next sector if this is o
B62E F0BA     09025                 BEQ LOOP073             ;
              09026
              09027 ;*** Compute distance to starbase by moving Zylon unit into 9 dire
B630 A008     09028                 LDY #8                  ; Loop over 8(+1) possible
B632 18       09029 LOOP076         CLC                     ;
B633 8A       09030                 TXA                     ;
B634 79C0BF   09031                 ADC COMPASSOFFTAB,Y     ; Add direction offset to
B637 856A     09032                 STA L.NEWSECTOR         ; Store new sector
              09033
B639 290F     09034                 AND #$0F                ; Calc distance ("block di
B63B 38       09035                 SEC                     ; ...starbase sector and t
B63C E594     09036                 SBC HUNTSECTCOLUMN      ;
B63E B004     09037                 BCS SKIP219             ;
B640 49FF     09038                 EOR #$FF                ;
B642 6901     09039                 ADC #1                  ;
B644 856B     09040 SKIP219         STA L.ABSDIFFCOLUMN     ;
B646 A56A     09041                 LDA L.NEWSECTOR         ;
B648 4A       09042                 LSR A                   ;
B649 4A       09043                 LSR A                   ;
B64A 4A       09044                 LSR A                   ;
B64B 4A       09045                 LSR A                   ;
B64C 38       09046                 SEC                     ;
B64D E595     09047                 SBC HUNTSECTROW         ;
B64F B004     09048                 BCS SKIP220             ;
B651 49FF     09049                 EOR #$FF                ;
B653 6901     09050                 ADC #1                  ;
B655 18       09051 SKIP220         CLC                     ;
B656 656B     09052                 ADC L.ABSDIFFCOLUMN     ;
              09053
B658 999600   09054                 STA NEWZYLONDIST,Y      ; Store distance in distan
B65B 88       09055                 DEY                     ;
B65C 10D4     09056                 BPL LOOP076             ; Next direction
              09057
              09058 ;*** Pick the shortest distance to starbase **********************
B65E A901     09059                 LDA #1                  ; Loop over compass rose d
B660 856B     09060                 STA L.LOOPCNT2          ; ...provoke movement rega
              09061
B662 A007     09062 LOOP077         LDY #7                  ;
B664 B99600   09063 LOOP078         LDA NEWZYLONDIST,Y      ; Loop over all 7(+1) comp
B667 C59E     09064                 CMP OLDZYLONDIST        ;
B669 B024     09065                 BCS SKIP222             ; Next direction if new di
              09066
B66B 18       09067                 CLC                     ; Calc new Galactic Chart
B66C 8A       09068                 TXA                     ;
B66D 79C0BF   09069                 ADC COMPASSOFFTAB,Y     ;
B670 301D     09070                 BMI SKIP222             ; Next direction if new se
              09071
B672 846A     09072                 STY L.DIRECTIONIND      ; Save compass rose direct
B674 A8       09073                 TAY                     ;
B675 B9C908   09074                 LDA GCMEMMAP,Y          ;
B678 D013     09075                 BNE SKIP221             ; Next direction if new se
              09076
B67A BDC908   09077                 LDA GCMEMMAP,X          ; Preload Zylon sector typ
B67D C490     09078                 CPY CURRSECTOR          ;
B67F F00C     09079                 BEQ SKIP221             ; Next direction if sector
              09080
B681 0920     09081                 ORA #$20                ; New sector for Zylon uni
B683 99C908   09082                 STA GCMEMMAP,Y          ; Temporarily mark that se
```

```
B686 A900     09083                 LDA #0                  ;
B688 9DC908   09084                 STA GCMEMMAP,X          ; Clear old Zylon unit sec
B68B F00B     09085                 BEQ SKIP223             ; Next sector (uncondition
              09086
B68D A46A     09087 SKIP221         LDY L.DIRECTIONIND      ; Restore compass rose dir
B68F 88       09088 SKIP222         DEY                     ; Next compass rose direct
B690 10D2     09089                 BPL LOOP078             ;
              09090
B692 E69E     09091                 INC OLDZYLONDIST        ; Increment center distanc
B694 C66B     09092                 DEC L.LOOPCNT2          ;
B696 10CA     09093                 BPL LOOP077             ; Loop over all compass ro
              09094
B698 4CEAB5   09095 SKIP223         JMP LOOP073             ; Next sector
              09096
              09097 ; ****************************************************************
              09098 ; *
              09099 ; *                                   ROTATE
              09100 ; *
              09101 ; *        Rotate position vector component (coordinate) by fixed
              09102 ; *
              09103 ; ****************************************************************
              09104
              09105 ; DESCRIPTION
              09106 ;
              09107 ; This subroutine rotates a position vector component (coordinate)
              09108 ; object by a fixed angle around the center of the 3D coordinate s
              09109 ; location of our starship. This is used in the Front, Aft, and Lo
              09110 ; views to rotate space objects in and out of the view. Although t
              09111 ; deceptively short, there is some interesting math involved, so a
              09112 ; discussion is in order.
              09113 ;
              09114 ; ROTATION MATHEMATICS
              09115 ;
              09116 ; The game uses a left-handed 3D coordinate system with the positi
              09117 ; pointing to the right, the positive y-axis pointing up, and the
              09118 ; z-axis pointing into flight direction.
              09119 ;
              09120 ; A rotation in this coordinate system around the y-axis (horizont
              09121 ; can be expressed as
              09122 ;
              09123 ;     x' :=   cos(ry) * x + sin(ry) * z    (1a)
              09124 ;     z' := - sin(ry) * x + cos(ry) * z    (1b)
              09125 ;
              09126 ; where ry is the clockwise rotation angle around the y-axis, x an
              09127 ; coordinates before this rotation, and the primed coordinates x'
              09128 ; coordinates after this rotation. The y-coordinate is not changed
              09129 ; rotation.
              09130 ;
              09131 ; A rotation in this coordinate system around the x-axis (vertical
              09132 ; be expressed as
              09133 ;
              09134 ;     z' :=   cos(rx) * z + sin(rx) * y    (2a)
              09135 ;     y' := - sin(rx) * z + cos(rx) * y    (2b)
              09136 ;
              09137 ; where rx is the clockwise rotation angle around the x-axis, z an
              09138 ; coordinates before this rotation, and the primed coordinates z'
              09139 ; coordinates after this rotation. The x-coordinate is not changed
              09140 ; rotation.
              09141 ;
```

```
09142 ; SUBROUTINE IMPLEMENTATION OVERVIEW
09143 ;
09144 ; A single call of this subroutine is able to compute one of the f
09145 ; expressions (1a)-(2b). To compute all four expressions to get th
09146 ; coordinates, this subroutine has to be called four times. This i
09147 ; in pairs in GAMELOOP ($A1F3) at $A391 and $A398, and at $A3AE an
09148 ; respectively.
09149 ;
09150 ; The first pair of calls calculates the new x and z coordinates o
09151 ; object due to a horizontal (left/right) rotation of our starship
09152 ; y-axis following expressions (1a) and (1b).
09153 ;
09154 ; The second pair of calls calculates the new y and z coordinates
09155 ; space object due to a vertical (up/down) rotation of our starshi
09156 ; x-axis following expressions (2a) and (2b).
09157 ;
09158 ; If you look at the code, you may be wondering how this calculati
09159 ; executed, as there is neither a sin() nor a cos() function call.
09160 ; actually find implemented, however, are the following calculatio
09161 ;
09162 ;     Joystick left                         Joystick right
09163 ;     ---------------------                 ---------------------
09164 ;     x :=  x      + z / 64    (3a)         x :=  x      - z / 64
09165 ;     z := -x / 64 + z         (3b)         z :=  x / 64 + z
09166 ;
09167 ;     Joystick down                         Joystick up
09168 ;     ---------------------                 ---------------------
09169 ;     y :=  y      + z / 64    (5a)         y :=  y      - z / 64
09170 ;     z := -y / 64 + z         (5b)         z :=  y / 64 + z
09171 ;
09172 ; CORDIC ALGORITHM
09173 ;
09174 ; When you compare expressions (1a)-(2b) with (3a)-(6b), notice th
09175 ; between the expressions if you substitute
09176 ;
09177 ;     sin(ry) -> 1 / 64,
09178 ;     cos(ry) -> 1,
09179 ;     sin(rx) -> 1 / 64, and
09180 ;     cos(rx) -> 1.
09181 ;
09182 ; From sin(ry) = 1 / 64 and sin(rx) = 1 / 64 you can derive that t
09183 ; angles ry and rx by which the space object is rotated per game l
09184 ; have a constant value of 0.89 degrees, as arcsine(1 / 64) = 0.89
09185 ;
09186 ; What about cos(ry) and cos(rx)? The substitution does not match
09187 ; angle exactly, because cos(0.89 degrees) = 0.99988 and is not ex
09188 ; However, this value is so close to 1 that substituting cos(0.89
09189 ; 1 is a very good approximation, simplifying calculations signifi
09190 ;
09191 ; Another significant simplification results from the division by
09192 ; the actual division operation can be replaced with a much faster
09193 ; operation.
09194 ;
09195 ; This calculation-friendly way of computing rotations is known as
09196 ; (COordinate Rotation DIgital Computer)" algorithm.
09197 ;
09198 ; MINSKY ROTATION
09199 ;
09200 ; There is one more interesting mathematical subtlety: Did you not
```

```
09201 ; expressions (1a)-(2b) use a new (primed) pair of variables to st
09202 ; resulting coordinates, whereas in the implemented expressions (3
09203 ; value of the first coordinate of a coordinate pair is overwritte
09204 ; value and this value is used in the subsequent calculation of th
09205 ; coordinate? For example, when the joystick is pushed left, the f
09206 ; this subroutine calculates the new value of x according to expre
09207 ; overwriting the old value of x. During the second call to calcul
09208 ; according to expression (3b), the new value of x is used instead
09209 ; one. Is this to save the memory needed to temporarily store the
09210 ; x? Is this a bug? If so, why does the rotation calculation actua
09211 ;
09212 ; Have a look at the expression pair (3a) and (3b) (the other expr
09213 ; (4a)-(6b) work in a similar fashion):
09214 ;
09215 ;     x :=  x      + z / 64
09216 ;     z := -x / 64 + z
09217 ;
09218 ; With the substitution 1 / 64 -> e, we get
09219 ;
09220 ;     x :=  x     + e * z
09221 ;     z := -e * x + z
09222 ;
09223 ; Note that x is calculated first and then used in the second expr
09224 ; using primed coordinates for the resulting coordinates after cal
09225 ; two expressions we get
09226 ;
09227 ;     x' := x + e * z
09228 ;     z' := -e * x' + z = -e * (x + e * z) + z = -e * x + (1 - e^2
09229 ;
09230 ; or in matrix form
09231 ;
09232 ;     |x'| := | 1       e   | * |x|
09233 ;     |z'|    |-e  (1 - e^2)|   |z|
09234 ;
09235 ; Surprisingly, this turns out to be a rotation matrix, because it
09236 ; is (1 * (1 - e^2) - (e * -e)) = 1.
09237 ;
09238 ; (Incidentally, the column vectors of this matrix do not form an
09239 ; basis, as their scalar product is 1 * e + (-e * (1 - e^2)) = -e^
09240 ; Orthogonality holds for e = 0 only.)
09241 ;
09242 ; This kind of rotation calculation is described by Marvin Minsky
09243 ; HAKMEM", Item 149, p. 73, MIT AI Lab, February 1972] and is call
09244 ; Rotation".
09245 ;
09246 ; SUBROUTINE IMPLEMENTATION DETAILS
09247 ;
09248 ; To better understand how the implementation of this subroutine w
09249 ; again a look at expressions (3a)-(6b). If you rearrange the expr
09250 ; little their structure is always of the form
09251 ;
09252 ;     TERM1 := TERM1 SIGN TERM2 / 64
09253 ;
09254 ;     or shorter
09255 ;
09256 ;     TERM1 := TERM1 SIGN TERM3
09257 ;
09258 ;     where
09259 ;
```

```
                09260 ;     TERM3 := TERM2 / 64
                09261 ;     SIGN := + or -
                09262 ;
                09263 ; and where TERM1 and TERM2 are position vector components (coordi
                09264 ; fact, this is all this subroutine actually does: It simply adds
                09265 ; by 64 to TERM1 or subtracts TERM2 divided by 64 from TERM1.
                09266 ;
                09267 ; When calling this subroutine the correct indices for the appropr
                09268 ; vector components (coordinates) TERM1 and TERM2 are passed in th
                09269 ; registers, respectively.
                09270 ;
                09271 ; What about SIGN between TERM1 and TERM3? Have again a look at ex
                09272 ; (3a)-(6b). To compute the two new coordinates after a rotation,
                09273 ; toggles from plus to minus and vice versa. The SIGN is initializ
                09274 ; JOYSTICKDELTA ($6D) before calling subroutine ROTATE ($B69B) and
                09275 ; inside every call of this subroutine before the addition or subt
                09276 ; terms takes place there. The initial value of SIGN should be pos
                09277 ; the rotation is clockwise (the joystick is pushed right or up) a
                09278 ; (-) if the rotation is counter-clockwise (the joystick is pushed
                09279 ; down), respectively. Because SIGN is always toggled inside the s
                09280 ; before the addition or subtraction of the terms actually happens
                09281 ; have to pass the already toggled value with the first call.
                09282 ;
                09283 ; NOTE: Unclear still are three instructions starting at address $
                09284 ; seem to set the two least significant bits of TERM3 in a random
                09285 ; this be some quick hack to avoid messing with exact but potentia
                09286 ; two-complement's arithmetic here?
                09287 ;
                09288 ; INPUT
                09289 ;
                09290 ;   X = Position vector component index of TERM2. Used values are:
                09291 ;     $00..$30 -> z-component (z-coordinate) of position vector 0.
                09292 ;     $31..$61 -> x-component (x-coordinate) of position vector 0.
                09293 ;     $62..$92 -> y-component (y-coordinate) of position vector 0.
                09294 ;
                09295 ;   Y = Position vector component index of TERM1. Used values are:
                09296 ;     $00..$30 -> z-component (z-coordinate) of position vector 0.
                09297 ;     $31..$61 -> x-component (x-coordinate) of position vector 0.
                09298 ;     $62..$92 -> y-component (y-coordinate) of position vector 0.
                09299 ;
                09300 ;   JOYSTICKDELTA ($6D) = Initial value of SIGN. Used values are:
                09301 ;     $01 -> (= Positive) Rotate right or up
                09302 ;     $FF -> (= Negative) Rotate left or down
                09303
                09304                                       ; TERM3 is a 24-bit value,
                09305                                       ; $(sign)(high byte)(low b
    =006A       09306 L.TERM3LO       = $6A                 ; TERM3 (high byte), where
    =006B       09307 L.TERM3HI       = $6B                 ; TERM3 (low byte),  where
    =006C       09308 L.TERM3SIGN     = $6C                 ; TERM3 (sign),      where
                09309
B69B BDAD09     09310 ROTATE          LDA ZPOSSIGN,X        ;
B69E 4901       09311                 EOR #$01              ;
B6A0 F002       09312                 BEQ SKIP224           ; Skip if sign of TERM2 is
B6A2 A9FF       09313                 LDA #$FF              ;
                09314
B6A4 856B       09315 SKIP224         STA L.TERM3HI         ; If TERM2 pos. -> TERM3 :
B6A6 856C       09316                 STA L.TERM3SIGN       ; If TERM2 neg. -> TERM3 :
B6A8 BD400A     09317                 LDA ZPOSHI,X          ; where xx := TERM2 (high
B6AB 856A       09318                 STA L.TERM3LO         ;
```

```
                09319
B6AD AD0AD2     09320                   LDA RANDOM              ; (?) Hack to avoid messin
B6B0 09BF       09321                   ORA #$BF                ; (?) arithmetic? Provides
B6B2 5DD30A     09322                   EOR ZPOSLO,X            ; (?) bits B1..0 in TERM3.
                09323
B6B5 0A         09324                   ASL A                   ; TERM3 := TERM3 * 4 (= TE
B6B6 266A       09325                   ROL L.TERM3LO           ;
B6B8 266B       09326                   ROL L.TERM3HI           ;
B6BA 0A         09327                   ASL A                   ;
B6BB 266A       09328                   ROL L.TERM3LO           ;
B6BD 266B       09329                   ROL L.TERM3HI           ;
                09330
B6BF A56D       09331                   LDA JOYSTICKDELTA       ; Toggle SIGN for next cal
B6C1 49FF       09332                   EOR #$FF                ;
B6C3 856D       09333                   STA JOYSTICKDELTA       ;
B6C5 301A       09334                   BMI SKIP225             ; If SIGN negative then su
                09335
                09336 ;*** Addition ***************************************************
B6C7 18         09337                   CLC                     ; TERM1 := TERM1 + TERM3
B6C8 B9D30A     09338                   LDA ZPOSLO,Y            ; (24-bit addition)
B6CB 656A       09339                   ADC L.TERM3LO           ;
B6CD 99D30A     09340                   STA ZPOSLO,Y            ;
                09341
B6D0 B9400A     09342                   LDA ZPOSHI,Y            ;
B6D3 656B       09343                   ADC L.TERM3HI           ;
B6D5 99400A     09344                   STA ZPOSHI,Y            ;
                09345
B6D8 B9AD09     09346                   LDA ZPOSSIGN,Y          ;
B6DB 656C       09347                   ADC L.TERM3SIGN         ;
B6DD 99AD09     09348                   STA ZPOSSIGN,Y          ;
B6E0 60         09349                   RTS                     ;
                09350
                09351 ;*** Subtraction ************************************************
B6E1 38         09352 SKIP225           SEC                     ; TERM1 := TERM1 - TERM3
B6E2 B9D30A     09353                   LDA ZPOSLO,Y            ; (24-bit subtraction)
B6E5 E56A       09354                   SBC L.TERM3LO           ;
B6E7 99D30A     09355                   STA ZPOSLO,Y            ;
                09356
B6EA B9400A     09357                   LDA ZPOSHI,Y            ;
B6ED E56B       09358                   SBC L.TERM3HI           ;
B6EF 99400A     09359                   STA ZPOSHI,Y            ;
                09360
B6F2 B9AD09     09361                   LDA ZPOSSIGN,Y          ;
B6F5 E56C       09362                   SBC L.TERM3SIGN         ;
B6F7 99AD09     09363                   STA ZPOSSIGN,Y          ;
B6FA 60         09364                   RTS                     ;
                09365
                09366 ;*****************************************************************
                09367 ;*
                09368 ;*                                 SCREENCOLUMN
                09369 ;*
                09370 ;*       Calculate pixel column number from centered pixel column
                09371 ;*
                09372 ;*****************************************************************
                09373
                09374 ; DESCRIPTION
                09375 ;
                09376 ; Converts a pixel column number relative to the horizontal screen
                09377 ; pixel column number relative to the top-left corner of the scree
```

```
                09378 ; the result in table PIXELCOLUMN ($0C2A). The passed relative pix
                09379 ; number is always positive. The sign is picked from the correspon
                09380 ; x-component of the position vector (x-coordinate).
                09381 ;
                09382 ; If the passed relative pixel column number is offscreen horizont
                09383 ; calculation is skipped and code execution returns. If the positi
                09384 ; corresponding to this pixel represents a PLAYFIELD space object
                09385 ; explosion fragments) a new position vector is initialized before
                09386 ; execution returns. If it represents a PLAYER space object the PL
                09387 ; offscreen before code execution returns.
                09388 ;
                09389 ; NOTE: The horizontal screen center's pixel column number for PLA
                09390 ; objects has a value of 80 = 160 PLAYFIELD pixels / 2. For PLAYER
                09391 ; it has a value of 125 Player/Missile (PM) pixels (from left to r
                09392 ; pixels to the horizontal screen center - 3 PM pixels relative of
                09393 ; PLAYER shape's horizontal center to its left edge = 125 PM pixel
                09394 ;
                09395 ; INPUT
                09396 ;
                09397 ;   A = Pixel column number relative to the horizontal screen cent
                09398 ;       positive. Used values are:
                09399 ;     0..80 -> Regular values, pixel is onscreen
                09400 ;     $FF   -> Pixel is offscreen
                09401 ;
                09402 ;   X = Position vector index. Used values are:
                09403 ;     0..4  -> Position vector of a PLAYER space object
                09404 ;     5..48 -> Position vector of a PLAYFIELD space object
                09405
    =006D       09406 L.PIXELCOLUMN   = $6D                   ; Saves relative pixel col
                09407
B6FB C950       09408 SCREENCOLUMN    CMP #80                 ; If pixel is offscreen (A
B6FD B05B       09409                 BCS SKIP233             ; ...return via initializi
                09410
B6FF 856D       09411                 STA L.PIXELCOLUMN       ; Save relative pixel colu
B701 A950       09412                 LDA #80                 ; If PLAYFIELD space objec
B703 E005       09413                 CPX #NUMSPCOBJ.PL       ; If PLAYER space object
B705 B002       09414                 BCS SKIP226             ;
B707 A97D       09415                 LDA #125                ;
                09416
B709 BCDE09     09417 SKIP226         LDY XPOSSIGN,X          ; Skip if x-coordinate pos
B70C D009       09418                 BNE SKIP227             ;
                09419
B70E 38         09420                 SEC                     ; Pixel in left screen hal
B70F E66D       09421                 INC L.PIXELCOLUMN       ;
B711 E56D       09422                 SBC L.PIXELCOLUMN       ;
B713 9D2A0C     09423                 STA PIXELCOLUMN,X       ; Pixel column := CENTERCO
B716 60         09424                 RTS                     ; Return
                09425
B717 18         09426 SKIP227         CLC                     ; Pixel in right screen ha
B718 656D       09427                 ADC L.PIXELCOLUMN       ;
B71A 9D2A0C     09428                 STA PIXELCOLUMN,X       ; Pixel column := CENTERCO
B71D 60         09429                 RTS                     ; Return
                09430
                09431 ;*****************************************************************
                09432 ;*
                09433 ;*                                  SCREENROW
                09434 ;*
                09435 ;*          Calculate pixel row number from centered pixel row num
                09436 ;*
```

```
                09437 ;*****************************************************************
                09438
                09439 ; Converts a pixel row number relative to the vertical screen cent
                09440 ; row number relative to the top-left corner of the screen and sto
                09441 ; in table PIXELROWNEW ($0BF9). The passed relative pixel row numb
                09442 ; positive. The sign is picked from the corresponding y-component
                09443 ; position vector (y-coordinate).
                09444 ;
                09445 ; If the passed relative pixel row number is offscreen vertically
                09446 ; calculation is skipped and code execution returns. If the positi
                09447 ; corresponding to this pixel represents a PLAYFIELD space object
                09448 ; explosion fragments) a new position vector is initialized in sub
                09449 ; INITPOSVEC ($B764) before code execution returns. If it represen
                09450 ; space object the PLAYER is pushed offscreen before code executio
                09451 ;
                09452 ; NOTE: The vertical screen center's pixel row number for PLAYFIEL
                09453 ; objects has a value of 50 = 100 PLAYFIELD pixels / 2. For PLAYER
                09454 ; it has a value of 122 Player/Missile (PM) pixels (from top to bo
                09455 ; pixels to start of Display List + 16 PM pixels to begin of PLAYF
                09456 ; pixels to vertical screen center - 2 PM pixels (?) = 122 PM pixe
                09457 ;
                09458 ; NOTE: If the position vector corresponding to the pixel represen
                09459 ; space object the passed pixel row number is doubled because 1 PL
                09460 ; has the same height as 2 PM pixels at single-line resolution.
                09461 ;
                09462 ; When in Long-Range Scan view the z-coordinate takes the place of
                09463 ; y-coordinate of the Front or Aft view. If the Long-Range Scan is
                09464 ; passed pixel row number is treated randomly as a negative or pos
                09465 ; (mirror effect).
                09466 ;
                09467 ; INPUT
                09468 ;
                09469 ;   A = Pixel row number relative to the vertical screen center, a
                09470 ;       positive. Used values are:
                09471 ;     0..50 -> Regular values, pixel is onscreen
                09472 ;     $FF   -> Pixel is offscreen
                09473 ;
                09474 ;   X = Position vector index. Used values are:
                09475 ;     0..4  -> Position vector of a PLAYER space object
                09476 ;     5..48 -> Position vector of a PLAYFIELD space object
                09477
    =006D       09478 L.PIXELROW      = $6D                   ; Saves relative pixel row
                09479
B71E C932       09480 SCREENROW       CMP #50                 ; If pixel is offscreen (A
B720 B038       09481                 BCS SKIP233             ; ...return via initializi
                09482
B722 856D       09483                 STA L.PIXELROW          ; Save relative pixel row
B724 A932       09484                 LDA #50                 ; If PLAYFIELD space objec
B726 E005       09485                 CPX #NUMSPCOBJ.PL       ;
B728 B004       09486                 BCS SKIP228             ;
B72A 066D       09487                 ASL L.PIXELROW          ; If PLAYER space object -
B72C A97A       09488                 LDA #122                ; If PLAYER space object -
                09489
B72E 24D0       09490 SKIP228         BIT SHIPVIEW            ; Skip if not in Long-Rang
B730 5013       09491                 BVC SKIP230             ;
                09492
B732 2C9609     09493                 BIT GCSTATLRS           ; Skip if Long-Range Scan
B735 1007       09494                 BPL SKIP229             ;
                09495
```

```
B737 2C0AD2   09496                BIT RANDOM             ; Long-Range Scan damaged.
B73A 500E     09497                BVC SKIP231            ; ...branch randomly to pi
B73C 7015     09498                BVS SKIP232            ; ...(mirror effect)
              09499
B73E BCAD09   09500 SKIP229        LDY ZPOSSIGN,X         ;
B741 D007     09501                BNE SKIP231            ; Skip if z-coordinate pos
B743 F00E     09502                BEQ SKIP232            ; Skip if z-coordinate neg
              09503
B745 BC0F0A   09504 SKIP230        LDY YPOSSIGN,X         ;
B748 F009     09505                BEQ SKIP232            ; Skip if y-coordinate neg
              09506
B74A 38       09507 SKIP231        SEC                    ; Pixel in upper screen ha
B74B E66D     09508                INC L.PIXELROW         ;
B74D E56D     09509                SBC L.PIXELROW         ;
B74F 9DF90B   09510                STA PIXELROWNEW,X      ; Pixel row  := CENTERROW
B752 60       09511                RTS                    ; Return
              09512
B753 18       09513 SKIP232        CLC                    ; Pixel in lower screen ha
B754 656D     09514                ADC L.PIXELROW         ;
B756 9DF90B   09515                STA PIXELROWNEW,X      ; Pixel row := CENTERROW +
B759 60       09516                RTS                    ; Return
              09517
B75A E005     09518 SKIP233        CPX #NUMSPCOBJ.PL      ; Space object is offscree
B75C B006     09519                BCS INITPOSVEC         ; ...PLAYFIELD space objec
B75E A9FB     09520                LDA #251               ; ...PLAYER space object
B760 9DF90B   09521                STA PIXELROWNEW,X      ;
B763 60       09522 SKIP234        RTS                    ; Return
              09523
              09524 ;*****************************************************************
              09525 ;*
              09526 ;*                                INITPOSVEC
              09527 ;*
              09528 ;*               Initialize position vector of a space object
              09529 ;*
              09530 ;*****************************************************************
              09531
              09532 ; DESCRIPTION
              09533 ;
              09534 ; Initializes the position vector of a space object.
              09535 ;
              09536 ; This subroutine executes the following steps:
              09537 ;
              09538 ; (1)  Set the pixel row and column number to an offscreen value (
              09539 ;
              09540 ; (2)  If the position vector represents an explosion fragment spa
              09541 ;      return code execution immediately. This avoids generating n
              09542 ;      fragment space objects. They are separately initialized in
              09543 ;      COPYPOSVEC ($ACAF), which is called from subroutine INITEXP
              09544 ;
              09545 ; (3)  Assign default values (see below) to the position vector co
              09546 ;      (coordinates) depending on our starship's view.
              09547 ;
              09548 ; Code execution continues into subroutine RNDINVXY ($B7BE) where
              09549 ; coordinates are inverted randomly.
              09550 ;
              09551 ; After passing through this and the next subroutine RNDINVXY ($B7
              09552 ; components of a position vector (coordinates) are assigned to on
              09553 ; following values depending on our starship's view:
              09554 ;
```

```
                09555 ; o   FRONT VIEW
                09556 ;
                09557 ;     +------------+---------------------------------------+
                09558 ;     | Coordinate |                Values                 |
                09559 ;     +------------+---------------------------------------+
                09560 ;     |     x      |  -4095..+4095 (-($0***)..+$0***) <KM> |
                09561 ;     |     y      |  -4095..+4095 (-($0***)..+$0***) <KM> |
                09562 ;     |     z      |  +3840..+4095 (          +$0F**) <KM> |
                09563 ;     +------------+---------------------------------------+
                09564 ;
                09565 ; o   AFT VIEW
                09566 ;
                09567 ;     +------------+---------------------------------------+
                09568 ;     | Coordinate |                Values                 |
                09569 ;     +------------+---------------------------------------+
                09570 ;     |     x      |  -3840..+3840 (-($0*00)..+$0*00) <KM> |
                09571 ;     |     y      |  -3840..+3840 (-($0*00)..+$0*00) <KM> |
                09572 ;     |     z      |  -3968.. -128 (-($0*80)        ) <KM> |
                09573 ;     +------------+---------------------------------------+
                09574 ;     Values of x, y, and z coordinates change in increments of 25
                09575 ;     Second digit of z-coordinate is -MAX(RNDY,RNDX), where
                09576 ;     RNDY := RND($00..$0F), RNDX := RND($00..$0F).
                09577 ;
                09578 ; o   LONG-RANGE SCAN VIEW
                09579 ;
                09580 ;     +------------+---------------------------------------+
                09581 ;     | Coordinate |                Values                 |
                09582 ;     +------------+---------------------------------------+
                09583 ;     |     x      | -65535..+65535 (-($****)..$****) <KM> |
                09584 ;     |     y      |  -4095..+4095  (-($0***)..$0***) <KM> |
                09585 ;     |     z      | -65535..+65535 (-($****)..$****) <KM> |
                09586 ;     +------------+---------------------------------------+
                09587 ;
                09588 ; INPUT
                09589 ;
                09590 ;   X = Position vector index. Used values are: 0..48.
                09591
    =006A       09592 L.MAXRNDXY      = $6A                 ; Saves MAX(new y-coordina
                09593                                       ;  ...new x-coordinate (hi
                09594
B764 A963       09595 INITPOSVEC      LDA #99               ; Init to offscreen pixel
B766 9DF90B     09596                 STA PIXELROWNEW,X     ;
B769 9D2A0C     09597                 STA PIXELCOLUMN,X     ;
                09598
B76C E011       09599                 CPX #NUMSPCOBJ.NORM   ; Return if pos vector is
B76E B0F3       09600                 BCS SKIP234           ; This avoids creating new
                09601
B770 AD0AD2     09602                 LDA RANDOM            ; RNDY := RND($00..$0F)
B773 290F       09603                 AND #$0F              ;
B775 856A       09604                 STA L.MAXRNDXY        ; Save RNDY
B777 9DA20A     09605                 STA YPOSHI,X          ; y-coordinate (high byte)
                09606
B77A AD0AD2     09607                 LDA RANDOM            ; RNDX := RND($00..$0F)
B77D 290F       09608                 AND #$0F              ;
B77F C56A       09609                 CMP L.MAXRNDXY        ;
B781 9002       09610                 BCC SKIP235           ;
B783 856A       09611                 STA L.MAXRNDXY        ; Save MAX(RNDY,RNDX)
B785 9D710A     09612 SKIP235         STA XPOSHI,X          ; x-coordinate (high byte)
                09613
```

```
B788 A90F     09614                LDA #$0F               ; z-coordinate (high byte)
B78A 9D400A   09615                STA ZPOSHI,X           ;
              09616
B78D A5D0     09617                LDA SHIPVIEW           ; z-coordinate (sign) := 1
B78F 4901     09618                EOR #$01               ;
B791 2901     09619                AND #$01               ;
B793 9DAD09   09620                STA ZPOSSIGN,X         ;
B796 D011     09621                BNE SKIP236            ; Skip if in Front or Long
              09622
              09623                                       ; Aft view only:
B798 9D040B   09624                STA XPOSLO,X           ; x-coordinate (low byte)
B79B 9D350B   09625                STA YPOSLO,X           ; y-coordinate (low byte)
B79E 38       09626                SEC                    ; z-coordinate (high byte)
B79F E56A     09627                SBC L.MAXRNDXY         ;
B7A1 9D400A   09628                STA ZPOSHI,X           ;
B7A4 A980     09629                LDA #$80               ; z-coordinate (low byte)
B7A6 9DD30A   09630                STA ZPOSLO,X           ;
              09631
B7A9 24D0     09632 SKIP236        BIT SHIPVIEW           ; If not in Long-Range Sca
B7AB 5011     09633                BVC RNDINVXY           ;
              09634
              09635                                       ; Long-Range Scan view onl
B7AD AD0AD2   09636                LDA RANDOM             ; x-coordinate (high byte)
B7B0 9D710A   09637                STA XPOSHI,X           ;
B7B3 AD0AD2   09638                LDA RANDOM             ; z-coordinate (high byte)
B7B6 9D400A   09639                STA ZPOSHI,X           ;
B7B9 2901     09640                AND #$01               ; Invert z-coordinate rand
B7BB 9DAD09   09641                STA ZPOSSIGN,X         ;
              09642
              09643 ;*****************************************************************
              09644 ;*
              09645 ;*                                  RNDINVXY
              09646 ;*
              09647 ;*         Randomly invert the x and y components of a position ve
              09648 ;*
              09649 ;*****************************************************************
              09650
              09651 ; DESCRIPTION
              09652 ;
              09653 ; Randomly inverts the x and y components of a position vector (x
              09654 ; coordinates). See also subroutine INITPOSVEC ($B764).
              09655 ;
              09656 ; INPUT
              09657 ;
              09658 ;   X = Position vector index. Used values are: 0..48.
              09659
B7BE AD0AD2   09660 RNDINVXY       LDA RANDOM             ; Set sign of y-coordinate
B7C1 2901     09661                AND #$01               ;
B7C3 9D0F0A   09662                STA YPOSSIGN,X         ;
B7C6 D00F     09663                BNE SKIP237            ; Skip if sign positive
              09664
B7C8 38       09665                SEC                    ; Sign negative -> Calc ne
B7C9 FD350B   09666                SBC YPOSLO,X           ; (calculate two's-complem
B7CC 9D350B   09667                STA YPOSLO,X           ;
B7CF A900     09668                LDA #0                 ;
B7D1 FDA20A   09669                SBC YPOSHI,X           ;
B7D4 9DA20A   09670                STA YPOSHI,X           ;
              09671
B7D7 AD0AD2   09672 SKIP237        LDA RANDOM             ; Set sign of x-coordinate
```

```
B7DA 2901     09673                AND #$01                ;
B7DC 9DDE09   09674                STA XPOSSIGN,X          ;
B7DF D00F     09675                BNE SKIP238             ; Skip if sign positive
              09676
B7E1 38       09677                SEC                     ; Sign negative -> Calc ne
B7E2 FD040B   09678                SBC XPOSLO,X            ; (calculate two's-complem
B7E5 9D040B   09679                STA XPOSLO,X            ;
B7E8 A900     09680                LDA #0                  ;
B7EA FD710A   09681                SBC XPOSHI,X            ;
B7ED 9D710A   09682                STA XPOSHI,X            ;
B7F0 60       09683 SKIP238        RTS                     ; Return
              09684
              09685 ;*****************************************************************
              09686 ;*
              09687 ;*                                ISSURROUNDED
              09688 ;*
              09689 ;*              Check if a sector is surrounded by Zylon units
              09690 ;*
              09691 ;*****************************************************************
              09692
              09693 ; DESCRIPTION
              09694 ;
              09695 ; Checks if a sector of the Galactic Chart is surrounded by Zylon
              09696 ; adjacent NORTH, EAST, SOUTH, and WEST sectors.
              09697 ;
              09698 ; INPUT
              09699 ;
              09700 ;   X = Sector of Galactic Chart. Used values are: $00..$7F with,
              09701 ;     $00 -> NORTHWEST corner sector
              09702 ;     $0F -> NORTHEAST corner sector
              09703 ;     $70 -> SOUTHWEST corner sector
              09704 ;     $7F -> SOUTHWEST corner sector
              09705 ;
              09706 ; OUTPUT
              09707 ;
              09708 ;   A = Returns if the sector is surrounded by Zylon units in the
              09709 ;       NORTH, EAST, SOUTH, and WEST sectors.
              09710 ;       0 -> Sector is not surrounded
              09711 ;     > 0 -> Sector is surrounded
              09712
B7F1 BDC808   09713 ISSURROUNDED   LDA GCMEMMAP-1,X        ; Check WEST sector
B7F4 F00D     09714                BEQ SKIP239             ;
B7F6 BDCA08   09715                LDA GCMEMMAP+1,X        ; Check EAST sector
B7F9 F008     09716                BEQ SKIP239             ;
B7FB BDB908   09717                LDA GCMEMMAP-16,X       ; Check NORTH sector
B7FE F003     09718                BEQ SKIP239             ;
B800 BDD908   09719                LDA GCMEMMAP+16,X       ; Check SOUTH sector
B803 60       09720 SKIP239        RTS                     ; Return
              09721
              09722 ;*****************************************************************
              09723 ;*
              09724 ;*                                  UPDPANEL
              09725 ;*
              09726 ;*                          Update Control Panel Display
              09727 ;*
              09728 ;*****************************************************************
              09729
              09730 ; DESCRIPTION
              09731 ;
```

```
                09732 ; This subroutine executes the following steps:
                09733 ;
                09734 ; (1)  Accelerate or decelerate our starship, update the VELOCITY
                09735 ;
                09736 ;      If the new velocity value is different from the current one
                09737 ;      increment or decrement the current velocity value toward th
                09738 ;
                09739 ;      If the Engines are damaged or destroyed (and hyperwarp is n
                09740 ;      then store a random value (less or equal than the current v
                09741 ;      the current velocity.
                09742 ;
                09743 ;      Display the updated velocity by the VELOCITY readout of the
                09744 ;      Display.
                09745 ;
                09746 ; (2)  Update THETA, PHI, and RANGE readouts.
                09747 ;
                09748 ;      If the Attack Computer is working then display the x, y, an
                09749 ;      coordinates of the currently tracked space object as THETA,
                09750 ;      RANGE readout values of the Control Panel Display.
                09751 ;
                09752 ; (3)  Calculate overall energy consumption.
                09753 ;
                09754 ;      Add the overall energy consumption per game loop iteration
                09755 ;      counter. This value is given in energy subunits (256 energy
                09756 ;      energy unit displayed by the 4-digit ENERGY readout of the
                09757 ;      Display). It is the total of the following items:
                09758 ;
                09759 ;      (1)  8 energy subunits if the Shields are up
                09760 ;
                09761 ;      (2)  2 energy subunits if the Attack Computer is on
                09762 ;
                09763 ;      (3)  1 energy subunit of the life support system
                09764 ;
                09765 ;      (4)  Our starship's Engines energy drain rate (depending on
                09766 ;
                09767 ;      If there is a carryover of the energy counter then decremen
                09768 ;      readout of the Control Panel Display by one energy unit aft
                09769 ;      execution has continued into subroutine DECENERGY ($B86F).
                09770
                09771 ;*** Accelerate or decelerate our starship **********************
B804 A670       09772 UPDPANEL        LDX VELOCITYLO          ; Skip if new velocity = c
B806 E471       09773                 CPX NEWVELOCITY         ;
B808 F008       09774                 BEQ SKIP241             ;
                09775
B80A 9004       09776                 BCC SKIP240             ; In/decrement current vel
B80C C670       09777                 DEC VELOCITYLO          ;
B80E B012       09778                 BCS SKIP242             ;
B810 E670       09779 SKIP240         INC VELOCITYLO          ;
                09780
B812 A5C0       09781 SKIP241         LDA WARPSTATE           ; Skip if hyperwarp engage
B814 D00C       09782                 BNE SKIP242             ;
                09783
B816 2C9309     09784                 BIT GCSTATENG           ; Skip if Engines are OK
B819 1007       09785                 BPL SKIP242             ;
                09786
B81B A571       09787                 LDA NEWVELOCITY         ; Store RND(0..current vel
B81D 2D0AD2     09788                 AND RANDOM              ;
B820 8570       09789                 STA VELOCITYLO          ;
                09790
```

```
B822 A001     09791 SKIP242         LDY #VELOCD1-PANELTXT-1 ; Update digits of VELOCIT
B824 20CDB8   09792                 JSR SHOWDIGITS          ;
              09793
              09794 ;*** Display coordinates of tracked space object of Control Panel
B827 2C9509   09795                 BIT GCSTATCOM           ; Skip if Attack Computer
B82A 3030     09796                 BMI SKIP243             ;
              09797
B82C A931     09798                 LDA #$31                ; Update THETA readout (x-
B82E A017     09799                 LDY #THETAC1-PANELTXT   ;
B830 20A7B8   09800                 JSR SHOWCOORD           ;
              09801
B833 A962     09802                 LDA #$62                ; Update PHI readout (y-co
B835 A01D     09803                 LDY #PHIC1-PANELTXT     ;
B837 20A7B8   09804                 JSR SHOWCOORD           ;
              09805
B83A A900     09806                 LDA #$00                ; Update RANGE readout (z-
B83C A023     09807                 LDY #RANGEC1-PANELTXT   ;
B83E 20A7B8   09808                 JSR SHOWCOORD           ;
              09809
B841 AD6E09   09810                 LDA RANGEC1+2           ; Hack to clear RANGE digi
B844 8D6F09   09811                 STA RANGEC1+3           ; Copy RANGE digit 2 to di
B847 C90A     09812                 CMP #CCS.9+1            ; Skip if digit character
B849 B011     09813                 BCS SKIP243             ;
              09814
B84B AE5C09   09815                 LDX TRACKDIGIT          ; Get z-coordinate (low by
B84E BDD30A   09816                 LDA ZPOSLO,X            ;
B851 4A       09817                 LSR A                   ; ...divide it by 16...
B852 4A       09818                 LSR A                   ;
B853 4A       09819                 LSR A                   ;
B854 4A       09820                 LSR A                   ;
B855 AA       09821                 TAX                     ;
B856 BDE90E   09822                 LDA MAPTOBCD99,X        ; ...map value of $00..$0F
B859 8D6F09   09823                 STA RANGEC1+3           ; ...and store it to RANGE
              09824
              09825 ;*** Calculate overall energy consumption ************************
B85C 18       09826 SKIP243         CLC                     ;
B85D A57F     09827                 LDA ENERGYCNT           ; Load energy counter
B85F 657D     09828                 ADC DRAINSHIELDS        ; Add energy drain rate of
B861 6580     09829                 ADC DRAINENGINES        ; Add energy drain rate of
B863 657E     09830                 ADC DRAINATTCOMP        ; Add energy drain rate of
B865 6901     09831                 ADC #$01                ; Add 1 energy subunit of
B867 C57F     09832                 CMP ENERGYCNT           ;
B869 857F     09833                 STA ENERGYCNT           ;
B86B B039     09834                 BCS SKIP246             ; Return if no energy coun
              09835
B86D A203     09836                 LDX #3                  ; Will decrement third ene
              09837
              09838 ;*****************************************************************
              09839 ;*
              09840 ;*                                  DECENERGY
              09841 ;*
              09842 ;*                               Decrease energy
              09843 ;*
              09844 ;*****************************************************************
              09845
              09846 ; DESCRIPTION
              09847 ;
              09848 ; When not in demo mode, subtract energy from the 4-digit ENERGY r
              09849 ; Control Panel Display. If crossing a 100-energy-unit boundary du
```

```
                    09850 ; subtraction the score is decremented by one unit. If the energy
                    09851 ; game is over.
                    09852 ;
                    09853 ; INPUT
                    09854 ;
                    09855 ;   X = ENERGY readout digit to be decremented. Used values are:
                    09856 ;     1 -> Subtract 100 units from ENERGY readout
                    09857 ;     2 -> Subtract  10 units from ENERGY readout
                    09858 ;     3 -> Subtract   1 unit  from ENERGY readout
                    09859
                    09860 ;*** Display ENERGY readout ************************************
B86F 2464           09861 DECENERGY       BIT ISDEMOMODE          ; Return if in demo mode
B871 7033           09862                 BVS SKIP246             ;
                    09863
B873 DE5509         09864                 DEC ENERGYD1,X          ; Decrement energy digit c
B876 BD5509         09865                 LDA ENERGYD1,X          ;
B879 C980           09866                 CMP #CCS.COL2!CCS.0     ;
B87B B029           09867                 BCS SKIP246             ; Return if digit characte
B87D A989           09868                 LDA #CCS.COL2!CCS.9     ;
B87F 9D5509         09869                 STA ENERGYD1,X          ; Store digit character '9
                    09870
                    09871 ;*** Decrement score when crossing a 100-energy-unit boundary whil
B882 E002           09872                 CPX #2                  ; Skip if no crossing of 1
B884 D008           09873                 BNE SKIP245             ;
                    09874
B886 A5CB           09875                 LDA SCORE               ; SCORE := SCORE - 1
B888 D002           09876                 BNE SKIP244             ;
B88A C6CC           09877                 DEC SCORE+1             ;
B88C C6CB           09878 SKIP244         DEC SCORE               ;
                    09879
B88E CA             09880 SKIP245         DEX                     ;
B88F 10DE           09881                 BPL DECENERGY           ; Next digit
                    09882
                    09883 ;*** Energy is zero, game over *********************************
B891 A20A           09884                 LDX #CCS.SPC            ; Clear 4-digit ENERGY rea
B893 8A             09885                 TXA                     ;
B894 A003           09886                 LDY #3                  ;
B896 995509         09887 LOOP079         STA ENERGYD1,Y          ;
B899 88             09888                 DEY                     ;
B89A 10FA           09889                 BPL LOOP079             ;
                    09890
B89C 2045B0         09891                 JSR SETVIEW             ; Set Front view
                    09892
B89F A031           09893                 LDY #$31                ; Set title phrase "MISSIO
B8A1 A204           09894                 LDX #$04                ; Set mission bonus offset
B8A3 200AB1         09895                 JSR GAMEOVER            ; Game over
                    09896
B8A6 60             09897 SKIP246         RTS                     ; Return
                    09898
                    09899 ;*****************************************************************
                    09900 ;*
                    09901 ;*                                    SHOWCOORD
                    09902 ;*
                    09903 ;*  Display a position vector component (coordinate) in Control Pa
                    09904 ;*
                    09905 ;*****************************************************************
                    09906
                    09907 ; DESCRIPTION
                    09908 ;
```

```
                09909 ; Displays a position vector component (coordinate) by one of the
                09910 ; RANGE readouts of the Control Panel Display.
                09911 ;
                09912 ; Write the sign to the Control Panel Display, then map the high b
                09913 ; respective coordinate (x -> THETA, y -> PHI, z -> RANGE) to a BC
                09914 ; 00..99. Code execution continues into subroutine SHOWDIGITS ($B8
                09915 ; digits are actually stored in the Control Panel Display.
                09916 ;
                09917 ; NOTE: If the digits of either the THETA or PHI readout are to be
                09918 ; the x or y position vector component (high byte) is $FF then twe
                09919 ; to $FE. This avoids accessing table MAPTOBCD99 ($0EE9) with an i
                09920 ; that would return the special value $EA. This value represents t
                09921 ; ($0E) and CCS.SPC ($0A) characters (see comments in subroutine I
                09922 ; ($B3BA)) that are displayed by the RANGE readout only.
                09923 ;
                09924 ; INPUT
                09925 ;
                09926 ;   A = Position vector component (coordinate) offset. Used values
                09927 ;     $00 -> z-coordinate
                09928 ;     $31 -> x-coordinate
                09929 ;     $62 -> y-coordinate
                09930 ;
                09931 ;   Y = Offset into the Control Panel Display memory map. Used val
                09932 ;     $17 -> First character (sign) of THETA readout (x-coordinate
                09933 ;            space object)
                09934 ;     $1D -> First character (sign) of PHI readout   (y-coordinate
                09935 ;            space object)
                09936 ;     $23 -> First character (sign) of RANGE readout (z-coordinate
                09937 ;            space object)
                09938
    =006A       09939 L.SIGNCHAR      = $6A                   ; Saves sign character
                09940
B8A7 18         09941 SHOWCOORD       CLC                     ; Add index of tracked spa
B8A8 6D5C09     09942                 ADC TRACKDIGIT          ; ...to position vector co
B8AB AA         09943                 TAX                     ; Save position vector com
                09944
                09945 ;*** Display sign in Control Panel Display **********************
B8AC A910       09946                 LDA #CCS.PLUS           ; Save '+' (CCS.PLUS) to s
B8AE 856A       09947                 STA L.SIGNCHAR          ;
                09948
B8B0 BDAD09     09949                 LDA ZPOSSIGN,X          ; Prep sign of coordinate
B8B3 4A         09950                 LSR A                   ;
B8B4 BD400A     09951                 LDA ZPOSHI,X            ; Prep coordinate (high by
B8B7 B004       09952                 BCS SKIP247             ; Skip if sign is positive
                09953
B8B9 49FF       09954                 EOR #$FF                ; Invert coordinate (high
B8BB C66A       09955                 DEC L.SIGNCHAR          ; Change saved sign charac
                09956
B8BD AA         09957 SKIP247         TAX                     ; Save coordinate (high by
B8BE A56A       09958                 LDA L.SIGNCHAR          ; Store sign character in
B8C0 994909     09959                 STA PANELTXT,Y          ;
                09960
                09961 ;*** Get RANGE digits ******************************************
B8C3 98         09962                 TYA                     ; Skip if RANGE is to be d
B8C4 2910       09963                 AND #$10                ;
B8C6 F005       09964                 BEQ SHOWDIGITS          ;
                09965
B8C8 E0FF       09966                 CPX #$FF                ; If coordinate (high byte
B8CA D001       09967                 BNE SHOWDIGITS          ; This avoids output of CC
```

```
B8CC CA       09968                DEX                      ; ...THETA and PHI readout
              09969
              09970 ;*****************************************************************
              09971 ;*
              09972 ;*                                SHOWDIGITS
              09973 ;*
              09974 ;*          Display a value by a readout of the Control Panel Disp
              09975 ;*
              09976 ;*****************************************************************
              09977
              09978 ; DESCRIPTION
              09979 ;
              09980 ; Converts a binary value in $00..$FF to a BCD-value in 0..99 and
              09981 ; a 2-digit number in the Control Panel Display.
              09982 ;
              09983 ; INPUT
              09984 ;
              09985 ;   X = Value to be displayed as a 2-digit BCD-value. Used values
              09986 ;
              09987 ;   Y = Offset into the Control Panel Display memory map relative
              09988 ;       character of the Control Panel Display (the 'V' of the VEL
              09989 ;       readout). Used values are:
              09990 ;     $01 -> Character before first digit of VELOCITY readout
              09991 ;     $17 -> First character (sign) of THETA readout (x-coordinate
              09992 ;            space object)
              09993 ;     $1D -> First character (sign) of PHI readout   (y-coordinate
              09994 ;            space object)
              09995 ;     $23 -> First character (sign) of RANGE readout (z-coordinate
              09996 ;            space object)
              09997
B8CD BDE90E   09998 SHOWDIGITS     LDA MAPTOBCD99,X        ; Map binary value to BCD-
B8D0 AA       09999                TAX                     ;
B8D1 290F     10000                AND #$0F                ;
B8D3 994B09   10001                STA PANELTXT+2,Y        ; Store 'ones' digit in Co
B8D6 8A       10002                TXA                     ;
B8D7 4A       10003                LSR A                   ;
B8D8 4A       10004                LSR A                   ;
B8D9 4A       10005                LSR A                   ;
B8DA 4A       10006                LSR A                   ;
B8DB 994A09   10007                STA PANELTXT+1,Y        ; Store 'tens' digit in Co
B8DE 60       10008                RTS                     ; Return
              10009
              10010 ;*****************************************************************
              10011 ;*
              10012 ;*               G A M E   D A T A   ( P A R T   2   O F   2 )
              10013 ;*
              10014 ;*****************************************************************
              10015
              10016 ;*** Color register offsets of PLAYER0..4 ************************
B8DF 00       10017 PLCOLOROFFTAB   .BYTE 0                                ; PLAYER0
B8E0 01       10018                 .BYTE 1                                ; PLAYER1
B8E1 02       10019                 .BYTE 2                                ; PLAYER2
B8E2 03       10020                 .BYTE 3                                ; PLAYER3
B8E3 07       10021                 .BYTE 7                                ; PLAYER4
              10022
              10023 ;*** Shape table 1 (PLAYER2..4) *********************************
B8E4 00       10024 PLSHAP1TAB      .BYTE $00                              ; ........
B8E5 18       10025                 .BYTE $18                              ; ...##...
B8E6 3C       10026                 .BYTE $3C                              ; ..####..
```

```
B8E7 7E      10027                .BYTE $7E                      ; .######.
B8E8 7E      10028                .BYTE $7E                      ; .######.
B8E9 76      10029                .BYTE $76                      ; .###.##.
B8EA F7      10030                .BYTE $F7                      ; ####.###
B8EB DF      10031                .BYTE $DF                      ; ##.#####
B8EC DF      10032                .BYTE $DF                      ; ##.#####
B8ED FF      10033                .BYTE $FF                      ; ########
B8EE FF      10034                .BYTE $FF                      ; ########
B8EF F7      10035                .BYTE $F7                      ; ####.###
B8F0 76      10036                .BYTE $76                      ; .###.##.
B8F1 7E      10037                .BYTE $7E                      ; .######.
B8F2 7E      10038                .BYTE $7E                      ; .######.
B8F3 3C      10039                .BYTE $3C                      ; ..####..
B8F4 18      10040                .BYTE $18                      ; ...##...
B8F5 10      10041                .BYTE $10                      ; ...#....
B8F6 38      10042                .BYTE $38                      ; ..###...
B8F7 7C      10043                .BYTE $7C                      ; .#####..
B8F8 7C      10044                .BYTE $7C                      ; .#####..
B8F9 FE      10045                .BYTE $FE                      ; #######.
B8FA DE      10046                .BYTE $DE                      ; ##.####.
B8FB DA      10047                .BYTE $DA                      ; ##.##.#.
B8FC FA      10048                .BYTE $FA                      ; #####.#.
B8FD EE      10049                .BYTE $EE                      ; ###.###.
B8FE EE      10050                .BYTE $EE                      ; ###.###.
B8FF 7C      10051                .BYTE $7C                      ; .#####..
B900 7C      10052                .BYTE $7C                      ; .#####..
B901 38      10053                .BYTE $38                      ; ..###...
B902 10      10054                .BYTE $10                      ; ...#....
B903 18      10055                .BYTE $18                      ; ...##...
B904 3C      10056                .BYTE $3C                      ; ..####..
B905 3C      10057                .BYTE $3C                      ; ..####..
B906 7E      10058                .BYTE $7E                      ; .######.
B907 6E      10059                .BYTE $6E                      ; .##.###.
B908 7A      10060                .BYTE $7A                      ; .####.#.
B909 7E      10061                .BYTE $7E                      ; .######.
B90A 76      10062                .BYTE $76                      ; .###.##.
B90B 7E      10063                .BYTE $7E                      ; .######.
B90C 3C      10064                .BYTE $3C                      ; ..####..
B90D 3C      10065                .BYTE $3C                      ; ..####..
B90E 18      10066                .BYTE $18                      ; ...##...
B90F 10      10067                .BYTE $10                      ; ...#....
B910 38      10068                .BYTE $38                      ; ..###...
B911 38      10069                .BYTE $38                      ; ..###...
B912 7C      10070                .BYTE $7C                      ; .#####..
B913 74      10071                .BYTE $74                      ; .###.#..
B914 7C      10072                .BYTE $7C                      ; .#####..
B915 6C      10073                .BYTE $6C                      ; .##.##..
B916 38      10074                .BYTE $38                      ; ..###...
B917 38      10075                .BYTE $38                      ; ..###...
B918 10      10076                .BYTE $10                      ; ...#....
B919 10      10077                .BYTE $10                      ; ...#....
B91A 18      10078                .BYTE $18                      ; ...##...
B91B 3C      10079                .BYTE $3C                      ; ..####..
B91C 2C      10080                .BYTE $2C                      ; ..#.##..
B91D 3C      10081                .BYTE $3C                      ; ..####..
B91E 3C      10082                .BYTE $3C                      ; ..####..
B91F 18      10083                .BYTE $18                      ; ...##...
B920 08      10084                .BYTE $08                      ; ....#...
B921 10      10085                .BYTE $10                      ; ...#....
```

```
B922 38       10086                .BYTE $38                      ; ..###...
B923 38       10087                .BYTE $38                      ; ..###...
B924 28       10088                .BYTE $28                      ; ..#.#...
B925 38       10089                .BYTE $38                      ; ..###...
B926 10       10090                .BYTE $10                      ; ...#....
B927 3C       10091                .BYTE $3C                      ; ..####..
B928 3C       10092                .BYTE $3C                      ; ..####..
B929 24       10093                .BYTE $24                      ; ..#..#..
B92A 3C       10094                .BYTE $3C                      ; ..####..
B92B 7E       10095                .BYTE $7E                      ; .######.
B92C 7E       10096                .BYTE $7E                      ; .######.
B92D 7E       10097                .BYTE $7E                      ; .######.
B92E 5A       10098                .BYTE $5A                      ; .#.##.#.
B92F FF       10099                .BYTE $FF                      ; ########
B930 FF       10100                .BYTE $FF                      ; ########
B931 42       10101                .BYTE $42                      ; .#....#.
B932 42       10102                .BYTE $42                      ; .#....#.
B933 42       10103                .BYTE $42                      ; .#....#.
B934 42       10104                .BYTE $42                      ; .#....#.
B935 42       10105                .BYTE $42                      ; .#....#.
B936 42       10106                .BYTE $42                      ; .#....#.
B937 1C       10107                .BYTE $1C                      ; ...###..
B938 1C       10108                .BYTE $1C                      ; ...###..
B939 14       10109                .BYTE $14                      ; ...#.#..
B93A 3E       10110                .BYTE $3E                      ; ..#####.
B93B 3E       10111                .BYTE $3E                      ; ..#####.
B93C 3E       10112                .BYTE $3E                      ; ..#####.
B93D 2A       10113                .BYTE $2A                      ; ..#.#.#.
B93E 7F       10114                .BYTE $7F                      ; .#######
B93F 7F       10115                .BYTE $7F                      ; .#######
B940 22       10116                .BYTE $22                      ; ..#...#.
B941 22       10117                .BYTE $22                      ; ..#...#.
B942 22       10118                .BYTE $22                      ; ..#...#.
B943 22       10119                .BYTE $22                      ; ..#...#.
B944 22       10120                .BYTE $22                      ; ..#...#.
B945 18       10121                .BYTE $18                      ; ...##...
B946 18       10122                .BYTE $18                      ; ...##...
B947 3C       10123                .BYTE $3C                      ; ..####..
B948 3C       10124                .BYTE $3C                      ; ..####..
B949 3C       10125                .BYTE $3C                      ; ..####..
B94A 3C       10126                .BYTE $3C                      ; ..####..
B94B 7E       10127                .BYTE $7E                      ; .######.
B94C 24       10128                .BYTE $24                      ; ..#..#..
B94D 24       10129                .BYTE $24                      ; ..#..#..
B94E 24       10130                .BYTE $24                      ; ..#..#..
B94F 24       10131                .BYTE $24                      ; ..#..#..
B950 10       10132                .BYTE $10                      ; ...#....
B951 10       10133                .BYTE $10                      ; ...#....
B952 38       10134                .BYTE $38                      ; ..###...
B953 38       10135                .BYTE $38                      ; ..###...
B954 38       10136                .BYTE $38                      ; ..###...
B955 7C       10137                .BYTE $7C                      ; .#####..
B956 28       10138                .BYTE $28                      ; ..#.#...
B957 28       10139                .BYTE $28                      ; ..#.#...
B958 28       10140                .BYTE $28                      ; ..#.#...
B959 18       10141                .BYTE $18                      ; ...##...
B95A 18       10142                .BYTE $18                      ; ...##...
B95B 3C       10143                .BYTE $3C                      ; ..####..
B95C 18       10144                .BYTE $18                      ; ...##...
```

```
B95D 18      10145                .BYTE $18                      ; ...##...
B95E 10      10146                .BYTE $10                      ; ...#....
B95F 10      10147                .BYTE $10                      ; ...#....
B960 38      10148                .BYTE $38                      ; ..###...
B961 10      10149                .BYTE $10                      ; ...#....
B962 18      10150                .BYTE $18                      ; ...##...
B963 7E      10151                .BYTE $7E                      ; .######.
B964 FF      10152                .BYTE $FF                      ; ########
B965 FF      10153                .BYTE $FF                      ; ########
B966 FF      10154                .BYTE $FF                      ; ########
B967 FF      10155                .BYTE $FF                      ; ########
B968 FF      10156                .BYTE $FF                      ; ########
B969 E7      10157                .BYTE $E7                      ; ###..###
B96A E7      10158                .BYTE $E7                      ; ###..###
B96B FF      10159                .BYTE $FF                      ; ########
B96C FF      10160                .BYTE $FF                      ; ########
B96D FF      10161                .BYTE $FF                      ; ########
B96E FF      10162                .BYTE $FF                      ; ########
B96F FF      10163                .BYTE $FF                      ; ########
B970 7E      10164                .BYTE $7E                      ; .######.
B971 7E      10165                .BYTE $7E                      ; .######.
B972 00      10166                .BYTE $00                      ; ........
B973 18      10167                .BYTE $18                      ; ...##...
B974 3C      10168                .BYTE $3C                      ; ..####..
B975 7E      10169                .BYTE $7E                      ; .######.
B976 FF      10170                .BYTE $FF                      ; ########
B977 FF      10171                .BYTE $FF                      ; ########
B978 FF      10172                .BYTE $FF                      ; ########
B979 E7      10173                .BYTE $E7                      ; ###..###
B97A 66      10174                .BYTE $66                      ; .##..##.
B97B FF      10175                .BYTE $FF                      ; ########
B97C FF      10176                .BYTE $FF                      ; ########
B97D FF      10177                .BYTE $FF                      ; ########
B97E FF      10178                .BYTE $FF                      ; ########
B97F 7E      10179                .BYTE $7E                      ; .######.
B980 7E      10180                .BYTE $7E                      ; .######.
B981 00      10181                .BYTE $00                      ; ........
B982 18      10182                .BYTE $18                      ; ...##...
B983 3C      10183                .BYTE $3C                      ; ..####..
B984 7E      10184                .BYTE $7E                      ; .######.
B985 FF      10185                .BYTE $FF                      ; ########
B986 FF      10186                .BYTE $FF                      ; ########
B987 E7      10187                .BYTE $E7                      ; ###..###
B988 66      10188                .BYTE $66                      ; .##..##.
B989 FF      10189                .BYTE $FF                      ; ########
B98A FF      10190                .BYTE $FF                      ; ########
B98B FF      10191                .BYTE $FF                      ; ########
B98C FF      10192                .BYTE $FF                      ; ########
B98D 3C      10193                .BYTE $3C                      ; ..####..
B98E 18      10194                .BYTE $18                      ; ...##...
B98F 3C      10195                .BYTE $3C                      ; ..####..
B990 FF      10196                .BYTE $FF                      ; ########
B991 FF      10197                .BYTE $FF                      ; ########
B992 E7      10198                .BYTE $E7                      ; ###..###
B993 66      10199                .BYTE $66                      ; .##..##.
B994 FF      10200                .BYTE $FF                      ; ########
B995 FF      10201                .BYTE $FF                      ; ########
B996 7E      10202                .BYTE $7E                      ; .######.
B997 3C      10203                .BYTE $3C                      ; ..####..
```

```
B998 00       10204                .BYTE $00                          ; ........
B999 18       10205                .BYTE $18                          ; ...##...
B99A 3C       10206                .BYTE $3C                          ; ..####..
B99B FF       10207                .BYTE $FF                          ; ########
B99C FF       10208                .BYTE $FF                          ; ########
B99D FF       10209                .BYTE $FF                          ; ########
B99E 3C       10210                .BYTE $3C                          ; ..####..
B99F 18       10211                .BYTE $18                          ; ...##...
B9A0 18       10212                .BYTE $18                          ; ...##...
B9A1 3C       10213                .BYTE $3C                          ; ..####..
B9A2 FF       10214                .BYTE $FF                          ; ########
B9A3 3C       10215                .BYTE $3C                          ; ..####..
B9A4 18       10216                .BYTE $18                          ; ...##...
B9A5 28       10217                .BYTE $28                          ; ..#.#...
B9A6 28       10218                .BYTE $28                          ; ..#.#...
B9A7 28       10219                .BYTE $28                          ; ..#.#...
B9A8 28       10220                .BYTE $28                          ; ..#.#...
B9A9 EE       10221                .BYTE $EE                          ; ###.###.
B9AA 00       10222                .BYTE $00                          ; ........
B9AB 00       10223                .BYTE $00                          ; ........
B9AC EE       10224                .BYTE $EE                          ; ###.###.
B9AD 28       10225                .BYTE $28                          ; ..#.#...
B9AE 28       10226                .BYTE $28                          ; ..#.#...
B9AF 28       10227                .BYTE $28                          ; ..#.#...
B9B0 28       10228                .BYTE $28                          ; ..#.#...
              10229
              10230 ;*** Shape table 2 (PLAYER0..1) ***********************************
B9B1 00       10231 PLSHAP2TAB     .BYTE $00                          ; ........
B9B2 81       10232                .BYTE $81                          ; #......#
B9B3 81       10233                .BYTE $81                          ; #......#
B9B4 81       10234                .BYTE $81                          ; #......#
B9B5 81       10235                .BYTE $81                          ; #......#
B9B6 BD       10236                .BYTE $BD                          ; #.####.#
B9B7 FF       10237                .BYTE $FF                          ; ########
B9B8 FF       10238                .BYTE $FF                          ; ########
B9B9 BD       10239                .BYTE $BD                          ; #.####.#
B9BA 81       10240                .BYTE $81                          ; #......#
B9BB 81       10241                .BYTE $81                          ; #......#
B9BC 81       10242                .BYTE $81                          ; #......#
B9BD 81       10243                .BYTE $81                          ; #......#
B9BE 82       10244                .BYTE $82                          ; #.....#.
B9BF 82       10245                .BYTE $82                          ; #.....#.
B9C0 BA       10246                .BYTE $BA                          ; #.###.#.
B9C1 FE       10247                .BYTE $FE                          ; #######.
B9C2 FE       10248                .BYTE $FE                          ; #######.
B9C3 BA       10249                .BYTE $BA                          ; #.###.#.
B9C4 82       10250                .BYTE $82                          ; #.....#.
B9C5 82       10251                .BYTE $82                          ; #.....#.
B9C6 42       10252                .BYTE $42                          ; .#....#.
B9C7 5A       10253                .BYTE $5A                          ; .#.##.#.
B9C8 7E       10254                .BYTE $7E                          ; .######.
B9C9 7E       10255                .BYTE $7E                          ; .######.
B9CA 5A       10256                .BYTE $5A                          ; .#.##.#.
B9CB 42       10257                .BYTE $42                          ; .#....#.
B9CC 44       10258                .BYTE $44                          ; .#...#..
B9CD 54       10259                .BYTE $54                          ; .#.#.#..
B9CE 7C       10260                .BYTE $7C                          ; .#####..
B9CF 7C       10261                .BYTE $7C                          ; .#####..
B9D0 54       10262                .BYTE $54                          ; .#.#.#..
```

```
B9D1 44       10263                .BYTE $44                      ; .#...#..
B9D2 24       10264                .BYTE $24                      ; ..#..#..
B9D3 3C       10265                .BYTE $3C                      ; ..####..
B9D4 3C       10266                .BYTE $3C                      ; ..####..
B9D5 24       10267                .BYTE $24                      ; ..#..#..
B9D6 28       10268                .BYTE $28                      ; ..#.#...
B9D7 38       10269                .BYTE $38                      ; ..###...
B9D8 38       10270                .BYTE $38                      ; ..###...
B9D9 28       10271                .BYTE $28                      ; ..#.#...
B9DA 18       10272                .BYTE $18                      ; ...##...
B9DB 18       10273                .BYTE $18                      ; ...##...
B9DC 10       10274                .BYTE $10                      ; ...#....
B9DD 10       10275                .BYTE $10                      ; ...#....
B9DE E0       10276                .BYTE $E0                      ; ###.....
B9DF F8       10277                .BYTE $F8                      ; #####...
B9E0 F8       10278                .BYTE $F8                      ; #####...
B9E1 FE       10279                .BYTE $FE                      ; #######.
B9E2 57       10280                .BYTE $57                      ; .#.#.###
B9E3 FE       10281                .BYTE $FE                      ; #######.
B9E4 F8       10282                .BYTE $F8                      ; #####...
B9E5 F8       10283                .BYTE $F8                      ; #####...
B9E6 C0       10284                .BYTE $C0                      ; ##......
B9E7 C0       10285                .BYTE $C0                      ; ##......
B9E8 F0       10286                .BYTE $F0                      ; ####....
B9E9 C0       10287                .BYTE $C0                      ; ##......
B9EA F0       10288                .BYTE $F0                      ; ####....
B9EB F0       10289                .BYTE $F0                      ; ####....
B9EC FC       10290                .BYTE $FC                      ; ######..
B9ED BE       10291                .BYTE $BE                      ; #.#####.
B9EE FC       10292                .BYTE $FC                      ; ######..
B9EF F0       10293                .BYTE $F0                      ; ####....
B9F0 80       10294                .BYTE $80                      ; #.......
B9F1 80       10295                .BYTE $80                      ; #.......
B9F2 C0       10296                .BYTE $C0                      ; ##......
B9F3 C0       10297                .BYTE $C0                      ; ##......
B9F4 F0       10298                .BYTE $F0                      ; ####....
B9F5 BC       10299                .BYTE $BC                      ; #.####..
B9F6 F0       10300                .BYTE $F0                      ; ####....
B9F7 C0       10301                .BYTE $C0                      ; ##......
B9F8 07       10302                .BYTE $07                      ; .....###
B9F9 1F       10303                .BYTE $1F                      ; ...#####
B9FA 1F       10304                .BYTE $1F                      ; ...#####
B9FB 7F       10305                .BYTE $7F                      ; .#######
B9FC EA       10306                .BYTE $EA                      ; ###.#.#.
B9FD 7F       10307                .BYTE $7F                      ; .#######
B9FE 1F       10308                .BYTE $1F                      ; ...#####
B9FF 1F       10309                .BYTE $1F                      ; ...#####
BA00 03       10310                .BYTE $03                      ; ......##
BA01 03       10311                .BYTE $03                      ; ......##
BA02 0F       10312                .BYTE $0F                      ; ....####
BA03 03       10313                .BYTE $03                      ; ......##
BA04 0F       10314                .BYTE $0F                      ; ....####
BA05 0F       10315                .BYTE $0F                      ; ....####
BA06 3F       10316                .BYTE $3F                      ; ..######
BA07 7D       10317                .BYTE $7D                      ; .#####.#
BA08 3F       10318                .BYTE $3F                      ; ..######
BA09 0F       10319                .BYTE $0F                      ; ....####
BA0A 01       10320                .BYTE $01                      ; .......#
BA0B 01       10321                .BYTE $01                      ; .......#
```

```
BA0C 03      10322                .BYTE $03                       ; ......##
BA0D 03      10323                .BYTE $03                       ; ......##
BA0E 0F      10324                .BYTE $0F                       ; ....####
BA0F 3D      10325                .BYTE $3D                       ; ..####.#
BA10 0F      10326                .BYTE $0F                       ; ....####
BA11 03      10327                .BYTE $03                       ; ......##
BA12 18      10328                .BYTE $18                       ; ...##...
BA13 3C      10329                .BYTE $3C                       ; ..####..
BA14 7E      10330                .BYTE $7E                       ; .######.
BA15 7E      10331                .BYTE $7E                       ; .######.
BA16 DB      10332                .BYTE $DB                       ; ##.##.##
BA17 C3      10333                .BYTE $C3                       ; ##....##
BA18 81      10334                .BYTE $81                       ; #......#
BA19 81      10335                .BYTE $81                       ; #......#
BA1A 81      10336                .BYTE $81                       ; #......#
BA1B 10      10337                .BYTE $10                       ; ...#....
BA1C 38      10338                .BYTE $38                       ; ..###...
BA1D 7C      10339                .BYTE $7C                       ; .#####..
BA1E 7C      10340                .BYTE $7C                       ; .#####..
BA1F D6      10341                .BYTE $D6                       ; ##.#.##.
BA20 C6      10342                .BYTE $C6                       ; ##...##.
BA21 82      10343                .BYTE $82                       ; #.....#.
BA22 82      10344                .BYTE $82                       ; #.....#.
BA23 18      10345                .BYTE $18                       ; ...##...
BA24 3C      10346                .BYTE $3C                       ; ..####..
BA25 3C      10347                .BYTE $3C                       ; ..####..
BA26 66      10348                .BYTE $66                       ; .##..##.
BA27 66      10349                .BYTE $66                       ; .##..##.
BA28 42      10350                .BYTE $42                       ; .#....#.
BA29 42      10351                .BYTE $42                       ; .#....#.
BA2A 10      10352                .BYTE $10                       ; ...#....
BA2B 38      10353                .BYTE $38                       ; ..###...
BA2C 38      10354                .BYTE $38                       ; ..###...
BA2D 6C      10355                .BYTE $6C                       ; .##.##..
BA2E 44      10356                .BYTE $44                       ; .#...#..
BA2F 44      10357                .BYTE $44                       ; .#...#..
BA30 18      10358                .BYTE $18                       ; ...##...
BA31 3C      10359                .BYTE $3C                       ; ..####..
BA32 24      10360                .BYTE $24                       ; ..#..#..
BA33 24      10361                .BYTE $24                       ; ..#..#..
BA34 10      10362                .BYTE $10                       ; ...#....
BA35 38      10363                .BYTE $38                       ; ..###...
BA36 28      10364                .BYTE $28                       ; ..#.#...
BA37 18      10365                .BYTE $18                       ; ...##...
BA38 3C      10366                .BYTE $3C                       ; ..####..
BA39 7E      10367                .BYTE $7E                       ; .######.
BA3A FF      10368                .BYTE $FF                       ; ########
BA3B 18      10369                .BYTE $18                       ; ...##...
BA3C 18      10370                .BYTE $18                       ; ...##...
BA3D FF      10371                .BYTE $FF                       ; ########
BA3E 7E      10372                .BYTE $7E                       ; .######.
BA3F 3C      10373                .BYTE $3C                       ; ..####..
BA40 18      10374                .BYTE $18                       ; ...##...
BA41 10      10375                .BYTE $10                       ; ...#....
BA42 38      10376                .BYTE $38                       ; ..###...
BA43 7C      10377                .BYTE $7C                       ; .#####..
BA44 FE      10378                .BYTE $FE                       ; #######.
BA45 38      10379                .BYTE $38                       ; ..###...
BA46 38      10380                .BYTE $38                       ; ..###...
```

```
BA47 FE       10381                .BYTE $FE                          ; #######.
BA48 7C       10382                .BYTE $7C                          ; .#####..
BA49 38       10383                .BYTE $38                          ; ..###...
BA4A 10       10384                .BYTE $10                          ; ...#....
BA4B 18       10385                .BYTE $18                          ; ...##...
BA4C 3C       10386                .BYTE $3C                          ; ..####..
BA4D 7E       10387                .BYTE $7E                          ; .######.
BA4E 18       10388                .BYTE $18                          ; ...##...
BA4F 7E       10389                .BYTE $7E                          ; .######.
BA50 3C       10390                .BYTE $3C                          ; ..####..
BA51 18       10391                .BYTE $18                          ; ...##...
BA52 10       10392                .BYTE $10                          ; ...#....
BA53 38       10393                .BYTE $38                          ; ..###...
BA54 7C       10394                .BYTE $7C                          ; .#####..
BA55 10       10395                .BYTE $10                          ; ...#....
BA56 7C       10396                .BYTE $7C                          ; .#####..
BA57 38       10397                .BYTE $38                          ; ..###...
BA58 10       10398                .BYTE $10                          ; ...#....
BA59 18       10399                .BYTE $18                          ; ...##...
BA5A 3C       10400                .BYTE $3C                          ; ..####..
BA5B 18       10401                .BYTE $18                          ; ...##...
BA5C 3C       10402                .BYTE $3C                          ; ..####..
BA5D 18       10403                .BYTE $18                          ; ...##...
BA5E 10       10404                .BYTE $10                          ; ...#....
BA5F 38       10405                .BYTE $38                          ; ..###...
BA60 38       10406                .BYTE $38                          ; ..###...
BA61 10       10407                .BYTE $10                          ; ...#....
              10408
              10409 ;*** Display List fragments ************************************
              10410 ;
              10411 ; LOCAL VARIABLES
    =1000     10412 PFMEM.C0R0      = PFMEM+0*40                      ; Start addr
    =10C8     10413 PFMEM.C0R5      = PFMEM+5*40                      ; Start addr
    =12A8     10414 PFMEM.C0R17     = PFMEM+17*40                     ; Start addr
              10415
              10416 ;*** Display List fragment for Control Panel Display (bottom text
BA62 8D       10417 DLSTFRAG        .BYTE $8D                         ; GR7 + DLI
BA63 00       10418                 .BYTE $00                         ; BLK1
BA64 464909   10419                 .BYTE $46,<PANELTXT,>PANELTXT     ; GR1 @ PANE
BA67 20       10420                 .BYTE $20                         ; BLK3
BA68 06       10421                 .BYTE $06                         ; GR1
BA69 00       10422                 .BYTE $00                         ; BLK1
              10423
              10424 ;*** Display List fragment for Galactic Chart view **************
BA6A 012EA1   10425 DLSTFRAGGC      .BYTE $01,<DLSTGC,>DLSTGC         ; JMP @ DLST
              10426
              10427 ;*** Display List fragment for Long-Range Scan view *************
BA6D 00       10428 DLSTFRAGLRS     .BYTE $00                         ; BLK1
BA6E 00       10429                 .BYTE $00                         ; BLK1
BA6F 46F8A0   10430                 .BYTE $46,<LRSHEADER,>LRSHEADER   ; GR1 @ LRSH
BA72 4DC810   10431                 .BYTE $4D,<PFMEM.C0R5,>PFMEM.C0R5 ; GR7 @ PFME
              10432
              10433 ;*** Display List fragment for Aft view *************************
BA75 00       10434 DLSTFRAGAFT     .BYTE $00                         ; BLK1
BA76 00       10435                 .BYTE $00                         ; BLK1
BA77 4609A1   10436                 .BYTE $46,<AFTHEADER,>AFTHEADER   ; GR1 @ AFTH
BA7A 4DC810   10437                 .BYTE $4D,<PFMEM.C0R5,>PFMEM.C0R5 ; GR7 @ PFME
              10438
              10439 ;*** Display List fragment for Front view and Title text line ****
```

```
BA7D 4D0010   10440 DLSTFRAGFRONT   .BYTE $4D,<PFMEM.C0R0,>PFMEM.C0R0     ; GR7 @ PFME
BA80 0D       10441                 .BYTE $0D                             ; GR7
BA81 0D       10442                 .BYTE $0D                             ; GR7
BA82 0D       10443                 .BYTE $0D                             ; GR7
BA83 0D       10444                 .BYTE $0D                             ; GR7
BA84 0D       10445                 .BYTE $0D                             ; GR7
BA85 30       10446                 .BYTE $30                             ; BLK4
BA86 461F0D   10447                 .BYTE $46,<TITLETXT,>TITLETXT         ; GR1 @ TITL
BA89 4DA812   10448                 .BYTE $4D,<PFMEM.C0R17,>PFMEM.C0R17   ; GR7 @ PFME
              10449
              10450 ;*** Display List fragment offsets relative to DLSTFRAG **********
BA8C 1B       10451 DLSTFRAGOFFTAB  .BYTE DLSTFRAGFRONT-DLSTFRAG          ; Front view
BA8D 13       10452                 .BYTE DLSTFRAGAFT-DLSTFRAG            ; Aft view
BA8E 0B       10453                 .BYTE DLSTFRAGLRS-DLSTFRAG            ; Long-Range
BA8F 08       10454                 .BYTE DLSTFRAGGC-DLSTFRAG             ; Galactic C
              10455
              10456 ;*** 1-byte bit patterns for 4 pixels of same color for PLAYFIELD
BA90 FF       10457 FOURCOLORPIXEL  .BYTE $FF                             ; COLOR3
BA91 FF       10458                 .BYTE $FF                             ; COLOR3
BA92 FF       10459                 .BYTE $FF                             ; COLOR3
BA93 FF       10460                 .BYTE $FF                             ; COLOR3
BA94 AA       10461                 .BYTE $AA                             ; COLOR2
BA95 FF       10462                 .BYTE $FF                             ; COLOR3
BA96 AA       10463                 .BYTE $AA                             ; COLOR2
BA97 FF       10464                 .BYTE $FF                             ; COLOR3
BA98 AA       10465                 .BYTE $AA                             ; COLOR2
BA99 AA       10466                 .BYTE $AA                             ; COLOR2
BA9A AA       10467                 .BYTE $AA                             ; COLOR2
BA9B FF       10468                 .BYTE $FF                             ; COLOR3
BA9C AA       10469                 .BYTE $AA                             ; COLOR2
BA9D AA       10470                 .BYTE $AA                             ; COLOR2
BA9E AA       10471                 .BYTE $AA                             ; COLOR2
BA9F AA       10472                 .BYTE $AA                             ; COLOR2
BAA0 AA       10473                 .BYTE $AA                             ; COLOR2
BAA1 AA       10474                 .BYTE $AA                             ; COLOR2
BAA2 AA       10475                 .BYTE $AA                             ; COLOR2
BAA3 55       10476                 .BYTE $55                             ; COLOR1
BAA4 55       10477                 .BYTE $55                             ; COLOR1
BAA5 AA       10478                 .BYTE $AA                             ; COLOR2
BAA6 55       10479                 .BYTE $55                             ; COLOR1
BAA7 AA       10480                 .BYTE $AA                             ; COLOR2
BAA8 55       10481                 .BYTE $55                             ; COLOR1
BAA9 55       10482                 .BYTE $55                             ; COLOR1
BAAA 55       10483                 .BYTE $55                             ; COLOR1
BAAB AA       10484                 .BYTE $AA                             ; COLOR2
BAAC 55       10485                 .BYTE $55                             ; COLOR1
BAAD 55       10486                 .BYTE $55                             ; COLOR1
BAAE 55       10487                 .BYTE $55                             ; COLOR1
BAAF 55       10488                 .BYTE $55                             ; COLOR1
              10489
              10490 ;*** Masks to filter 1 pixel (2 bits) from 4 pixels (1 byte of PLA
BAB0 C0       10491 PIXELMASKTAB    .BYTE $C0                             ; ##......
BAB1 30       10492                 .BYTE $30                             ; ..##....
BAB2 0C       10493                 .BYTE $0C                             ; ....##..
BAB3 03       10494                 .BYTE $03                             ; ......##
              10495
              10496 ;*** Velocity table ***************************************
BAB4 00       10497 VELOCITYTAB     .BYTE 0                               ; Speed 0 =
BAB5 01       10498                 .BYTE 1                               ; Speed 1 =
```

```
BAB6 02       10499                .BYTE 2                                ; Speed 2 =
BAB7 04       10500                .BYTE 4                                ; Speed 3 =
BAB8 08       10501                .BYTE 8                                ; Speed 4 =
BAB9 10       10502                .BYTE 16                               ; Speed 5 =
BABA 20       10503                .BYTE 32                               ; Speed 6 =
BABB 40       10504                .BYTE 64                               ; Speed 7 =
BABC 60       10505                .BYTE 96                               ; Speed 8 =
BABD 70       10506                .BYTE 112                              ; Speed 9 =
              10507
              10508 ;*** Keyboard code lookup table ***********************************
BABE F2       10509 KEYTAB         .BYTE $F2                              ; '0'   - Sp
BABF DF       10510                .BYTE $DF                              ; '1'   - Sp
BAC0 DE       10511                .BYTE $DE                              ; '2'   - Sp
BAC1 DA       10512                .BYTE $DA                              ; '3'   - Sp
BAC2 D8       10513                .BYTE $D8                              ; '4'   - Sp
BAC3 DD       10514                .BYTE $DD                              ; '5'   - Sp
BAC4 DB       10515                .BYTE $DB                              ; '6'   - Sp
BAC5 F3       10516                .BYTE $F3                              ; '7'   - Sp
BAC6 F5       10517                .BYTE $F5                              ; '8'   - Sp
BAC7 F0       10518                .BYTE $F0                              ; '9'   - Sp
BAC8 F8       10519                .BYTE $F8                              ; 'F'   - Fr
BAC9 FF       10520                .BYTE $FF                              ; 'A'   - Af
BACA C0       10521                .BYTE $C0                              ; 'L'   - Lo
BACB FD       10522                .BYTE $FD                              ; 'G'   - Ga
BACC ED       10523                .BYTE $ED                              ; 'T'   - Tr
BACD FE       10524                .BYTE $FE                              ; 'S'   - Sh
BACE D2       10525                .BYTE $D2                              ; 'C'   - At
BACF F9       10526                .BYTE $F9                              ; 'H'   - Hy
BAD0 E5       10527                .BYTE $E5                              ; 'M'   - Ma
BAD1 CA       10528                .BYTE $CA                              ; 'P'   - Pa
BAD2 E7       10529                .BYTE $E7                              ; 'INV' - Ab
              10530
              10531 ;*** Engines energy drain rates per game loop iteration in energy
BAD3 00       10532 DRAINRATETAB   .BYTE 0                                ;
BAD4 04       10533                .BYTE 4                                ;
BAD5 06       10534                .BYTE 6                                ;
BAD6 08       10535                .BYTE 8                                ;
BAD7 0A       10536                .BYTE 10                               ;
BAD8 0C       10537                .BYTE 12                               ;
BAD9 0E       10538                .BYTE 14                               ;
BADA 1E       10539                .BYTE 30                               ;
BADB 2D       10540                .BYTE 45                               ;
BADC 3C       10541                .BYTE 60                               ;
              10542
              10543 ;*** Hyperwarp energy depending on distance ***********************
BADD 0A       10544 WARPENERGYTAB  .BYTE 10                               ; =  100 ene
BADE 0D       10545                .BYTE 13                               ; =  130 ene
BADF 10       10546                .BYTE 16                               ; =  160 ene
BAE0 14       10547                .BYTE 20                               ; =  200 ene
BAE1 17       10548                .BYTE 23                               ; =  230 ene
BAE2 32       10549                .BYTE 50                               ; =  500 ene
BAE3 46       10550                .BYTE 70                               ; =  700 ene
BAE4 50       10551                .BYTE 80                               ; =  800 ene
BAE5 5A       10552                .BYTE 90                               ; =  900 ene
BAE6 78       10553                .BYTE 120                              ; = 1200 ene
BAE7 7D       10554                .BYTE 125                              ; = 1250 ene
BAE8 82       10555                .BYTE 130                              ; = 1300 ene
BAE9 87       10556                .BYTE 135                              ; = 1350 ene
BAEA 8C       10557                .BYTE 140                              ; = 1400 ene
```

```
BAEB 9B       10558                .BYTE 155                          ; = 1550 ene
BAEC AA       10559                .BYTE 170                          ; = 1700 ene
BAED B8       10560                .BYTE 184                          ; = 1840 ene
BAEE C8       10561                .BYTE 200                          ; = 2000 ene
BAEF D0       10562                .BYTE 208                          ; = 2080 ene
BAF0 D8       10563                .BYTE 216                          ; = 2160 ene
BAF1 DF       10564                .BYTE 223                          ; = 2230 ene
BAF2 E8       10565                .BYTE 232                          ; = 2320 ene
BAF3 F1       10566                .BYTE 241                          ; = 2410 ene
BAF4 FA       10567                .BYTE 250                          ; = 2500 ene
              10568
              10569 ;*** Joystick increments ***************************************
BAF5 00       10570 STICKINCTAB     .BYTE 0                           ; Centered
BAF6 01       10571                .BYTE 1                            ; Right or u
BAF7 FF       10572                .BYTE -1                           ; Left or do
BAF8 00       10573                .BYTE 0                            ; Centered
              10574
              10575 ;*** 3-byte elements to draw cross hairs and Attack Computer Displ
              10576 ;   Byte 1 : Pixel column number of line start
              10577 ;   Byte 2 : Pixel row number of line start
              10578 ;   Byte 3 : B7 = 0 -> Draw line to the right
              10579 ;            B7 = 1 -> Draw line down
              10580 ;            B6..0  -> Length of line in pixels. Possible values a
              10581 ;
              10582 ;                    #                                    
              10583 ;                    #                          4         
              10584 ;                    #                    ##############  
              10585 ;                    #1                   #             # 
              10586 ;                    #                    #             #11
              10587 ;                    #                    #             # 
              10588 ;                    #                    #5            #    8
              10589 ;                                         #      ###############
              10590 ;                                         #      #             #
              10591 ;          15                  16         #   7  #             #
              10592 ; ################       ################  ########             ##
              10593 ;                                         #      #12           # 
              10594 ;                                         #      #             #1
              10595 ;                                         #      ###############
              10596 ;                    #                    #         9   #      
              10597 ;                    #                    #             #      
              10598 ;                    #                    #             #14    
              10599 ;                    #                    #       3     #      
              10600 ;                    #2                   ##############       
              10601 ;                    #                                        
              10602 ;                    #                                        
              10603 ;
              10604 ;          Front/Aft Cross Hairs               Attack Computer Dis
              10605 ;
              10606 ; LOCAL VARIABLES
    =0080     10607 DOWN            = $80
    =0000     10608 RIGHT           = $00
              10609
BAF9 502887   10610 DRAWLINESTAB    .BYTE 80,40,DOWN!7                 ; Line 1
BAFC 503687   10611                .BYTE 80,54,DOWN!7                 ; Line 2
              10612
BAFF 77461E   10613                .BYTE 119,70,RIGHT!30              ; Line 3
BB02 77561E   10614                .BYTE 119,86,RIGHT!30              ; Line 4
BB05 774691   10615                .BYTE 119,70,DOWN!17               ; Line 5
BB08 944691   10616                .BYTE 148,70,DOWN!17               ; Line 6
```

```
BB0B 784E06   10617                .BYTE 120,78,RIGHT!6                  ; Line 7
BB0E 7E4B0F   10618                .BYTE 126,75,RIGHT!15                 ; Line 8
BB11 7E510F   10619                .BYTE 126,81,RIGHT!15                 ; Line 9
BB14 8D4E07   10620                .BYTE 141,78,RIGHT!7                  ; Line 10
BB17 854784   10621                .BYTE 133,71,DOWN!4                   ; Line 11
BB1A 7E4C85   10622                .BYTE 126,76,DOWN!5                   ; Line 12
BB1D 8C4C85   10623                .BYTE 140,76,DOWN!5                   ; Line 13
BB20 855284   10624                .BYTE 133,82,DOWN!4                   ; Line 14
              10625
BB23 3E320F   10626                .BYTE 62,50,RIGHT!15                  ; Line 15
BB26 54320F   10627                .BYTE 84,50,RIGHT!15                  ; Line 16
BB29 FE       10628                .BYTE $FE                             ; End marker
              10629
              10630 ;*** 3-byte elements to draw our starship's shape in Long-Range Sc
              10631 ;
              10632 ;   Line  17 18 19 20 21
              10633 ;               ##
              10634 ;               ##
              10635 ;            ## ## ##
              10636 ;         ## ## ## ## ##
              10637 ;         ##    ##    ##
              10638
BB2A 4E3582   10639                .BYTE 78,53,DOWN!2                    ; Line 17
BB2D 4F3482   10640                .BYTE 79,52,DOWN!2                    ; Line 18
BB30 503285   10641                .BYTE 80,50,DOWN!5                    ; Line 19
BB33 513482   10642                .BYTE 81,52,DOWN!2                    ; Line 20
BB36 523582   10643                .BYTE 82,53,DOWN!2                    ; Line 21
BB39 FE       10644                .BYTE $FE                             ; End marker
              10645
              10646 ;*** Initial x and y coordinates of a star during hyperwarp ******
              10647 ; The following two tables are used to determine the initial x and
              10648 ; (high byte) of a star during hyperwarp. An index in 0..3 picks b
              10649 ; y coordinate, thus 4 pairs of coordinates are possible:
              10650 ;
              10651 ; Y           +-------+----------------------------+--------------
              10652 ; ^           | Index |        x-coordinate        |        y-coor
              10653 ; |           +-------+----------------------------+--------------
              10654 ; |.32.       |   0   | +1024..+1279 (+$04**) <KM> |  +512..+767
              10655 ; |...1       |   1   | +1024..+1279 (+$04**) <KM> |  +768..+1023
              10656 ; |...0       |   2   |  +768..+1023 (+$03**) <KM> | +1024..+1279
              10657 ; |....       |   3   |  +512..+767  (+$02**) <KM> | +1024..+1279
              10658 ; 0----->X    +-------+----------------------------+--------------
              10659
              10660 ;*** Initial x-coordinate (high byte) of star in hyperwarp *******
BB3A 04       10661 WARPSTARXTAB    .BYTE $04                             ; +1024..+12
BB3B 04       10662                 .BYTE $04                             ; +1024..+12
BB3C 03       10663                 .BYTE $03                             ;  +768..+10
BB3D 02       10664                 .BYTE $02                             ;  +512..+76
              10665
              10666 ;*** Initial y-coordinate (high byte) of star in hyperwarp *******
BB3E 02       10667 WARPSTARYTAB    .BYTE $02                             ;  +512..+76
BB3F 03       10668                 .BYTE $03                             ;  +768..+10
BB40 04       10669                 .BYTE $04                             ; +1024..+12
BB41 04       10670                 .BYTE $04                             ; +1024..+12
              10671
              10672 ;*** Text of Control Panel Display (encoded in custom character se
              10673 ; Row 1: "V:00 K:00 E:9999 T:0"
              10674 ; Row 2: " O:-00 O:-00 R:-000 "
              10675
```

```
BB42 12       10676 PANELTXTTAB     .BYTE CCS.V
BB43 0B       10677                 .BYTE CCS.COLON
BB44 00       10678                 .BYTE CCS.0
BB45 00       10679                 .BYTE CCS.0
BB46 0A       10680                 .BYTE CCS.SPC
BB47 55       10681                 .BYTE CCS.COL1!CCS.K
BB48 4B       10682                 .BYTE CCS.COL1!CCS.COLON
BB49 40       10683                 .BYTE CCS.COL1!CCS.0
BB4A 40       10684                 .BYTE CCS.COL1!CCS.0
BB4B 0A       10685                 .BYTE CCS.SPC
BB4C 8D       10686                 .BYTE CCS.COL2!CCS.E
BB4D 8B       10687                 .BYTE CCS.COL2!CCS.COLON
BB4E 89       10688                 .BYTE CCS.COL2!CCS.9
BB4F 89       10689                 .BYTE CCS.COL2!CCS.9
BB50 89       10690                 .BYTE CCS.COL2!CCS.9
BB51 89       10691                 .BYTE CCS.COL2!CCS.9
BB52 0A       10692                 .BYTE CCS.SPC
BB53 16       10693                 .BYTE CCS.T
BB54 0B       10694                 .BYTE CCS.COLON
BB55 00       10695                 .BYTE CCS.0
              10696
BB56 0A       10697                 .BYTE CCS.SPC
BB57 14       10698                 .BYTE CCS.THETA
BB58 0B       10699                 .BYTE CCS.COLON
BB59 0F       10700                 .BYTE CCS.MINUS
BB5A 00       10701                 .BYTE CCS.0
BB5B 00       10702                 .BYTE CCS.0
BB5C 0A       10703                 .BYTE CCS.SPC
BB5D 51       10704                 .BYTE CCS.COL1!CCS.PHI
BB5E 4B       10705                 .BYTE CCS.COL1!CCS.COLON
BB5F 0F       10706                 .BYTE CCS.MINUS
BB60 00       10707                 .BYTE CCS.0
BB61 00       10708                 .BYTE CCS.0
BB62 0A       10709                 .BYTE CCS.SPC
BB63 93       10710                 .BYTE CCS.COL2!CCS.R
BB64 8B       10711                 .BYTE CCS.COL2!CCS.COLON
BB65 0F       10712                 .BYTE CCS.MINUS
BB66 00       10713                 .BYTE CCS.0
BB67 00       10714                 .BYTE CCS.0
BB68 00       10715                 .BYTE CCS.0
BB69 0A       10716                 .BYTE CCS.SPC
              10717
              10718 ;*** Text of Galactic Chart Panel Display ************************
              10719 ; Row 1: "WARP ENERGY:   0    "
              10720 ; Row 2: "TARGETS:  DC:PESCLR "
              10721 ; Row 3: "STAR DATE:00.00     "
              10722
BB6A 37       10723                 .BYTE ROM.W
BB6B 21       10724                 .BYTE ROM.A
BB6C 32       10725                 .BYTE ROM.R
BB6D 30       10726                 .BYTE ROM.P
BB6E 00       10727                 .BYTE ROM.SPC
BB6F 25       10728                 .BYTE ROM.E
BB70 2E       10729                 .BYTE ROM.N
BB71 25       10730                 .BYTE ROM.E
BB72 32       10731                 .BYTE ROM.R
BB73 27       10732                 .BYTE ROM.G
BB74 39       10733                 .BYTE ROM.Y
BB75 1A       10734                 .BYTE ROM.COLON
```

```
BB76 00       10735                .BYTE ROM.SPC
BB77 00       10736                .BYTE ROM.SPC
BB78 00       10737                .BYTE ROM.SPC
BB79 10       10738                .BYTE ROM.0
BB7A 00       10739                .BYTE ROM.SPC
BB7B 00       10740                .BYTE ROM.SPC
BB7C 00       10741                .BYTE ROM.SPC
BB7D 00       10742                .BYTE ROM.SPC
              10743
BB7E B4       10744                .BYTE CCS.COL2!ROM.T
BB7F A1       10745                .BYTE CCS.COL2!ROM.A
BB80 B2       10746                .BYTE CCS.COL2!ROM.R
BB81 A7       10747                .BYTE CCS.COL2!ROM.G
BB82 A5       10748                .BYTE CCS.COL2!ROM.E
BB83 B4       10749                .BYTE CCS.COL2!ROM.T
BB84 B3       10750                .BYTE CCS.COL2!ROM.S
BB85 9A       10751                .BYTE CCS.COL2!ROM.COLON
BB86 00       10752                .BYTE ROM.SPC
BB87 00       10753                .BYTE ROM.SPC
BB88 24       10754                .BYTE ROM.D
BB89 23       10755                .BYTE ROM.C
BB8A 1A       10756                .BYTE ROM.COLON
BB8B 30       10757                .BYTE ROM.P
BB8C 25       10758                .BYTE ROM.E
BB8D 33       10759                .BYTE ROM.S
BB8E 23       10760                .BYTE ROM.C
BB8F 2C       10761                .BYTE ROM.L
BB90 32       10762                .BYTE ROM.R
BB91 00       10763                .BYTE ROM.SPC
              10764
BB92 F3       10765                .BYTE CCS.COL3!ROM.S
BB93 F4       10766                .BYTE CCS.COL3!ROM.T
BB94 E1       10767                .BYTE CCS.COL3!ROM.A
BB95 F2       10768                .BYTE CCS.COL3!ROM.R
BB96 00       10769                .BYTE ROM.SPC
BB97 E4       10770                .BYTE CCS.COL3!ROM.D
BB98 E1       10771                .BYTE CCS.COL3!ROM.A
BB99 F4       10772                .BYTE CCS.COL3!ROM.T
BB9A E5       10773                .BYTE CCS.COL3!ROM.E
BB9B DA       10774                .BYTE CCS.COL3!ROM.COLON
BB9C D0       10775                .BYTE CCS.COL3!ROM.0
BB9D D0       10776                .BYTE CCS.COL3!ROM.0
BB9E CE       10777                .BYTE CCS.COL3!ROM.DOT
BB9F D0       10778                .BYTE CCS.COL3!ROM.0
BBA0 D0       10779                .BYTE CCS.COL3!ROM.0
BBA1 00       10780                .BYTE ROM.SPC
BBA2 00       10781                .BYTE ROM.SPC
BBA3 00       10782                .BYTE ROM.SPC
BBA4 00       10783                .BYTE ROM.SPC
BBA5 00       10784                .BYTE ROM.SPC
              10785
              10786 ;*** Galactic Chart sector type table ****************************
BBA6 CF       10787 SECTORTYPETAB   .BYTE $CF                              ; Starbase
BBA7 04       10788                .BYTE $04                              ; 4 Zylon sh
BBA8 03       10789                .BYTE $03                              ; 3 Zylon sh
BBA9 02       10790                .BYTE $02                              ; 1 or 2 Zyl
              10791
              10792 ;*** Phrase table ***********************************************
              10793 ; Phrases consist of phrase tokens. These are bytes that encode wo
```

```
                10794 ; (multiple words that fit into a single line of text), and how th
                10795 ;
                10796 ; LOCAL VARIABLES
    =0040       10797 EOP             = $40                                 ; End of phr
    =0080       10798 EOS             = $80                                 ; End of seg
    =00C0       10799 LONG            = $C0                                 ; Display ti
                10800
                10801                                                       ; Title Phra
BBAA 00         10802 PHRASETAB       .BYTE $00                             ; (unused)
BBAB 050642     10803                 .BYTE $05,$06,$02!EOP                 ; $01  "ATTA
BBAE 050643     10804                 .BYTE $05,$06,$03!EOP                 ; $04  "ATTA
BBB1 0442       10805                 .BYTE $04,$02!EOP                     ; $07  "SHIE
BBB3 0443       10806                 .BYTE $04,$03!EOP                     ; $09  "SHIE
BBB5 060742     10807                 .BYTE $06,$07,$02!EOP                 ; $0B  "COMP
BBB8 0743       10808                 .BYTE $07,$03!EOP                     ; $0E  "TRAC
BBBA 48         10809                 .BYTE $08!EOP                         ; $10  "WHAT
BBBB 094A       10810                 .BYTE $09,$0A!EOP                     ; $11  "HYPE
BBBD 0BCD       10811                 .BYTE $0B,$0D!LONG                    ; $13  "STAR
BBBF 0BCC       10812                 .BYTE $0B,$0C!LONG                    ; $15  "STAR
BBC1 094E       10813                 .BYTE $09,$0E!EOP                     ; $17  "HYPE
BBC3 094F       10814                 .BYTE $09,$0F!EOP                     ; $19  "HYPE
BBC5 D0         10815                 .BYTE $10!LONG                        ; $1B  "HYPE
BBC6 1192       10816                 .BYTE $11,$12!EOS                     ; $1C  "ORBI
BBC8 56         10817                 .BYTE $16!EOP                         ; $1E  "STAN
BBC9 134E       10818                 .BYTE $13,$0E!EOP                     ; $1F  "DOCK
BBCB 154F       10819                 .BYTE $15,$0F!EOP                     ; $21  "TRAN
BBCD B8         10820                 .BYTE $38!EOS                         ; $23  " "
BBCE 97         10821                 .BYTE $17!EOS                         ; $24  "STAR
BBCF 99         10822                 .BYTE $19!EOS                         ; $25  "ALL
BBD0 98         10823                 .BYTE $18!EOS                         ; $26  "STAR
BBD1 8C         10824                 .BYTE $0C!EOS                         ; $27  "DEST
BBD2 9D         10825                 .BYTE $1D!EOS                         ; $28  "BY Z
BBD3 1E9F       10826                 .BYTE $1E,$1F!EOS                     ; $29  "POST
BBD5 FD         10827                 .BYTE $FD                             ; $2B  "<PLA
BBD6 25FC       10828                 .BYTE $25,$FC                         ; $2C  "CLAS
BBD8 78         10829                 .BYTE $38!EOP                         ; $2E  " "
BBD9 9B         10830                 .BYTE $1B!EOS                         ; $2F  "STAR
BBDA 60         10831                 .BYTE $20!EOP                         ; $30  "COPY
BBDB B8         10832                 .BYTE $38!EOS                         ; $31  " "
BBDC 97         10833                 .BYTE $17!EOS                         ; $32  "STAR
BBDD 98         10834                 .BYTE $18!EOS                         ; $33  "STAR
BBDE 1A8E       10835                 .BYTE $1A,$0E!EOS                     ; $34  "MISS
BBE0 1C94       10836                 .BYTE $1C,$14!EOS                     ; $36  "ZERO
BBE2 249F       10837                 .BYTE $24,$1F!EOS                     ; $38  "NEW
BBE4 FD         10838                 .BYTE $FD                             ; $3A  "<PLA
BBE5 25FC       10839                 .BYTE $25,$FC                         ; $3B  "CLAS
BBE7 A7         10840                 .BYTE $27!EOS                         ; $3D  "REPO
BBE8 68         10841                 .BYTE $28!EOP                         ; $3E  "FOR
BBE9 B8         10842                 .BYTE $38!EOS                         ; $3F  " "
BBEA 97         10843                 .BYTE $17!EOS                         ; $40  "STAR
BBEB 98         10844                 .BYTE $18!EOS                         ; $41  "STAR
BBEC 1A8F       10845                 .BYTE $1A,$0F!EOS                     ; $42  "MISS
BBEE 249F       10846                 .BYTE $24,$1F!EOS                     ; $44  "NEW
BBF0 FD         10847                 .BYTE $FD                             ; $46  "<PLA
BBF1 25FC       10848                 .BYTE $25,$FC                         ; $47  "CLAS
BBF3 66         10849                 .BYTE $26!EOP                         ; $49  "CONG
BBF4 2C5A       10850                 .BYTE $2C,$1A!EOP                     ; $4A  "NOVI
BBF6 2E5A       10851                 .BYTE $2E,$1A!EOP                     ; $4C  "PILO
BBF8 315A       10852                 .BYTE $31,$1A!EOP                     ; $4E  "WARR
```

```
BBFA 335A     10853                 .BYTE $33,$1A!EOP                       ; $50  "COMM
BBFC B8       10854                 .BYTE $38!EOS                           ; $52  " "
BBFD 3476     10855                 .BYTE $34,$36!EOP                       ; $53  "DAMA
BBFF 37B5     10856                 .BYTE $37,$35!EOS                       ; $55  "PHOT
BC01 78       10857                 .BYTE $38!EOP                           ; $57  " "
BC02 378C     10858                 .BYTE $37,$0C!EOS                       ; $58  "PHOT
BC04 78       10859                 .BYTE $38!EOP                           ; $5A  " "
BC05 23B5     10860                 .BYTE $23,$35!EOS                       ; $5B  "ENGI
BC07 78       10861                 .BYTE $38!EOP                           ; $5D  " "
BC08 238C     10862                 .BYTE $23,$0C!EOS                       ; $5E  "ENGI
BC0A 78       10863                 .BYTE $38!EOP                           ; $60  " "
BC0B 04B5     10864                 .BYTE $04,$35!EOS                       ; $61  "SHIE
BC0D 78       10865                 .BYTE $38!EOP                           ; $63  " "
BC0E 048C     10866                 .BYTE $04,$0C!EOS                       ; $64  "SHIE
BC10 78       10867                 .BYTE $38!EOP                           ; $66  " "
BC11 06B5     10868                 .BYTE $06,$35!EOS                       ; $67  "COMP
BC13 78       10869                 .BYTE $38!EOP                           ; $69  " "
BC14 068C     10870                 .BYTE $06,$0C!EOS                       ; $6A  "COMP
BC16 78       10871                 .BYTE $38!EOP                           ; $6C  " "
BC17 A2       10872                 .BYTE $22!EOS                           ; $6D  "SECT
BC18 75       10873                 .BYTE $35!EOP                           ; $6E  "DAMA
BC19 A2       10874                 .BYTE $22!EOS                           ; $6F  "SECT
BC1A 4C       10875                 .BYTE $0C!EOP                           ; $70  "DEST
BC1B A1       10876                 .BYTE $21!EOS                           ; $71  "SUB-
BC1C 75       10877                 .BYTE $35!EOP                           ; $72  "DAMA
BC1D A1       10878                 .BYTE $21!EOS                           ; $73  "SUB-
BC1E 4C       10879                 .BYTE $0C!EOP                           ; $74  "DEST
BC1F C1       10880                 .BYTE $01!LONG                          ; $75  "RED
BC20 B8       10881                 .BYTE $38!EOS                           ; $76  " "
BC21 97       10882                 .BYTE $17!EOS                           ; $77  "STAR
BC22 98       10883                 .BYTE $18!EOS                           ; $78  "STAR
BC23 1A8E     10884                 .BYTE $1A,$0E!EOS                       ; $79  "MISS
BC25 249F     10885                 .BYTE $24,$1F!EOS                       ; $7B  "NEW
BC27 FD       10886                 .BYTE $FD                               ; $7D  "<PLA
BC28 25FC     10887                 .BYTE $25,$FC                           ; $7E  "CLAS
BC2A 66       10888                 .BYTE $26!EOP                           ; $80  "CONG
              10889
              10890 ;*** Word table ***********************************************
              10891 ; Bit B7 of the first byte of a word is the end-of-word marker of
              10892 ; word
              10893 ;
              10894 ; LOCAL VARIABLES
    =0080     10895 EOW             = $80                                   ; End of wor
              10896
BC2B A0202020 10897 WORDTAB         .BYTE EOW!$20,"    RED ALERT"           ; Word $01
BC2F 20524544
BC33 20414C45
BC37 5254
BC39 CF4E     10898                 .BYTE EOW!'O,"N"                        ; Word $02
BC3B CF4646   10899                 .BYTE EOW!'O,"FF"                       ; Word $03
BC3E D3484945 10900                 .BYTE EOW!'S,"HIELDS"                   ; Word $04
BC42 4C4453
BC45 C1545441 10901                 .BYTE EOW!'A,"TTACK"                    ; Word $05
BC49 434B
BC4B C34F4D50 10902                 .BYTE EOW!'C,"OMPUTER"                  ; Word $06
BC4F 55544552
BC53 D4524143 10903                 .BYTE EOW!'T,"RACKING"                  ; Word $07
BC57 4B494E47
BC5B D7484154 10904                 .BYTE EOW!'W,"HATS WRONG?"              ; Word $08
```

```
BC5F 53205752
BC63 4F4E473F
BC67 C8595045 10905                .BYTE EOW!'H,"YPERWARP"              ; Word $09
BC6B 52574152
BC6F 50
BC70 C54E4741 10906                .BYTE EOW!'E,"NGAGED"                ; Word $0A
BC74 474544
BC77 D3544152 10907                .BYTE EOW!'S,"TARBASE"               ; Word $0B
BC7B 42415345
BC7F C4455354 10908                .BYTE EOW!'D,"ESTROYED"              ; Word $0C
BC83 524F5945
BC87 44
BC88 D3555252 10909                .BYTE EOW!'S,"URROUNDED"             ; Word $0D
BC8C 4F554E44
BC90 4544
BC92 C1424F52 10910                .BYTE EOW!'A,"BORTED"                ; Word $0E
BC96 544544
BC99 C34F4D50 10911                .BYTE EOW!'C,"OMPLETE"               ; Word $0F
BC9D 4C455445
BCA1 C8595045 10912                .BYTE EOW!'H,"YPERSPACE"             ; Word $10
BCA5 52535041
BCA9 4345
BCAB CF524249 10913                .BYTE EOW!'O,"RBIT"                  ; Word $11
BCAF 54
BCB0 C5535441 10914                .BYTE EOW!'E,"STABLISHED"            ; Word $12
BCB4 424C4953
BCB8 484544
BCBB C44F434B 10915                .BYTE EOW!'D,"OCKING"                ; Word $13
BCBF 494E47
BCC2 C54E4552 10916                .BYTE EOW!'E,"NERGY"                 ; Word $14
BCC6 4759
BCC8 D452414E 10917                .BYTE EOW!'T,"RANSFER"               ; Word $15
BCCC 53464552
BCD0 D354414E 10918                .BYTE EOW!'S,"TANDBY"                ; Word $16
BCD4 444259
BCD7 D3544152 10919                .BYTE EOW!'S,"TAR FLEET TO"          ; Word $17
BCDB 20464C45
BCDF 45542054
BCE3 4F
BCE4 D3544152 10920                .BYTE EOW!'S,"TAR CRUISER 7"         ; Word $18
BCE8 20435255
BCEC 49534552
BCF0 2037
BCF2 C14C4C20 10921                .BYTE EOW!'A,"LL UNITS"              ; Word $19
BCF6 554E4954
BCFA 53
BCFB CD495353 10922                .BYTE EOW!'M,"ISSION"                ; Word $1A
BCFF 494F4E
BD02 A0202020 10923                .BYTE EOW!$20,"   STAR RAIDERS"      ; Word $1B
BD06 53544152
BD0A 20524149
BD0E 44455253
BD12 DA45524F 10924                .BYTE EOW!'Z,"ERO"                   ; Word $1C
BD16 C259205A 10925                .BYTE EOW!'B,"Y ZYLON FIRE"          ; Word $1D
BD1A 594C4F4E
BD1E 20464952
BD22 45
BD23 D04F5354 10926                .BYTE EOW!'P,"OSTHUMOUS"             ; Word $1E
BD27 48554D4F
```

```
BD2B 5553
BD2D D2414E4B 10927               .BYTE EOW!'R,"ANK IS:"                ; Word $1F
BD31 2049533A
BD35 C34F5059 10928               .BYTE EOW!'C,"OPYRIGHT ATARI 1979"    ; Word $20
BD39 52494748
BD3D 54204154
BD41 41524920
BD45 31393739
BD49 D355422D 10929               .BYTE EOW!'S,"UB-SPACE RADIO"         ; Word $21
BD4D 53504143
BD51 45205241
BD55 44494F
BD58 D3454354 10930               .BYTE EOW!'S,"ECTOR SCAN"             ; Word $22
BD5C 4F522053
BD60 43414E
BD63 C54E4749 10931               .BYTE EOW!'E,"NGINES"                 ; Word $23
BD67 4E4553
BD6A CE4557   10932               .BYTE EOW!'N,"EW"                     ; Word $24
BD6D C34C4153 10933               .BYTE EOW!'C,"LASS"                   ; Word $25
BD71 53
BD72 C34F4E47 10934               .BYTE EOW!'C,"ONGRATULATIONS"         ; Word $26
BD76 52415455
BD7A 4C415449
BD7E 4F4E53
BD81 D245504F 10935               .BYTE EOW!'R,"EPORT TO BASE"          ; Word $27
BD85 52542054
BD89 4F204241
BD8D 5345
BD8F C64F5220 10936               .BYTE EOW!'F,"OR TRAINING"            ; Word $28
BD93 54524149
BD97 4E494E47
BD9B C7414C41 10937               .BYTE EOW!'G,"ALACTIC COOK"           ; Word $29
BD9F 43544943
BDA3 20434F4F
BDA7 4B
BDA8 C7415242 10938               .BYTE EOW!'G,"ARBAGE SCOW CAPTAIN"    ; Word $2A
BDAC 41474520
BDB0 53434F57
BDB4 20434150
BDB8 5441494E
BDBC D24F4F4B 10939               .BYTE EOW!'R,"OOKIE"                  ; Word $2B
BDC0 4945
BDC2 CE4F5649 10940               .BYTE EOW!'N,"OVICE"                  ; Word $2C
BDC6 4345
BDC8 C54E5349 10941               .BYTE EOW!'E,"NSIGN"                  ; Word $2D
BDCC 474E
BDCE D0494C4F 10942               .BYTE EOW!'P,"ILOT"                   ; Word $2E
BDD2 54
BDD3 C14345   10943               .BYTE EOW!'A,"CE"                     ; Word $2F
BDD6 CC494555 10944               .BYTE EOW!'L,"IEUTENANT"              ; Word $30
BDDA 54454E41
BDDE 4E54
BDE0 D7415252 10945               .BYTE EOW!'W,"ARRIOR"                 ; Word $31
BDE4 494F52
BDE7 C3415054 10946               .BYTE EOW!'C,"APTAIN"                 ; Word $32
BDEB 41494E
BDEE C34F4D4D 10947               .BYTE EOW!'C,"OMMANDER"               ; Word $33
BDF2 414E4445
BDF6 52
```

```
BDF7 C4414D41 10948                .BYTE EOW!'D,"AMAGE"                  ; Word $34
BDFB 4745
BDFD C4414D41 10949                .BYTE EOW!'D,"AMAGED"                 ; Word $35
BE01 474544
BE04 C34F4E54 10950                .BYTE EOW!'C,"ONTROL"                 ; Word $36
BE08 524F4C
BE0B D0484F54 10951                .BYTE EOW!'P,"HOTONS"                 ; Word $37
BE0F 4F4E53
BE12 A0       10952                .BYTE EOW!$20                         ; Word $38
BE13 D3544152 10953                .BYTE EOW!'S,"TAR COMMANDER"          ; Word $39
BE17 20434F4D
BE1B 4D414E44
BE1F 4552
BE21 80       10954                .BYTE EOW!$00                         ;
              10955
              10956 ;*** View modes ***********************************************
BE22 00       10957 VIEWMODETAB    .BYTE $00                             ; Front view
BE23 01       10958                .BYTE $01                             ; Aft view
BE24 40       10959                .BYTE $40                             ; Long-Range
BE25 80       10960                .BYTE $80                             ; Galactic C
              10961
              10962 ;*** Title phrase offsets of "TRACKING OFF", "SHIELDS OFF", "COMPU
BE26 0E       10963 MSGOFFTAB      .BYTE $0E                             ; "TRACKING
BE27 09       10964                .BYTE $09                             ; "SHIELDS O
BE28 04       10965                .BYTE $04                             ; "COMPUTER
              10966
              10967 ;*** Masks to test if Tracking Computer, Shields, or Attack Comput
BE29 FF       10968 MSGBITTAB      .BYTE $FF                             ; Mask Track
BE2A 08       10969                .BYTE $08                             ; Mask Shiel
BE2B 02       10970                .BYTE $02                             ; Mask Attac
              10971
              10972 ;*** Title phrase offsets of "COMPUTER TRACKING ON", "SHIELDS ON",
BE2C 0B       10973 MSGONTAB       .BYTE $0B                             ; "COMPUTER
BE2D 07       10974                .BYTE $07                             ; "SHIELDS O
BE2E 01       10975                .BYTE $01                             ; "COMPUTER
              10976
              10977 ;*** The following two tables encode the PLAYER shapes ***********
              10978 ;
              10979 ; PHOTON TORPEDO (shape type 0, data in shape table PLSHAP1TAB)
              10980 ; Numbers at top indicate the shape table offset of the first and
              10981 ;
              10982 ; $01..$10  $11..$1E  $1F..$2A  $2B..$34  $35..$3C  $3D..$42  $75.
              10983 ; ...##...  ...#....  ...##...  ...#....  ...#....  ...#....  ...#
              10984 ; ..####..  ..###...  ..####..  ..###...  ...##...  ..###...  ...#
              10985 ; .######.  .#####..  ..####..  ..###...  ..####..  ..###...
              10986 ; .######.  .#####..  .######.  .#####..  ..#.##..  ..#.#...
              10987 ; .###.##.  #######.  .##.###.  .###.#..  ..####..  ..###...
              10988 ; ####.###  ##.####.  .####.#.  .#####..  ..####..  ...#....
              10989 ; ##.#####  ##.##.#.  .######.  .##.##..  ...##...
              10990 ; ##.#####  #####.#.  .###.##.  ..###...  ....#...
              10991 ; ########  ###.###.  .######.  ..###...
              10992 ; ########  ###.###.  ..####..  ...#....
              10993 ; ####.###  .#####..  ..####..
              10994 ; .###.##.  .#####..  ...##...
              10995 ; .######.  ..###...
              10996 ; .######.  ...#....
              10997 ; ..####..
              10998 ; ...##...
              10999 ;
```

```
11000 ; ZYLON FIGHTER (shape type 1, data in shape table PLSHAP2TAB)
11001 ; Numbers at top indicate the shape table offset of the first and
11002 ;
11003 ; $01..$0C  $0D..$14  $15..$1A  $1B..$20  $21..$24  $25..$28  $29.
11004 ; #......#  #.....#.  .#....#.  .#...#..  ..#..#..  ..#.#...  ...#
11005 ; #......#  #.....#.  .#.##.#.  .#.#.#..  ..####..  ..###...  ...#
11006 ; #......#  #.###.#.  .######.  .#####..  ..####..  ..###...
11007 ; #......#  #######.  .######.  .#####..  ..#..#..  ..#.#...
11008 ; #.####.#  #######.  .#.##.#.  .#.#.#..
11009 ; ########  #.###.#.  .#....#.  .#...#..
11010 ; ########  #.....#.
11011 ; #.####.#  #.....#.
11012 ; #......#
11013 ; #......#
11014 ; #......#
11015 ; #......#
11016 ;
11017 ; STARBASE RIGHT (shape type 2, data in shape table PLSHAP2TAB)
11018 ; Numbers at top indicate the shape table offset of the first and
11019 ;
11020 ; $2D..$36  $38..$40  $41..$46  $36..$38  $36       $00       $00
11021 ; ###.....  ##......  ##......  ##......  ##......  ........  ....
11022 ; #####...  ####....  ##......  ####....
11023 ; #####...  ####....  ####....  ##......
11024 ; #######.  ######..  #.####..
11025 ;  #.#.###  #.#####.  ####....
11026 ; #######.  ######..  ##......
11027 ; #####...  ####....
11028 ; #####...  #.......
11029 ; ##......  #.......
11030 ; ##......
11031 ;
11032 ; STARBASE CENTER (shape type 3, data in shape table PLSHAP1TAB)
11033 ; Numbers at top indicate the shape table offset of the first and
11034 ;
11035 ; $7E..$8D  $8E..$9C  $9D..$A9  $AA..$B3  $B4..$BB  $BC..$C0  $7B.
11036 ; ...##...  ........  ........  ...##...  ........  ...##...  ...#
11037 ; .######.  ...##...  ...##...  ..####..  ...##...  ..####..  ..##
11038 ; ########  ..####..  ..####..  ########  ..####..  ########  ...#
11039 ; ########  .######.  .######.  ########  ########  ..####..
11040 ; ########  ########  ########  ###..###  ########  ...##...
11041 ; ########  ########  ########  .##..##.  ########
11042 ; ########  ########  ###..###  ########  ..####..
11043 ; ###..###  ###..###  .##..##.  ########  ...##...
11044 ; ###..###  .##..##.  ########  .######.
11045 ; ########  ########  ########  ..####..
11046 ; ########  ########  ########
11047 ; ########  ########  ########
11048 ; ########  ########  ..####..
11049 ; ########  .######.
11050 ; .######.  .######.
11051 ; .######.
11052 ;
11053 ; STARBASE LEFT (shape type 4, data in shape table PLSHAP2TAB)
11054 ; Numbers at top indicate the shape table offset of the first and
11055 ;
11056 ; $47..$50  $52..$5A  $5B..$60  $50..$52  $50       $00       $00
11057 ; .....###  ......##  ......##  ......##  ......##  ........  ....
11058 ; ...#####  ....####  ......##  ....####
```

```
11059 ; ...#####  ....####  ....####  ......##
11060 ; .#######  ..######  ..####.#
11061 ; ###.#.#.  .#####.#  ....####
11062 ; .#######  ..######  ......##
11063 ; ...#####  ....####
11064 ; ...#####  .......#
11065 ; ......##  .......#
11066 ; ......##
11067 ;
11068 ; TRANSFER VESSEL (shape type 5, data in shape table PLSHAP1TAB)
11069 ; Numbers at top indicate the shape table offset of the first and
11070 ;
11071 ; $43..$52  $53..$60  $61..$6B  $6C..$74  $75..$79  $7A..$7D  $75.
11072 ; ..####..  ...###..  ...##...  ...#....  ...##...  ...#....  ...#
11073 ; ..####..  ...###..  ...##...  ...#....  ...##...  ...#....  ...#
11074 ; ..#..#..  ...#.#..  ..####..  ..###...  ..####..  ..###...
11075 ; ..####..  ..#####.  ..####..  ..###...  ...##...  ...#....
11076 ; .######.  ..#####.  ..####..  ..###...  ...##...
11077 ; .######.  ..#####.  ..####..  .#####..
11078 ; .######.  ..#.#.#.  .######.  ..#.#...
11079 ; .#.##.#.  .#######  ..#..#..  ..#.#...
11080 ; ########  .#######  ..#..#..  ..#.#...
11081 ; ########  ..#...#.  ..#..#..
11082 ; .#....#.  ..#...#.  ..#..#..
11083 ; .#....#.  ..#...#.
11084 ; .#....#.  ..#...#.
11085 ; .#....#.  ..#...#.
11086 ; .#....#.
11087 ; .#....#.
11088 ;
11089 ; METEOR (shape type 6, data in shape table PLSHAP1TAB)
11090 ; Numbers at top indicate the shape table offset of the first and
11091 ;
11092 ; $01..$10  $11..$1E  $1F..$2A  $2B..$34  $35..$3C  $3D..$42  $75.
11093 ; ...##...  ...#....  ...##...  ...#....  ...#....  ...#....  ...#
11094 ; ..####..  ..###...  ..####..  ..###...  ...##...  ..###...  ...#
11095 ; .######.  .#####..  ..####..  ..###...  ..####..  ..###...
11096 ; .######.  .#####..  .######.  .#####..  ..#.##..  ..#.#...
11097 ; .###.##.  #######.  .##.###.  .###.#..  ..####..  ..###...
11098 ; ####.###  ##.####.  .####.#.  .#####..  ..####..  ...#....
11099 ; ##.#####  ##.##.#.  .######.  .##.##..  ...##...
11100 ; ##.#####  #####.#.  .###.##.  ..###...  ....#...
11101 ; ########  ###.###.  .######.  ..###...
11102 ; ########  ###.###.  ..####..  ...#....
11103 ; ####.###  .#####..  ..####..
11104 ; .###.##.  .#####..  ...##...
11105 ; .######.  ..###...
11106 ; .######.  ...#....
11107 ; ..####..
11108 ; ...##...
11109 ;
11110 ; ZYLON CRUISER (shape type 7, data in shape table PLSHAP2TAB)
11111 ; Numbers at top indicate the shape table offset of the first and
11112 ;
11113 ; $61..$69  $6A..$71  $72..$78  $79..$7E  $7F..$82  $83..$85  $29.
11114 ; ...##...  ...#....  ...##...  ...#....  ...##...  ...#....  ...#
11115 ; ..####..  ..###...  ..####..  ..###...  ..####..  ..###...  ...#
11116 ; .######.  .#####..  ..####..  ..###...  ..#..#..  ..#.#...
11117 ; .######.  .#####..  .##..##.  .##.##..  ..#..#..
```

```
                11118 ; ##.##.##  ##.#.##.  .##..##.  .#...#..
                11119 ; ##....##  ##...##.  .#....#.  .#...#..
                11120 ; #......#  #.....#.  .#....#.
                11121 ; #......#  #.....#.
                11122 ; #......#
                11123 ;
                11124 ; ZYLON BASESTAR (shape type 8, data in shape table PLSHAP2TAB)
                11125 ; Numbers at top indicate the shape table offset of the first and
                11126 ;
                11127 ; $86..$8F  $90..$99  $9A..$A0  $A1..$A7  $A8..$AC  $AD..$B0  $29.
                11128 ; ...##...  ...#....  ...##...  ...#....  ...##...  ...#....  ...#
                11129 ; ..####..  ..###...  ..####..  ..###...  ..####..  ..###...  ...#
                11130 ; .######.  .#####..  .######.  .#####..  ...##...  ..###...
                11131 ; ########  #######.  ...##...  ...#....  ..####..  ...#....
                11132 ; ...##...  ..###...  .######.  .#####..  ...##...
                11133 ; ...##...  ..###...  ..####..  ..###...
                11134 ; ########  #######.  ...##...  ...#....
                11135 ; .######.  .#####..
                11136 ; ..####..  ..###...
                11137 ; ...##...  ...#....
                11138 ;
                11139 ; HYPERWARP TARGET MARKER (shape type 9, data in shape table PLSHA
                11140 ; Numbers at top indicate the shape table offset of the first and
                11141 ;
                11142 ; $C1..$CC  $C1..$CC  $C1..$CC  $C1..$CC  $C1..$CC  $C1..$CC  $75.
                11143 ; ..#.#...  ..#.#...  ..#.#...  ..#.#...  ..#.#...  ..#.#...  ...#
                11144 ; ..#.#...  ..#.#...  ..#.#...  ..#.#...  ..#.#...  ..#.#...  ...#
                11145 ; ..#.#...  ..#.#...  ..#.#...  ..#.#...  ..#.#...  ..#.#...
                11146 ; ..#.#...  ..#.#...  ..#.#...  ..#.#...  ..#.#...  ..#.#...
                11147 ; ###.###.  ###.###.  ###.###.  ###.###.  ###.###.  ###.###.
                11148 ; ........  ........  ........  ........  ........  ........
                11149 ; ........  ........  ........  ........  ........  ........
                11150 ; ###.###.  ###.###.  ###.###.  ###.###.  ###.###.  ###.###.
                11151 ; ..#.#...  ..#.#...  ..#.#...  ..#.#...  ..#.#...  ..#.#...
                11152 ; ..#.#...  ..#.#...  ..#.#...  ..#.#...  ..#.#...  ..#.#...
                11153 ; ..#.#...  ..#.#...  ..#.#...  ..#.#...  ..#.#...  ..#.#...
                11154 ; ..#.#...  ..#.#...  ..#.#...  ..#.#...  ..#.#...  ..#.#...
                11155
                11156 ;*** Shape type 0..9 offset table (10 shape cell offsets of shape
BE2F 01111F2B   11157 PLSHAPOFFTAB    .BYTE $01,$11,$1F,$2B,$35,$3D,$75,$7A ; ...0 into
BE33 353D757A
BE37 010D151B   11158                 .BYTE $01,$0D,$15,$1B,$21,$25,$29,$2B ; ...1 into
BE3B 2125292B
BE3F 2D384136   11159                 .BYTE $2D,$38,$41,$36,$36,$00,$00,$00 ; ...2 into
BE43 36000000
BE47 7E8E9DAA   11160                 .BYTE $7E,$8E,$9D,$AA,$B4,$BC,$7B,$7A ; ...3 into
BE4B B4BC7B7A
BE4F 47525B50   11161                 .BYTE $47,$52,$5B,$50,$50,$00,$00,$00 ; ...4 into
BE53 50000000
BE57 4353616C   11162                 .BYTE $43,$53,$61,$6C,$75,$7A,$75,$7A ; ...5 into
BE5B 757A757A
BE5F 01111F2B   11163                 .BYTE $01,$11,$1F,$2B,$35,$3D,$75,$7A ; ...6 into
BE63 353D757A
BE67 616A7279   11164                 .BYTE $61,$6A,$72,$79,$7F,$83,$29,$2B ; ...7 into
BE6B 7F83292B
BE6F 86909AA1   11165                 .BYTE $86,$90,$9A,$A1,$A8,$AD,$29,$2B ; ...8 into
BE73 A8AD292B
BE77 C1C1C1C1   11166                 .BYTE $C1,$C1,$C1,$C1,$C1,$C1,$75,$C1 ; ...9 into
BE7B C1C175C1
```

```
                11167
                11168 ;*** Shape type 0..9 height table (10 shape cell heights of shape
BE7F 0F0D0B09   11169 PLSHAPEIGHTTAB .BYTE $0F,$0D,$0B,$09,$07,$05,$01,$01 ; ...0
BE83 07050101
BE87 0B070505   11170                .BYTE $0B,$07,$05,$05,$03,$03,$01,$01 ; ...1
BE8B 03030101
BE8F 09080502   11171                .BYTE $09,$08,$05,$02,$00,$00,$00,$00 ; ...2
BE93 00000000
BE97 0F0E0C09   11172                .BYTE $0F,$0E,$0C,$09,$07,$04,$02,$01 ; ...3
BE9B 07040201
BE9F 09080502   11173                .BYTE $09,$08,$05,$02,$00,$00,$00,$00 ; ...4
BEA3 00000000
BEA7 0F0D0A08   11174                .BYTE $0F,$0D,$0A,$08,$04,$03,$01,$01 ; ...5
BEAB 04030101
BEAF 0F0D0B09   11175                .BYTE $0F,$0D,$0B,$09,$07,$05,$01,$01 ; ...6
BEB3 07050101
BEB7 08070605   11176                .BYTE $08,$07,$06,$05,$03,$02,$01,$01 ; ...7
BEBB 03020101
BEBF 09090606   11177                .BYTE $09,$09,$06,$06,$04,$03,$01,$01 ; ...8
BEC3 04030101
BEC7 0B0B0B0B   11178                .BYTE $0B,$0B,$0B,$0B,$0B,$0B,$01,$0B ; ...9
BECB 0B0B010B
                11179
                11180 ;*** Keyboard codes to switch to Front or Aft view when Tracking C
BECF F8         11181 TRACKKEYSTAB    .BYTE $F8                             ; 'F' - Fron
BED0 FF         11182                 .BYTE $FF                             ; 'A' - Aft
                11183
                11184 ;*** Galactic Chart sector character codes (encoded in custom char
BED1 0C         11185 SECTORCHARTAB   .BYTE CCS.BORDERSW                    ; Empty sect
BED2 1E         11186                 .BYTE CCS.2ZYLONS                     ; Sector con
BED3 1E         11187                 .BYTE CCS.2ZYLONS                     ; Sector con
BED4 1D         11188                 .BYTE CCS.3ZYLONS                     ; Sector con
BED5 1C         11189                 .BYTE CCS.4ZYLONS                     ; Sector con
BED6 1B         11190                 .BYTE CCS.STARBASE                    ; Sector con
                11191
                11192 ;*** Mask to limit veer-off velocity of Hyperwarp Target Marker in
BED7 9F         11193 VEERMASKTAB     .BYTE NEG!31                          ;  -31..+31
BED8 BF         11194                 .BYTE NEG!63                          ;  -63..+63
BED9 DF         11195                 .BYTE NEG!95                          ;  -95..+95
BEDA FF         11196                 .BYTE NEG!127                         ; -127..+127
                11197
                11198 ;*** Horizontal PLAYER offsets for PLAYER0..1 (STARBASE LEFT, STAR
BEDB F8         11199 PLSTARBAOFFTAB  .BYTE -8                              ; -8 Player/
BEDC 08         11200                 .BYTE 8                               ; +8 Player/
                11201
                11202 ;*** Mission bonus table ****************************************
BEDD 50         11203 BONUSTAB        .BYTE 80                              ; Mission co
BEDE 4C         11204                 .BYTE 76                              ; Mission co
BEDF 3C         11205                 .BYTE 60                              ; Mission co
BEE0 6F         11206                 .BYTE 111                             ; Mission co
                11207
BEE1 3C         11208                 .BYTE 60                              ; Mission ab
BEE2 3C         11209                 .BYTE 60                              ; Mission ab
BEE3 32         11210                 .BYTE 50                              ; Mission ab
BEE4 64         11211                 .BYTE 100                             ; Mission ab
                11212
BEE5 28         11213                 .BYTE 40                              ; Starship d
BEE6 32         11214                 .BYTE 50                              ; Starship d
BEE7 28         11215                 .BYTE 40                              ; Starship d
```

```
BEE8 5A       11216                .BYTE 90                          ; Starship d
              11217
              11218 ;*** Title phrase offsets of scored class rank ******************
BEE9 A9       11219 RANKTAB        .BYTE $29!EOS                     ; "GALACTIC
BEEA AA       11220                .BYTE $2A!EOS                     ; "GARBAGE S
BEEB AA       11221                .BYTE $2A!EOS                     ; "GARBAGE S
BEEC AB       11222                .BYTE $2B!EOS                     ; "ROOKIE"
BEED AB       11223                .BYTE $2B!EOS                     ; "ROOKIE"
BEEE AC       11224                .BYTE $2C!EOS                     ; "NOVICE"
BEEF AC       11225                .BYTE $2C!EOS                     ; "NOVICE"
BEF0 AD       11226                .BYTE $2D!EOS                     ; "ENSIGN"
BEF1 AD       11227                .BYTE $2D!EOS                     ; "ENSIGN"
BEF2 AE       11228                .BYTE $2E!EOS                     ; "PILOT"
BEF3 AE       11229                .BYTE $2E!EOS                     ; "PILOT"
BEF4 AF       11230                .BYTE $2F!EOS                     ; "ACE"
BEF5 B0       11231                .BYTE $30!EOS                     ; "LIEUTENAN
BEF6 B1       11232                .BYTE $31!EOS                     ; "WARRIOR"
BEF7 B2       11233                .BYTE $32!EOS                     ; "CAPTAIN"
BEF8 B3       11234                .BYTE $33!EOS                     ; "COMMANDER
BEF9 B3       11235                .BYTE $33!EOS                     ; "COMMANDER
BEFA B9       11236                .BYTE $39!EOS                     ; "STAR COMM
BEFB B9       11237                .BYTE $39!EOS                     ; "STAR COMM
              11238
              11239 ;*** Scored class number table **********************************
BEFC 95       11240 CLASSTAB       .BYTE CCS.COL2!ROM.5              ; Class 5
BEFD 95       11241                .BYTE CCS.COL2!ROM.5              ; Class 5
BEFE 95       11242                .BYTE CCS.COL2!ROM.5              ; Class 5
BEFF 94       11243                .BYTE CCS.COL2!ROM.4              ; Class 4
BF00 94       11244                .BYTE CCS.COL2!ROM.4              ; Class 4
BF01 94       11245                .BYTE CCS.COL2!ROM.4              ; Class 4
BF02 94       11246                .BYTE CCS.COL2!ROM.4              ; Class 4
BF03 93       11247                .BYTE CCS.COL2!ROM.3              ; Class 3
BF04 93       11248                .BYTE CCS.COL2!ROM.3              ; Class 3
BF05 93       11249                .BYTE CCS.COL2!ROM.3              ; Class 3
BF06 92       11250                .BYTE CCS.COL2!ROM.2              ; Class 2
BF07 92       11251                .BYTE CCS.COL2!ROM.2              ; Class 2
BF08 92       11252                .BYTE CCS.COL2!ROM.2              ; Class 2
BF09 91       11253                .BYTE CCS.COL2!ROM.1              ; Class 1
BF0A 91       11254                .BYTE CCS.COL2!ROM.1              ; Class 1
BF0B 91       11255                .BYTE CCS.COL2!ROM.1              ; Class 1
              11256
              11257 ;*** Title phrase offsets of mission level **********************
BF0C 4A       11258 MISSIONPHRTAB  .BYTE $4A                         ; "NOVICE MI
BF0D 4C       11259                .BYTE $4C                         ; "PILOT MIS
BF0E 4E       11260                .BYTE $4E                         ; "WARRIOR M
BF0F 50       11261                .BYTE $50                         ; "COMMANDER
              11262
              11263 ;*** Damage probability of subsystems depending on mission level *
BF10 00       11264 DAMAGEPROBTAB  .BYTE 0                           ;  0% (  0:2
BF11 50       11265                .BYTE 80                          ; 31% ( 80:2
BF12 B4       11266                .BYTE 180                         ; 70% (180:2
BF13 FE       11267                .BYTE 254                         ; 99% (254:2
              11268
              11269 ;*** Title phrase offsets of damaged subsystems *****************
BF14 55       11270 DAMAGEPHRTAB   .BYTE $55                         ; "PHOTON TO
BF15 5B       11271                .BYTE $5B                         ; "ENGINES D
BF16 61       11272                .BYTE $61                         ; "SHIELDS D
BF17 67       11273                .BYTE $67                         ; "COMPUTER
BF18 6D       11274                .BYTE $6D                         ; "LONG RANG
```

```
BF19 71       11275                .BYTE $71                          ; "SUB-SPACE
              11276
              11277 ;*** Title phrase offsets of destroyed subsystems ***************
BF1A 58       11278 DESTROYPHRTAB   .BYTE $58                          ; "PHOTON TO
BF1B 5E       11279                .BYTE $5E                          ; "ENGINES D
BF1C 64       11280                .BYTE $64                          ; "SHIELDS D
BF1D 6A       11281                .BYTE $6A                          ; "COMPUTER
BF1E 6F       11282                .BYTE $6F                          ; "LONG RANG
BF1F 73       11283                .BYTE $73                          ; "SUB-SPACE
              11284
              11285 ;*** 3 x 10-byte noise sound patterns (bytes 0..7 stored in revers
              11286 ;
              11287 ; (9) AUDCTL        ($D208) POKEY: Audio control
              11288 ; (8) AUDF3         ($D204) POKEY: Audio channel 3 frequency
              11289 ; (7) NOISETORPTIM  ($DA)   Timer for PHOTON TORPEDO LAUNCHED nois
              11290 ; (6) NOISEEXPLTIM  ($DB)   Timer for SHIELD and ZYLON EXPLOSION n
              11291 ; (5) NOISEAUDC2    ($DC)   Audio channel 1/2 control shadow regis
              11292 ; (4) NOISEAUDC3    ($DD)   Audio channel 3   control shadow regis
              11293 ; (3) NOISEAUDF1    ($DE)   Audio channel 1 frequency shadow regis
              11294 ; (2) NOISEAUDF2    ($DF)   Audio channel 2 frequency shadow regis
              11295 ; (1) NOISEFRQINC   ($E0)   Audio channel 1/2 frequency increment
              11296 ; (0) NOISELIFE     ($E1)   Noise sound pattern lifetime
              11297 ;
              11298 ;                     (0),(1),(2),(3),(4),(5),(6),(7),(8),(9)
BF20 18FF0200 11299 NOISEPATTAB     .BYTE $18,$FF,$02,$00,$8A,$A0,$00,$08,$50,$00; PHO
BF24 8AA00008
BF28 5000
BF2A 40400103 11300                .BYTE $40,$40,$01,$03,$88,$AF,$08,$00,$50,$04; SHI
BF2E 88AF0800
BF32 5004
BF34 30400103 11301                .BYTE $30,$40,$01,$03,$84,$A8,$04,$00,$50,$04; ZYL
BF38 84A80400
BF3C 5004
              11302
              11303 ;*** 5 x 6-byte beeper sound patterns (bytes 0..5 stored in revers
              11304 ;
              11305 ; (5) BEEPFRQIND    ($D2) Running index into frequency table BEEPF
              11306 ; (4) BEEPREPEAT    ($D3) Number of times the beeper sound pattern
              11307 ; (3) BEEPTONELIFE  ($D4) Lifetime of tone in TICKs - 1
              11308 ; (2) BEEPPAUSELIFE ($D5) Lifetime of pause in TICKs - 1 ($FF -> N
              11309 ; (1) BEEPPRIORITY  ($D6) Beeper sound pattern priority. A playing
              11310 ;                         stopped if a beeper sound pattern of hig
              11311 ;                         played. A value of 0 indicates that no b
              11312 ;                         playing at the moment.
              11313 ; (0) BEEPFRQSTART  ($D7) Index to first byte of the beeper sound
              11314 ;                         BEEPFRQTAB ($BF5C)
              11315 ;
              11316 ; Frequency-over-TICKs diagrams for all beeper sound patterns:
              11317 ;
              11318 ; HYPERWARP TRANSIT
              11319 ;
              11320 ;      FRQ
              11321 ;       |
              11322 ;   $18 |-4--
              11323 ;       |
              11324 ;   $00 |    -3-
              11325 ;       +-------> TICKS
              11326 ;        <13 x >
              11327 ;
```

```
                11328 ; RED ALERT
                11329 ;
                11330 ;      FRQ
                11331 ;       |
                11332 ;   $60 |                 --------17-------
                11333 ;       |
                11334 ;   $40 |--------17-------
                11335 ;       |
                11336 ;       +---------------------------------> TICKS
                11337 ;        <-------------- 8 x ------------->
                11338 ;
                11339 ; ACKNOWLEDGE
                11340 ;
                11341 ;      FRQ
                11342 ;       |
                11343 ;   $10 |-3-   -3-   -3-
                11344 ;       |
                11345 ;   $00 |   -3-   -3-   -3-
                11346 ;       +------------------> TICKS
                11347 ;        <------ 1 x ----->
                11348 ;
                11349 ; DAMAGE REPORT (not to scale)
                11350 ;
                11351 ;      FRQ
                11352 ;       |
                11353 ;   $40 |------------33-------------
                11354 ;       |
                11355 ;   $20 |                           ------------33-------------
                11356 ;       |
                11357 ;       +-----------------------------------------------------> T
                11358 ;        <----------------------- 3 x ----------------------->
                11359 ;
                11360 ; MESSAGE FROM STARBASE (not to scale)
                11361 ;
                11362 ;      FRQ
                11363 ;       |
                11364 ;   $51 |                                   -----33-----
                11365 ;   $48 |-----33-----
                11366 ;   $40 |                 -----33-----
                11367 ;       |
                11368 ;   $00 |            --9--            --9--            --9--
                11369 ;       +-----------------------------------------------------> TIC
                11370 ;        <---------------------- 1 x ---------------------->
                11371 ;
                11372 ;                     (0),(1),(2),(3),(4),(5)
BF3E 02020203   11373 BEEPPATTAB      .BYTE $02,$02,$02,$03,$0C,$02         ; HYPERWARP
BF42 0C02
BF44 0403FF10   11374                 .BYTE $04,$03,$FF,$10,$07,$04         ; RED ALERT
BF48 0704
BF4A 07040202   11375                 .BYTE $07,$04,$02,$02,$00,$07         ; ACKNOWLEDG
BF4E 0007
BF50 0B05FF20   11376                 .BYTE $0B,$05,$FF,$20,$02,$0B         ; DAMAGE REP
BF54 020B
BF56 0E060820   11377                 .BYTE $0E,$06,$08,$20,$00,$0E         ; MESSAGE FR
BF5A 000E
                11378
                11379 ;*** Beeper sound pattern frequency table ***********************
BF5C 10FF       11380 BEEPFRQTAB      .BYTE $10,$FF                         ; (unused) (
BF5E 18FF       11381                 .BYTE $18,$FF                         ; HYPERWARP
```

```
BF60 4060FF   11382                .BYTE $40,$60,$FF                    ; RED ALERT
BF63 101010FF 11383                .BYTE $10,$10,$10,$FF                ; ACKNOWLEDG
BF67 4020FF   11384                .BYTE $40,$20,$FF                    ; DAMAGE REP
BF6A 484051FF 11385                .BYTE $48,$40,$51,$FF                ; MESSAGE FR
              11386
              11387 ;*** Shape of blip in Attack Computer Display ********************
BF6E 84       11388 BLIPSHAPTAB    .BYTE $84                            ; #....#..
BF6F B4       11389                .BYTE $B4                            ; #.##.#..
BF70 FC       11390                .BYTE $FC                            ; ######..
BF71 B4       11391                .BYTE $B4                            ; #.##.#..
BF72 84       11392                .BYTE $84                            ; #....#..
              11393
              11394 ;*** Initial x-coordinate (high byte) of our starship's photon tor
BF73 FF       11395 BARRELXTAB     .BYTE $FF                            ; Left barre
BF74 01       11396                .BYTE $01                            ; Right barr
              11397
              11398 ;*** Maximum photon torpedo hit z-coordinate (high byte) *********
BF75 0C       11399 HITMAXZTAB     .BYTE $0C                            ; < 3328 ($0
BF76 0C       11400                .BYTE $0C                            ; < 3328 ($0
BF77 0C       11401                .BYTE $0C                            ; < 3328 ($0
BF78 0C       11402                .BYTE $0C                            ; < 3328 ($0
BF79 0E       11403                .BYTE $0E                            ; < 3840 ($0
BF7A 0E       11404                .BYTE $0E                            ; < 3840 ($0
BF7B 0E       11405                .BYTE $0E                            ; < 3840 ($0
BF7C 20       11406                .BYTE $20                            ; < 8448 ($2
              11407
              11408 ;*** Minimum photon torpedo hit z-coordinate (high byte) *********
BF7D 00       11409 HITMINZTAB     .BYTE $00                            ; >=    0 ($
BF7E 00       11410                .BYTE $00                            ; >=    0 ($
BF7F 00       11411                .BYTE $00                            ; >=    0 ($
BF80 02       11412                .BYTE $02                            ; >=  512 ($
BF81 04       11413                .BYTE $04                            ; >= 1024 ($
BF82 06       11414                .BYTE $06                            ; >= 1536 ($
BF83 08       11415                .BYTE $08                            ; >= 2048 ($
BF84 0C       11416                .BYTE $0C                            ; >= 3072 ($
              11417
              11418 ;*** Velocity of homing Zylon photon torpedo **********************
BF85 81       11419 ZYLONHOMVELTAB .BYTE NEG!1                          ;  -1 <KM/H>
BF86 84       11420                .BYTE NEG!4                          ;  -4 <KM/H>
BF87 88       11421                .BYTE NEG!8                          ;  -8 <KM/H>
BF88 94       11422                .BYTE NEG!20                         ; -20 <KM/H>
              11423
              11424 ;*** Zylon shape type table ***************************************
BF89 80       11425 ZYLONSHAPTAB   .BYTE SHAP.ZBASESTAR                 ; ZYLON BASE
BF8A 10       11426                .BYTE SHAP.ZFIGHTER                  ; ZYLON FIGH
BF8B 10       11427                .BYTE SHAP.ZFIGHTER                  ; ZYLON FIGH
BF8C 10       11428                .BYTE SHAP.ZFIGHTER                  ; ZYLON FIGH
BF8D 70       11429                .BYTE SHAP.ZCRUISER                  ; ZYLON CRUI
BF8E 70       11430                .BYTE SHAP.ZCRUISER                  ; ZYLON CRUI
BF8F 70       11431                .BYTE SHAP.ZCRUISER                  ; ZYLON CRUI
BF90 10       11432                .BYTE SHAP.ZFIGHTER                  ; ZYLON FIGH
              11433
              11434 ;*** Zylon flight pattern table **********************************
BF91 04       11435 ZYLONFLPATTAB  .BYTE 4                              ; Flight pat
BF92 04       11436                .BYTE 4                              ; Flight pat
BF93 00       11437                .BYTE 0                              ; Attack Fli
BF94 00       11438                .BYTE 0                              ; Attack Fli
BF95 00       11439                .BYTE 0                              ; Attack Fli
BF96 01       11440                .BYTE 1                              ; Flight pat
```

```
BF97 00       11441                .BYTE 0                                ; Attack Fli
BF98 00       11442                .BYTE 0                                ; Attack Fli
              11443
              11444 ;*** Zylon velocity table ***************************************
BF99 3E       11445 ZYLONVELTAB    .BYTE 62                               ; +62 <KM/H>
BF9A 1E       11446                .BYTE 30                               ; +30 <KM/H>
BF9B 10       11447                .BYTE 16                               ; +16 <KM/H>
BF9C 08       11448                .BYTE 8                                ;  +8 <KM/H>
BF9D 04       11449                .BYTE 4                                ;  +4 <KM/H>
BF9E 02       11450                .BYTE 2                                ;  +2 <KM/H>
BF9F 01       11451                .BYTE 1                                ;  +1 <KM/H>
BFA0 00       11452                .BYTE 0                                ;   0 <KM/H>
BFA1 00       11453                .BYTE 0                                ;   0 <KM/H>
BFA2 81       11454                .BYTE NEG!1                            ;  -1 <KM/H>
BFA3 82       11455                .BYTE NEG!2                            ;  -2 <KM/H>
BFA4 84       11456                .BYTE NEG!4                            ;  -4 <KM/H>
BFA5 88       11457                .BYTE NEG!8                            ;  -8 <KM/H>
BFA6 90       11458                .BYTE NEG!16                           ; -16 <KM/H>
BFA7 9E       11459                .BYTE NEG!30                           ; -30 <KM/H>
BFA8 BE       11460                .BYTE NEG!62                           ; -62 <KM/H>
              11461
              11462 ;*** PLAYFIELD colors (including PLAYFIELD colors during DLI) ****
BFA9 A6       11463 PFCOLORTAB     .BYTE $A6                              ; PF0COLOR
BFAA AA       11464                .BYTE $AA                              ; PF1COLOR
BFAB AF       11465                .BYTE $AF                              ; PF2COLOR
BFAC 00       11466                .BYTE $00                              ; PF3COLOR
BFAD 00       11467                .BYTE $00                              ; BGRCOLOR
BFAE B8       11468                .BYTE $B8                              ; PF0COLORDL
BFAF 5A       11469                .BYTE $5A                              ; PF1COLORDL
BFB0 FC       11470                .BYTE $FC                              ; PF2COLORDL
BFB1 5E       11471                .BYTE $5E                              ; PF3COLORDL
BFB2 90       11472                .BYTE $90                              ; BGRCOLORDL
              11473
              11474 ;*** Vicinity mask table. Confines coordinates of space objects in
BFB3 FF       11475 VICINITYMASKTAB .BYTE $FF                             ; <= 65535 (
BFB4 FF       11476                .BYTE $FF                              ; <= 65535 (
BFB5 3F       11477                .BYTE $3F                              ; <= 16383 (
BFB6 0F       11478                .BYTE $0F                              ; <=  4095 (
BFB7 3F       11479                .BYTE $3F                              ; <= 16383 (
BFB8 7F       11480                .BYTE $7F                              ; <= 32767 (
BFB9 FF       11481                .BYTE $FF                              ; <= 65535 (
BFBA FF       11482                .BYTE $FF                              ; <= 65535 (
              11483
              11484 ;*** Movement probability of sector types in Galactic Chart ******
BFBB 00       11485 MOVEPROBTAB    .BYTE 0                                ; Empty sect
BFBC FF       11486                .BYTE 255                              ; 1 Zylon sh
BFBD FF       11487                .BYTE 255                              ; 2 Zylon sh
BFBE C0       11488                .BYTE 192                              ; 3 Zylon sh
BFBF 20       11489                .BYTE 32                               ; 4 Zylon sh
              11490
              11491 ;*** Galactic Chart sector offset to adjacent sector *************
BFC0 F0       11492 COMPASSOFFTAB  .BYTE -16                              ; NORTH
BFC1 EF       11493                .BYTE -17                              ; NORTHWEST
BFC2 FF       11494                .BYTE -1                               ; WEST
BFC3 0F       11495                .BYTE 15                               ; SOUTHWEST
BFC4 10       11496                .BYTE 16                               ; SOUTH
BFC5 11       11497                .BYTE 17                               ; SOUTHEAST
BFC6 01       11498                .BYTE 1                                ; EAST
BFC7 F1       11499                .BYTE -15                              ; NORTHEAST
```

```
BFC8 00       11500                .BYTE 0                                ; CENTER
              11501
              11502 ;*** Homing velocities of photon torpedoes 0..1 depending on dista
BFC9 00       11503 HOMVELTAB      .BYTE 0                                ;  +0 <KM/H>
BFCA 08       11504                .BYTE 8                                ;  +8 <KM/H>
BFCB 10       11505                .BYTE 16                               ; +16 <KM/H>
BFCC 18       11506                .BYTE 24                               ; +24 <KM/H>
BFCD 28       11507                .BYTE 40                               ; +40 <KM/H>
BFCE 30       11508                .BYTE 48                               ; +48 <KM/H>
BFCF 38       11509                .BYTE 56                               ; +56 <KM/H>
BFD0 40       11510                .BYTE 64                               ; +64 <KM/H>
              11511
              11512 ;*** PLAYER shape color table (bits B7..4 of color/brightness) ***
BFD1 50       11513 PLSHAPCOLORTAB .BYTE $50                              ; PHOTON TOR
BFD2 00       11514                .BYTE $00                              ; ZYLON FIGH
BFD3 20       11515                .BYTE $20                              ; STARBASE R
BFD4 20       11516                .BYTE $20                              ; STARBASE C
BFD5 20       11517                .BYTE $20                              ; STARBASE L
BFD6 00       11518                .BYTE $00                              ; TRANSFER V
BFD7 A0       11519                .BYTE $A0                              ; METEOR
BFD8 00       11520                .BYTE $00                              ; ZYLON CRUI
BFD9 00       11521                .BYTE $00                              ; ZYLON BASE
BFDA 9F       11522                .BYTE $9F                              ; HYPERWARP
              11523
              11524 ;*** PLAYER shape brightness table (bits B3..0 of color/brightness
BFDB 0E       11525 PLSHAPBRITTAB  .BYTE $0E                              ; ##########
BFDC 0E       11526                .BYTE $0E                              ; ##########
BFDD 0E       11527                .BYTE $0E                              ; ##########
BFDE 0C       11528                .BYTE $0C                              ; ##########
BFDF 0C       11529                .BYTE $0C                              ; ##########
BFE0 0C       11530                .BYTE $0C                              ; ##########
BFE1 0A       11531                .BYTE $0A                              ; ##########
BFE2 0A       11532                .BYTE $0A                              ; ##########
BFE3 0A       11533                .BYTE $0A                              ; ##########
BFE4 08       11534                .BYTE $08                              ; ########..
BFE5 08       11535                .BYTE $08                              ; ########..
BFE6 08       11536                .BYTE $08                              ; ########..
BFE7 06       11537                .BYTE $06                              ; ######....
BFE8 06       11538                .BYTE $06                              ; ######....
BFE9 04       11539                .BYTE $04                              ; ####......
BFEA 04       11540                .BYTE $04                              ; ####......
              11541
              11542 ;*** PHOTON TORPEDO LAUNCHED noise bit and volume (stored in rever
BFEB 8A       11543 NOISETORPVOLTAB .BYTE $8A                             ; ##########
BFEC 8F       11544                .BYTE $8F                              ; ##########
BFED 8D       11545                .BYTE $8D                              ; ##########
BFEE 8B       11546                .BYTE $8B                              ; ##########
BFEF 89       11547                .BYTE $89                              ; #########.
BFF0 87       11548                .BYTE $87                              ; #######...
BFF1 85       11549                .BYTE $85                              ; ######....
BFF2 83       11550                .BYTE $83                              ; ###.......
              11551
              11552 ;*** PHOTON TORPEDO LAUNCHED noise frequency table (stored in reve
BFF3 00       11553 NOISETORPFRQTAB .BYTE $00                             ;
BFF4 04       11554                .BYTE $04                              ;
BFF5 01       11555                .BYTE $01                              ;
BFF6 04       11556                .BYTE $04                              ;
BFF7 01       11557                .BYTE $01                              ;
BFF8 04       11558                .BYTE $04                              ;
```

```
BFF9 01       11559                .BYTE $01                          ;
BFFA 04       11560                .BYTE $04                          ;
              11561
BFFB 07       11562                .BYTE $07                          ; (unused)
              11563
BFFC 00       11564                .BYTE $00                          ; Always 0 f
BFFD 80       11565                .BYTE $80                          ; On SYSTEM
BFFE 4AA1     11566                .WORD INITCOLD                     ; Cartridge

SYMBOLS (SORTED BY NAME): 898

 A980  ABORTWARP
 A109  AFTHEADER
=0092  ARRVSECTOR
=D203  AUDC2
=D205  AUDC3
=D207  AUDC4
=D208  AUDCTL
=D200  AUDF1
=D202  AUDF2
=D204  AUDF3
=D206  AUDF4
=0087  BARRELNR
 BF73  BARRELXTAB
 B3A6  BEEP
=00D2  BEEPFRQIND
=00D7  BEEPFRQSTART
 BF5C  BEEPFRQTAB
=00D8  BEEPLIFE
 BF3E  BEEPPATTAB
=00D5  BEEPPAUSELIFE
=00D6  BEEPPRIORITY
=00D3  BEEPREPEAT
=00D9  BEEPTOGGLE
=00D4  BEEPTONELIFE
=00F6  BGRCOLOR
=00FB  BGRCOLORDLI
=00A0  BLIPCOLUMN
=00A2  BLIPCYCLECNT
=00A1  BLIPROW
 BF6E  BLIPSHAPTAB
 BEDD  BONUSTAB
 B1A7  CALCWARP
=0000  CCS.0
=0001 ?CCS.1
=0002 ?CCS.2
=001E  CCS.2ZYLONS
=0003 ?CCS.3
=001D  CCS.3ZYLONS
=0004 ?CCS.4
=001C  CCS.4ZYLONS
=0005 ?CCS.5
=0006 ?CCS.6
=0007 ?CCS.7
=0008 ?CCS.8
=0009  CCS.9
=0018  CCS.BORDERS
=000C  CCS.BORDERSW
=0019  CCS.BORDERW
```

```
=0017 ?CCS.C
=0040  CCS.COL1
=0080  CCS.COL2
=00C0  CCS.COL3
=000B  CCS.COLON
=001A  CCS.CORNERSW
=000D  CCS.E
=000E  CCS.INF
=0015  CCS.K
=000F  CCS.MINUS
=0011  CCS.PHI
=0010  CCS.PLUS
=0013  CCS.R
=000A  CCS.SPC
=001B  CCS.STARBASE
=0016  CCS.T
=0014  CCS.THETA
=0012  CCS.V
 A000  CHARSET
=D409  CHBASE
 BEFC  CLASSTAB
 A98D  CLEANUPWARP
=0074  CLOCKTIM
 AE0F  CLRMEM
 AE0D  CLRPLAYFIELD
 AF3D  COLLISION
=D016  COLPF0
=D012  COLPM0
 BFC0  COMPASSOFFTAB
=D01F  CONSOL
 ACAF  COPYPOSVEC
 ACC1  COPYPOSXY
=0076  COUNT256
=0072  COUNT8
=00A7  CTRLDZYLON
=0090  CURRSECTOR
 AEE1  DAMAGE
 BF14  DAMAGEPHRTAB
 BF10  DAMAGEPROBTAB
 B86F  DECENERGY
 BF1A  DESTROYPHRTAB
=00A4  DIRLEN
=006A  DIVIDEND
=D402  DLIST
 BA62  DLSTFRAG
 BA75  DLSTFRAGAFT
 BA7D  DLSTFRAGFRONT
 BA6A  DLSTFRAGGC
 BA6D  DLSTFRAGLRS
 BA8C  DLSTFRAGOFFTAB
 A12E  DLSTGC
 A718  DLSTHNDLR
=D400  DMACTL
 ACE6  DOCKING
=0075  DOCKSTATE
=0080  DOWN
=007E  DRAINATTCOMP
=0080  DRAINENGINES
 BAD3  DRAINRATETAB
```

```
=007D  DRAINSHIELDS
 B4B9  DRAWGC
 A782  DRAWLINE
 A784  DRAWLINE2
 A76F  DRAWLINES
 BAF9  DRAWLINESTAB
=0280  DSPLST
 A987  ENDWARP
=007F  ENERGYCNT
=0955  ENERGYD1
=0040  EOP
=0080  EOS
=0080  EOW
=0073  EXPLLIFE
=0063  FKEYCODE
 B4E4  FLUSHGAMELOOP
 BA90  FOURCOLORPIXEL
 A1F3  GAMELOOP
 B10A  GAMEOVER
 B121  GAMEOVER2
 A11A  GCHEADER
=08C9  GCMEMMAP
=0D35  GCPFMEM
=09A3  GCSTARDAT
=0995  GCSTATCOM
=0993  GCSTATENG
=0996  GCSTATLRS
=0992  GCSTATPHO
=0997  GCSTATRAD
=0994  GCSTATSHL
=098D  GCTRGCNT
=0971  GCTXT
=097D  GCWARPD1
=D01D  GRACTL
=008A  HITBADNESS
=D01E  HITCLR
 BF75  HITMAXZTAB
 BF7D  HITMINZTAB
 AECA  HOMINGVEL
 BFC9  HOMVELTAB
=D004  HPOSM0
=D005  HPOSM1
=D006  HPOSM2
=D007  HPOSM3
=D000  HPOSP0
=D001  HPOSP1
=D002  HPOSP2
=D003  HPOSP3
=0094  HUNTSECTCOLUMN
=0093  HUNTSECTOR
=0095  HUNTSECTROW
=009F  HUNTTIM
 A89B  HYPERWARP
=0066  IDLECNTHI
=0077  IDLECNTLO
 A14A  INITCOLD
 A15C  INITDEMO
 AC6B  INITEXPL
 B3BA  INITIALIZE
```

```
 B764  INITPOSVEC
 A15A  INITSELECT
 A15E  INITSTART
 A9B4  INITTRAIL
=D20E  IRQEN
 A751  IRQHNDLR
=00B8  ISBACKATTACK0
=00B9 ?ISBACKATTACK1
=0064  ISDEMOMODE
=00A3  ISINLOCKON
=007B  ISSTARBASESECT
 B7F1  ISSURROUNDED
=007C  ISTRACKCOMPON
=0086  ISTRACKING
=0067  ISVBISYNC
=006D  JOYSTICKDELTA
=00C8  JOYSTICKX
=00C9  JOYSTICKY
 A47D  JUMP001
 A4A4  JUMP002
 A579  JUMP003
 A74B  JUMP004
=D209  KBCODE
 AFFE  KEYBOARD
=00CA  KEYCODE
 BABE  KEYTAB
=0950  KILLCNTD1
=006B  L.ABSDIFFCOLUMN
=006B  L.BITPAT
=006B  L.COLORMASK
=006B  L.COLUMNPOS
=006A  L.CTRLDZYLON
=006A  L.DELTAC
=006A  L.DIRECTIONIND
=006E  L.DIRSAV
=0068  L.DIVISOR
=006A  L.FOURCOLORPIX
=006A  L.GCMEMMAPIND
=006A  L.HEIGHTCNT
=006A  L.ISDESTROYED
=006A  L.KEYCODE
=006E  L.LOOPCNT
=006B  L.LOOPCNT2
=006A  L.MAXRNDXY
=0068  L.MEMPTR1
=006A  L.MEMPTR2
=006A  L.NEWSECTOR
=006A  L.NUMBYTES
=006A  L.PIXELBYTEOFF
=006D  L.PIXELCOLUMN
=006D  L.PIXELROW
=006B  L.PLHIT
=006D  L.QUOTIENT
=0068  L.RANGE
=006A  L.RANGEINDEX
=006B  L.SECTORCNT
=006A  L.SECTORTYPE
=006C  L.SHIFTSHAP
=006A  L.SIGNCHAR
```

```
=006B  L.TERM3HI
=006A  L.TERM3LO
=006C  L.TERM3SIGN
=006C  L.TOKEN
=006E  L.TRAILCNT
=006A  L.VECCOMPIND
=006B  L.VELOCITYHI
=006A  L.VELSIGN
=006C  L.VIEWDIR
=006A  L.WARPARRVCOL
=006A  L.WORD
=006E  L.ZPOSOFF
=0088  LOCKONLIFE
=00C0  LONG
 A165  LOOP001
 A201  LOOP002
 A227  LOOP003
 A26A  LOOP004
 A277  LOOP005
 A284  LOOP006
 A291  LOOP007
 A29E  LOOP008
 A2BA  LOOP009
 A2E0  LOOP010
 A306  LOOP011
 A327  LOOP012
 A343  LOOP013
 A389  LOOP014
 A3A6  LOOP015
 A3BD  LOOP016
 A3E4  LOOP017
 A3EB  LOOP018
 A422  LOOP019
 A428  LOOP020
 A453  LOOP021
 A4C0  LOOP022
 A4E7  LOOP023
 A4FC  LOOP024
 A593  LOOP025
 A6F6  LOOP026
 A730  LOOP027
 A765  LOOP028
 A78E  LOOP029
 A7CF  LOOP030
 A83A  LOOP031
 A83C  LOOP032
 A947  LOOP033
 A9E5  LOOP034
 AA52  LOOP035
 AAB5  LOOP036
 ABB3  LOOP037
 ABCA  LOOP038
 ABFC  LOOP039
 AC73  LOOP040
 ADCA  LOOP041
 ADD7  LOOP042
 ADF4  LOOP043
 ADFB  LOOP044
 AE1A  LOOP045
```

```
 AEB3  LOOP046
 AEE7  LOOP047
 AF3F  LOOP048
 AFD5  LOOP049
 AFEC  LOOP050
 B011  LOOP051
 B056  LOOP052
 B1FE  LOOP053
 B200  LOOP054
 B234  LOOP055
 B276  LOOP056
 B286  LOOP057
 B2C1  LOOP058
 B3AF  LOOP059
 B3BC  LOOP060
 B3EE  LOOP061
 B41B  LOOP062
 B441  LOOP063
 B44C  LOOP064
 B488  LOOP065
 B492  LOOP066
 B4BD  LOOP067
 B51C  LOOP068
 B54E  LOOP069
 B57C  LOOP070
 B5C1  LOOP071
 B5D1  LOOP072
 B5EA  LOOP073
 B5EF  LOOP074
 B601  LOOP075
 B632  LOOP076
 B662  LOOP077
 B664  LOOP078
 B896  LOOP079
 A0F8  LRSHEADER
=D008  M0PL
=D009  M1PL
=D00A  M2PL
=D00B  M3PL
 AA79  MANEUVER
=0DE9  MAPTO80
=0EE9  MAPTOBCD99
=0079  MAXSPCOBJIND
=0068  MEMPTR
=00AA  MILESTTIM0
=00AB ?MILESTTIM1
=00AE ?MILESTVELINDX0
=00AF ?MILESTVELINDX1
=00B0 ?MILESTVELINDY0
=00B1 ?MILESTVELINDY1
=00AC  MILESTVELINDZ0
=00AD ?MILESTVELINDZ1
=0062  MISSIONLEVEL
 BF0C  MISSIONPHRTAB
 ADF1  MODDLST
 BFBB  MOVEPROBTAB
 BE29  MSGBITTAB
 BE26  MSGOFFTAB
 BE2C  MSGONTAB
```

```
=0080  NEG
=0065  NEWTITLEPHR
=0071  NEWVELOCITY
=0096  NEWZYLONDIST
=D40E  NMIEN
 AEA8  NOISE
=00DC  NOISEAUDC2
=00DD  NOISEAUDC3
=00DE  NOISEAUDF1
=00DF  NOISEAUDF2
=00DB  NOISEEXPLTIM
=00E0  NOISEFRQINC
=00E3  NOISEHITLIFE
=00E1  NOISELIFE
 BF20  NOISEPATTAB
 BFF3  NOISETORPFRQTAB
=00DA  NOISETORPTIM
 BFEB  NOISETORPVOLTAB
=00E2  NOISEZYLONTIM
=0031  NUMSPCOBJ.ALL
=0011  NUMSPCOBJ.NORM
=0005  NUMSPCOBJ.PL
=000C  NUMSPCOBJ.STARS
=007A  OLDMAXSPCOBJIND
=0084  OLDTRIG0
=009E  OLDZYLONDIST
=D00F  P3PL
=D302  PACTL
=0949  PANELTXT
 BB42  PANELTXTTAB
=00A6  PENCOLUMN
=00A5  PENROW
=00F2  PF0COLOR
=00F7  PF0COLORDLI
=00F3 ?PF1COLOR
=00F8 ?PF1COLORDLI
=00F4  PF2COLOR
=00F9 ?PF2COLORDLI
=00F5 ?PF3COLOR
=00FA ?PF3COLORDLI
 BFA9  PFCOLORTAB
=1000  PFMEM
=1000  PFMEM.C0R0
=12A8  PFMEM.C0R17
=10C8  PFMEM.C0R5
=1B36  PFMEM.C120R71
=1BFE  PFMEM.C120R76
=1C9E  PFMEM.C120R80
=1D40  PFMEM.C128R84
=1D68  PFMEM.C128R85
=1D42  PFMEM.C136R84
=1D6A  PFMEM.C136R85
=1C04  PFMEM.C144R76
=1CA4  PFMEM.C144R80
=17BB  PFMEM.C76R49
=17E3  PFMEM.C76R50
=17BC  PFMEM.C80R49
=17E4  PFMEM.C80R50
=0864  PFMEMROWHI
```

```
=0800  PFMEMROWLO
=0966  PHIC1
 BBAA  PHRASETAB
=0CEE  PIXELBYTE
=0C8C  PIXELBYTEOFF
=0C2A  PIXELCOLUMN
 BAB0  PIXELMASKTAB
=0C5B  PIXELROW
=0BF9  PIXELROWNEW
=0CBD  PIXELSAVE
=00EE  PL0COLOR
=0C2A  PL0COLUMN
=0400  PL0DATA
=0CBD  PL0HEIGHT
=0CEE  PL0HEIGHTNEW
=00E9  PL0LIFE
=0C5B  PL0ROW
=0BF9  PL0ROWNEW
=00E4  PL0SHAPOFF
=0C8C  PL0SHAPTYPE
=0B97 ?PL0XVEL
=0BC8 ?PL0YVEL
=0A40  PL0ZPOSHI
=0B66 ?PL0ZVEL
=00EF ?PL1COLOR
=0C2B  PL1COLUMN
=0500  PL1DATA
=0CBE  PL1HEIGHT
=0CEF  PL1HEIGHTNEW
=00EA  PL1LIFE
=0C5C  PL1ROW
=0BFA  PL1ROWNEW
=00E5  PL1SHAPOFF
=0C8D  PL1SHAPTYPE
=0B98 ?PL1XVEL
=0BC9 ?PL1YVEL
=0B67 ?PL1ZVEL
=00F0 ?PL2COLOR
=0C2C  PL2COLUMN
=0600  PL2DATA
=0CBF  PL2HEIGHT
=0CF0  PL2HEIGHTNEW
=00EB  PL2LIFE
=0C5D  PL2ROW
=0BFB  PL2ROWNEW
=00E6  PL2SHAPOFF
=0C8E  PL2SHAPTYPE
=0A73  PL2XPOSHI
=0B06 ?PL2XPOSLO
=09E0 ?PL2XPOSSIGN
=0B99  PL2XVEL
=0AA4  PL2YPOSHI
=0B37 ?PL2YPOSLO
=0A11  PL2YPOSSIGN
=0BCA  PL2YVEL
=0A42  PL2ZPOSHI
=0AD5  PL2ZPOSLO
=09AF  PL2ZPOSSIGN
=0B68  PL2ZVEL
```

```
=00F1 ?PL3COLOR
=0C2D  PL3COLUMN
=0700  PL3DATA
=0CC0  PL3HEIGHT
=0CF1  PL3HEIGHTNEW
=0082  PL3HIT
=00EC  PL3LIFE
=0C5E  PL3ROW
=0BFC  PL3ROWNEW
=00E7  PL3SHAPOFF
=0C8F  PL3SHAPTYPE
=0A74  PL3XPOSHI
=0B07  PL3XPOSLO
=09E1  PL3XPOSSIGN
=0B9A  PL3XVEL
=0AA5  PL3YPOSHI
=0B38  PL3YPOSLO
=0A12  PL3YPOSSIGN
=0BCB  PL3YVEL
=0A43  PL3ZPOSHI
=0AD6  PL3ZPOSLO
=09B0  PL3ZPOSSIGN
=0B69  PL3ZVEL
=0C2E  PL4COLUMN
=0300  PL4DATA
=0CC1  PL4HEIGHT
=0CF2  PL4HEIGHTNEW
=0083  PL4HIT
=00ED  PL4LIFE
=0C5F  PL4ROW
=0BFD  PL4ROWNEW
=00E8  PL4SHAPOFF
=0C90  PL4SHAPTYPE
=0A75  PL4XPOSHI
=0B08 ?PL4XPOSLO
=09E2  PL4XPOSSIGN
=0B9B  PL4XVEL
=0AA6  PL4YPOSHI
=0B39 ?PL4YPOSLO
=0A13  PL4YPOSSIGN
=0BCC  PL4YVEL
=0A44  PL4ZPOSHI
=0AD7 ?PL4ZPOSLO
=09B1  PL4ZPOSSIGN
=0B6A  PL4ZVEL
 B8DF  PLCOLOROFFTAB
 B8E4  PLSHAP1TAB
 B9B1  PLSHAP2TAB
 BFDB  PLSHAPBRITTAB
 BFD1  PLSHAPCOLORTAB
 BE7F  PLSHAPHEIGHTTAB
 BE2F  PLSHAPOFFTAB
 BEDB  PLSTARBAOFFTAB
=0089  PLTRACKED
=D407  PMBASE
=D300  PORTA
=D01B  PRIOR
 AA21  PROJECTION
=D20A  RANDOM
```

```
=096C  RANGEC1
 BEE9  RANKTAB
=008B  REDALERTLIFE
=0000  RIGHT
 B7BE  RNDINVXY
=0010  ROM.0
=0011  ROM.1
=0012  ROM.2
=0013  ROM.3
=0014  ROM.4
=0015  ROM.5
=0019  ROM.9
=0021  ROM.A
=0023  ROM.C
=001A  ROM.COLON
=0024  ROM.D
=000E  ROM.DOT
=0025  ROM.E
=0027  ROM.G
=002C  ROM.L
=002E  ROM.N
=0030  ROM.P
=0032  ROM.R
=0033  ROM.S
=0000  ROM.SPC
=0034  ROM.T
=0037  ROM.W
=0039  ROM.Y
=E000  ROMCHARSET
 B69B  ROTATE
=00CB  SCORE
=00CE  SCOREDCLASSIND
=00CD  SCOREDRANKIND
 B6FB  SCREENCOLUMN
 B71E  SCREENROW
 BED1  SECTORCHARTAB
 BBA6  SECTORTYPETAB
 B162  SELECTWARP
 B223  SETTITLE
 B045  SETVIEW
=0090  SHAP.HYPERWARP
=0060  SHAP.METEOR
=0030  SHAP.STARBASEC
=0020  SHAP.STARBASEL
=0040  SHAP.STARBASER
=0000 ?SHAP.TORPEDO
=0050  SHAP.TRANSVSSL
=0080  SHAP.ZBASESTAR
=0070  SHAP.ZCRUISER
=0010  SHAP.ZFIGHTER
=0081  SHIELDSCOLOR
=00D0  SHIPVIEW
 B8A7  SHOWCOORD
 B8CD  SHOWDIGITS
=D20F  SKCTL
 A172  SKIP001
 A21F  SKIP002
 A250  SKIP003
 A262  SKIP004
```

A2C2 SKIP005
A2E8 SKIP006
A30E SKIP007
A39E SKIP008
A3BB SKIP009
A3C6 SKIP010
A3DF SKIP011
A3EA SKIP012
A3FE SKIP013
A43C SKIP014
A43F SKIP015
A473 SKIP016
A49A SKIP017
A4A7 SKIP018
A4AD SKIP019
A4CA SKIP020
A4DB SKIP021
A4E5 SKIP022
A4ED SKIP023
A503 SKIP024
A52A SKIP025
A52E SKIP026
A53E SKIP027
A548 SKIP028
A569 SKIP029
A58D SKIP030
A5A3 SKIP031
A5A5 SKIP032
A5AB SKIP033
A5D0 SKIP034
A600 SKIP035
A60C SKIP036
A61B SKIP037
A635 SKIP038
A687 SKIP039
A69B SKIP040
A6B7 SKIP041
A6C2 SKIP042
A6E9 SKIP043
A6EA SKIP044
A6F2 SKIP045
A715 SKIP046
A728 SKIP047
A77A SKIP048
A781 SKIP049
A7B8 SKIP050
A7BA SKIP051
A7E1 SKIP052
A7E9 SKIP053
A7EC SKIP054
A804 SKIP055
A80A SKIP056
A821 SKIP057
A827 SKIP058
A830 SKIP059
A850 SKIP060
A85F SKIP061
A898 SKIP062
A8AC SKIP063

A8E8 SKIP064
A8EC SKIP065
A900 SKIP066
A901 SKIP067
A915 SKIP068
A91E SKIP069
A96F SKIP070
A97F SKIP071
A9A6 SKIP072
AA1A SKIP073
AA20 SKIP074
AA40 SKIP075
AA66 SKIP076
AA6F SKIP077
AA78 SKIP078
AA90 SKIP079
AAB3 SKIP080
AAC8 SKIP081
AACF SKIP082
AAD5 SKIP083
AADD SKIP084
AAE0 SKIP085
AAF4 SKIP086
AB00 SKIP087
AB03 SKIP088
AB09 SKIP089
AB11 SKIP090
AB36 SKIP091
AB37 SKIP092
AB66 SKIP093
AB84 SKIP094
AB98 SKIP095
AB9C SKIP096
ABAE SKIP097
ABBA SKIP098
ABC4 SKIP099
ABDD SKIP100
ABE1 SKIP101
ABE5 SKIP102
ABE9 SKIP103
ABEB SKIP104
ABFA SKIP105
AC08 SKIP106
AC0A SKIP107
AC31 SKIP108
AC32 SKIP109
AC4F SKIP110
ACE5 SKIP111
ACF3 SKIP112
AD12 SKIP113
AD26 SKIP114
AD35 SKIP115
AD61 SKIP116
AD6C SKIP117
AD70 SKIP118
AD71 SKIP119
AD82 SKIP120
ADB8 SKIP121
ADB9 SKIP122

AE03 SKIP123
AE40 SKIP124
AE41 SKIP125
AE56 SKIP126
AE58 SKIP127
AE66 SKIP128
AEB1 SKIP129
AEC9 SKIP130
AED2 SKIP131
AEDA SKIP132
AF10 SKIP133
AF19 SKIP134
AF1E SKIP135
AF32 SKIP136
AF3C SKIP137
AF43 SKIP138
AF58 SKIP139
AF64 SKIP140
AF6F SKIP141
AF94 SKIP142
AFC6 SKIP143
AFE7 SKIP144
AFF3 SKIP145
AFFD SKIP146
B020 SKIP147
B02B SKIP148
B036 SKIP149
B040 SKIP150
B041 SKIP151
B060 SKIP152
B073 SKIP153
B082 SKIP154
B096 SKIP155
B099 SKIP156
B0E6 SKIP157
B0ED SKIP158
B0FB SKIP159
B0FC SKIP160
B106 SKIP161
B14A SKIP162
B15A SKIP163
B15D SKIP164
B161 SKIP165
B16A SKIP166
B16B SKIP167
B173 SKIP168
B1BE SKIP169
B1C8 SKIP170
B1D3 SKIP171
B1E0 SKIP172
B212 SKIP173
B21E SKIP174
B21F SKIP175
B22E SKIP176
B23A SKIP177
B249 SKIP178
B25F SKIP179
B268 SKIP180
B27C SKIP181

B2A2 SKIP182
B2A8 SKIP183
B2E1 SKIP184
B2E6 SKIP185
B2F3 SKIP186
B32B SKIP187
B337 SKIP188
B349 SKIP189
B357 SKIP190
B369 SKIP191
B397 SKIP192
B39F SKIP193
B3B9 SKIP194
B3CA SKIP195
B47C SKIP196
B4C6 SKIP197
B4F5 SKIP198
B50F SKIP199
B511 SKIP200
B516 SKIP201
B51A SKIP202
B527 SKIP203
B536 SKIP204
B53E SKIP205
B544 SKIP206
B562 SKIP207
B565 SKIP208
B569 SKIP209
B56A SKIP210
B574 SKIP211
B59A SKIP212
B5B0 SKIP213
B5BB SKIP214
B5DA SKIP215
B619 SKIP216
B61C SKIP217
B61D SKIP218
B644 SKIP219
B655 SKIP220
B68D SKIP221
B68F SKIP222
B698 SKIP223
B6A4 SKIP224
B6E1 SKIP225
B709 SKIP226
B717 SKIP227
B72E SKIP228
B73E SKIP229
B745 SKIP230
B74A SKIP231
B753 SKIP232
B75A SKIP233
B763 SKIP234
B785 SKIP235
B7A9 SKIP236
B7D7 SKIP237
B7F0 SKIP238
B803 SKIP239
B810 SKIP240

```
 B812  SKIP241
 B822  SKIP242
 B85C  SKIP243
 B88C  SKIP244
 B88E  SKIP245
 B8A6  SKIP246
 B8BD  SKIP247
 B2AB  SOUND
 BAF5  STICKINCTAB
=D209  STIMER
=0960  THETAC1
=00CF  TITLELIFE
=00D1  TITLEPHR
=0D1F  TITLETXT
=00BE  TORPEDODELAY
=095A  TRACKC1
=095C  TRACKDIGIT
 BECF  TRACKKEYSTAB
=00C2  TRAILDELAY
=00C3  TRAILIND
=D010  TRIG0
 AE29  TRIGGER
 A7BF  UPDATTCOMP
 B804  UPDPANEL
 B07B  UPDSCREEN
 B216  UPDTITLE
 A6D1  VBIHNDLR
=D40B  VCOUNT
=0200  VDSLST
=00C6  VEERMASK
 BED7  VEERMASKTAB
=094B  VELOCD1
=00C1  VELOCITYHI
=0070  VELOCITYLO
 BAB4  VELOCITYTAB
=00C7  VICINITYMASK
 BFB3  VICINITYMASKTAB
 BE22  VIEWMODETAB
=0216  VIMIRQ
=0222  VVBLKI
=008F  WARPARRVCOLUMN
=008E  WARPARRVROW
=008D  WARPDEPRCOLUMN
=008C  WARPDEPRROW
=0091  WARPENERGY
 BADD  WARPENERGYTAB
 BB3A  WARPSTARXTAB
 BB3E  WARPSTARYTAB
=00C0  WARPSTATE
=00C4  WARPTEMPCOLUMN
=00C5  WARPTEMPROW
 BC2B  WORDTAB
=D40A  WSYNC
=0A71  XPOSHI
=0B04  XPOSLO
=09DE  XPOSSIGN
=0B97  XVEL
=0AA2  YPOSHI
=0B35  YPOSLO
```

```
=0A0F  YPOSSIGN
=0BC8  YVEL
=0A40  ZPOSHI
=0AD3  ZPOSLO
=09AD  ZPOSSIGN
=0B66  ZVEL
=00BF  ZYLONATTACKER
=00A8  ZYLONFLPAT0
=00A9 ?ZYLONFLPAT1
 BF91  ZYLONFLPATTAB
 BF85  ZYLONHOMVELTAB
 BF89  ZYLONSHAPTAB
=00BA  ZYLONTIMX0
=00BB ?ZYLONTIMX1
=00BC ?ZYLONTIMY0
=00BD ?ZYLONTIMY1
=0078  ZYLONUNITTIM
=00B4  ZYLONVELINDX0
=00B5 ?ZYLONVELINDX1
=00B6 ?ZYLONVELINDY0
=00B7 ?ZYLONVELINDY1
=00B2  ZYLONVELINDZ0
=00B3 ?ZYLONVELINDZ1
 BF99  ZYLONVELTAB

SYMBOLS (SORTED BY VALUE): 898

=0000  CCS.0
=0000  RIGHT
=0000  ROM.SPC
=0000 ?SHAP.TORPEDO
=0001 ?CCS.1
=0002 ?CCS.2
=0003 ?CCS.3
=0004 ?CCS.4
=0005 ?CCS.5
=0005  NUMSPCOBJ.PL
=0006 ?CCS.6
=0007 ?CCS.7
=0008 ?CCS.8
=0009  CCS.9
=000A  CCS.SPC
=000B  CCS.COLON
=000C  CCS.BORDERSW
=000C  NUMSPCOBJ.STARS
=000D  CCS.E
=000E  CCS.INF
=000E  ROM.DOT
=000F  CCS.MINUS
=0010  CCS.PLUS
=0010  ROM.0
=0010  SHAP.ZFIGHTER
=0011  CCS.PHI
=0011  NUMSPCOBJ.NORM
=0011  ROM.1
=0012  CCS.V
=0012  ROM.2
=0013  CCS.R
=0013  ROM.3
```

```
=0014  CCS.THETA
=0014  ROM.4
=0015  CCS.K
=0015  ROM.5
=0016  CCS.T
=0017 ?CCS.C
=0018  CCS.BORDERS
=0019  CCS.BORDERW
=0019  ROM.9
=001A  CCS.CORNERSW
=001A  ROM.COLON
=001B  CCS.STARBASE
=001C  CCS.4ZYLONS
=001D  CCS.3ZYLONS
=001E  CCS.2ZYLONS
=0020  SHAP.STARBASEL
=0021  ROM.A
=0023  ROM.C
=0024  ROM.D
=0025  ROM.E
=0027  ROM.G
=002C  ROM.L
=002E  ROM.N
=0030  ROM.P
=0030  SHAP.STARBASEC
=0031  NUMSPCOBJ.ALL
=0032  ROM.R
=0033  ROM.S
=0034  ROM.T
=0037  ROM.W
=0039  ROM.Y
=0040  CCS.COL1
=0040  EOP
=0040  SHAP.STARBASER
=0050  SHAP.TRANSVSSL
=0060  SHAP.METEOR
=0062  MISSIONLEVEL
=0063  FKEYCODE
=0064  ISDEMOMODE
=0065  NEWTITLEPHR
=0066  IDLECNTHI
=0067  ISVBISYNC
=0068  L.DIVISOR
=0068  L.MEMPTR1
=0068  L.RANGE
=0068  MEMPTR
=006A  DIVIDEND
=006A  L.CTRLDZYLON
=006A  L.DELTAC
=006A  L.DIRECTIONIND
=006A  L.FOURCOLORPIX
=006A  L.GCMEMMAPIND
=006A  L.HEIGHTCNT
=006A  L.ISDESTROYED
=006A  L.KEYCODE
=006A  L.MAXRNDXY
=006A  L.MEMPTR2
=006A  L.NEWSECTOR
=006A  L.NUMBYTES
```

```
=006A  L.PIXELBYTEOFF
=006A  L.RANGEINDEX
=006A  L.SECTORTYPE
=006A  L.SIGNCHAR
=006A  L.TERM3LO
=006A  L.VECCOMPIND
=006A  L.VELSIGN
=006A  L.WARPARRVCOL
=006A  L.WORD
=006B  L.ABSDIFFCOLUMN
=006B  L.BITPAT
=006B  L.COLORMASK
=006B  L.COLUMNPOS
=006B  L.LOOPCNT2
=006B  L.PLHIT
=006B  L.SECTORCNT
=006B  L.TERM3HI
=006B  L.VELOCITYHI
=006C  L.SHIFTSHAP
=006C  L.TERM3SIGN
=006C  L.TOKEN
=006C  L.VIEWDIR
=006D  JOYSTICKDELTA
=006D  L.PIXELCOLUMN
=006D  L.PIXELROW
=006D  L.QUOTIENT
=006E  L.DIRSAV
=006E  L.LOOPCNT
=006E  L.TRAILCNT
=006E  L.ZPOSOFF
=0070  SHAP.ZCRUISER
=0070  VELOCITYLO
=0071  NEWVELOCITY
=0072  COUNT8
=0073  EXPLLIFE
=0074  CLOCKTIM
=0075  DOCKSTATE
=0076  COUNT256
=0077  IDLECNTLO
=0078  ZYLONUNITTIM
=0079  MAXSPCOBJIND
=007A  OLDMAXSPCOBJIND
=007B  ISSTARBASESECT
=007C  ISTRACKCOMPON
=007D  DRAINSHIELDS
=007E  DRAINATTCOMP
=007F  ENERGYCNT
=0080  CCS.COL2
=0080  DOWN
=0080  DRAINENGINES
=0080  EOS
=0080  EOW
=0080  NEG
=0080  SHAP.ZBASESTAR
=0081  SHIELDSCOLOR
=0082  PL3HIT
=0083  PL4HIT
=0084  OLDTRIG0
=0086  ISTRACKING
```

```
=0087  BARRELNR
=0088  LOCKONLIFE
=0089  PLTRACKED
=008A  HITBADNESS
=008B  REDALERTLIFE
=008C  WARPDEPRROW
=008D  WARPDEPRCOLUMN
=008E  WARPARRVROW
=008F  WARPARRVCOLUMN
=0090  CURRSECTOR
=0090  SHAP.HYPERWARP
=0091  WARPENERGY
=0092  ARRVSECTOR
=0093  HUNTSECTOR
=0094  HUNTSECTCOLUMN
=0095  HUNTSECTROW
=0096  NEWZYLONDIST
=009E  OLDZYLONDIST
=009F  HUNTTIM
=00A0  BLIPCOLUMN
=00A1  BLIPROW
=00A2  BLIPCYCLECNT
=00A3  ISINLOCKON
=00A4  DIRLEN
=00A5  PENROW
=00A6  PENCOLUMN
=00A7  CTRLDZYLON
=00A8  ZYLONFLPAT0
=00A9 ?ZYLONFLPAT1
=00AA  MILESTTIM0
=00AB ?MILESTTIM1
=00AC  MILESTVELINDZ0
=00AD ?MILESTVELINDZ1
=00AE ?MILESTVELINDX0
=00AF ?MILESTVELINDX1
=00B0 ?MILESTVELINDY0
=00B1 ?MILESTVELINDY1
=00B2  ZYLONVELINDZ0
=00B3 ?ZYLONVELINDZ1
=00B4  ZYLONVELINDX0
=00B5 ?ZYLONVELINDX1
=00B6 ?ZYLONVELINDY0
=00B7 ?ZYLONVELINDY1
=00B8  ISBACKATTACK0
=00B9 ?ISBACKATTACK1
=00BA  ZYLONTIMX0
=00BB ?ZYLONTIMX1
=00BC ?ZYLONTIMY0
=00BD ?ZYLONTIMY1
=00BE  TORPEDODELAY
=00BF  ZYLONATTACKER
=00C0  CCS.COL3
=00C0  LONG
=00C0  WARPSTATE
=00C1  VELOCITYHI
=00C2  TRAILDELAY
=00C3  TRAILIND
=00C4  WARPTEMPCOLUMN
=00C5  WARPTEMPROW
```

```
=00C6  VEERMASK
=00C7  VICINITYMASK
=00C8  JOYSTICKX
=00C9  JOYSTICKY
=00CA  KEYCODE
=00CB  SCORE
=00CD  SCOREDRANKIND
=00CE  SCOREDCLASSIND
=00CF  TITLELIFE
=00D0  SHIPVIEW
=00D1  TITLEPHR
=00D2  BEEPFRQIND
=00D3  BEEPREPEAT
=00D4  BEEPTONELIFE
=00D5  BEEPPAUSELIFE
=00D6  BEEPPRIORITY
=00D7  BEEPFRQSTART
=00D8  BEEPLIFE
=00D9  BEEPTOGGLE
=00DA  NOISETORPTIM
=00DB  NOISEEXPLTIM
=00DC  NOISEAUDC2
=00DD  NOISEAUDC3
=00DE  NOISEAUDF1
=00DF  NOISEAUDF2
=00E0  NOISEFRQINC
=00E1  NOISELIFE
=00E2  NOISEZYLONTIM
=00E3  NOISEHITLIFE
=00E4  PL0SHAPOFF
=00E5  PL1SHAPOFF
=00E6  PL2SHAPOFF
=00E7  PL3SHAPOFF
=00E8  PL4SHAPOFF
=00E9  PL0LIFE
=00EA  PL1LIFE
=00EB  PL2LIFE
=00EC  PL3LIFE
=00ED  PL4LIFE
=00EE  PL0COLOR
=00EF ?PL1COLOR
=00F0 ?PL2COLOR
=00F1 ?PL3COLOR
=00F2  PF0COLOR
=00F3 ?PF1COLOR
=00F4  PF2COLOR
=00F5 ?PF3COLOR
=00F6  BGRCOLOR
=00F7  PF0COLORDLI
=00F8 ?PF1COLORDLI
=00F9 ?PF2COLORDLI
=00FA ?PF3COLORDLI
=00FB  BGRCOLORDLI
=0200  VDSLST
=0216  VIMIRQ
=0222  VVBLKI
=0280  DSPLST
=0300  PL4DATA
=0400  PL0DATA
```

```
=0500  PL1DATA
=0600  PL2DATA
=0700  PL3DATA
=0800  PFMEMROWLO
=0864  PFMEMROWHI
=08C9  GCMEMMAP
=0949  PANELTXT
=094B  VELOCD1
=0950  KILLCNTD1
=0955  ENERGYD1
=095A  TRACKC1
=095C  TRACKDIGIT
=0960  THETAC1
=0966  PHIC1
=096C  RANGEC1
=0971  GCTXT
=097D  GCWARPD1
=098D  GCTRGCNT
=0992  GCSTATPHO
=0993  GCSTATENG
=0994  GCSTATSHL
=0995  GCSTATCOM
=0996  GCSTATLRS
=0997  GCSTATRAD
=09A3  GCSTARDAT
=09AD  ZPOSSIGN
=09AF  PL2ZPOSSIGN
=09B0  PL3ZPOSSIGN
=09B1  PL4ZPOSSIGN
=09DE  XPOSSIGN
=09E0 ?PL2XPOSSIGN
=09E1  PL3XPOSSIGN
=09E2  PL4XPOSSIGN
=0A0F  YPOSSIGN
=0A11  PL2YPOSSIGN
=0A12  PL3YPOSSIGN
=0A13  PL4YPOSSIGN
=0A40  PL0ZPOSHI
=0A40  ZPOSHI
=0A42  PL2ZPOSHI
=0A43  PL3ZPOSHI
=0A44  PL4ZPOSHI
=0A71  XPOSHI
=0A73  PL2XPOSHI
=0A74  PL3XPOSHI
=0A75  PL4XPOSHI
=0AA2  YPOSHI
=0AA4  PL2YPOSHI
=0AA5  PL3YPOSHI
=0AA6  PL4YPOSHI
=0AD3  ZPOSLO
=0AD5  PL2ZPOSLO
=0AD6  PL3ZPOSLO
=0AD7 ?PL4ZPOSLO
=0B04  XPOSLO
=0B06 ?PL2XPOSLO
=0B07  PL3XPOSLO
=0B08 ?PL4XPOSLO
=0B35  YPOSLO
```

```
=0B37 ?PL2YPOSLO
=0B38  PL3YPOSLO
=0B39 ?PL4YPOSLO
=0B66 ?PL0ZVEL
=0B66  ZVEL
=0B67 ?PL1ZVEL
=0B68  PL2ZVEL
=0B69  PL3ZVEL
=0B6A  PL4ZVEL
=0B97 ?PL0XVEL
=0B97  XVEL
=0B98 ?PL1XVEL
=0B99  PL2XVEL
=0B9A  PL3XVEL
=0B9B  PL4XVEL
=0BC8 ?PL0YVEL
=0BC8  YVEL
=0BC9 ?PL1YVEL
=0BCA  PL2YVEL
=0BCB  PL3YVEL
=0BCC  PL4YVEL
=0BF9  PIXELROWNEW
=0BF9  PL0ROWNEW
=0BFA  PL1ROWNEW
=0BFB  PL2ROWNEW
=0BFC  PL3ROWNEW
=0BFD  PL4ROWNEW
=0C2A  PIXELCOLUMN
=0C2A  PL0COLUMN
=0C2B  PL1COLUMN
=0C2C  PL2COLUMN
=0C2D  PL3COLUMN
=0C2E  PL4COLUMN
=0C5B  PIXELROW
=0C5B  PL0ROW
=0C5C  PL1ROW
=0C5D  PL2ROW
=0C5E  PL3ROW
=0C5F  PL4ROW
=0C8C  PIXELBYTEOFF
=0C8C  PL0SHAPTYPE
=0C8D  PL1SHAPTYPE
=0C8E  PL2SHAPTYPE
=0C8F  PL3SHAPTYPE
=0C90  PL4SHAPTYPE
=0CBD  PIXELSAVE
=0CBD  PL0HEIGHT
=0CBE  PL1HEIGHT
=0CBF  PL2HEIGHT
=0CC0  PL3HEIGHT
=0CC1  PL4HEIGHT
=0CEE  PIXELBYTE
=0CEE  PL0HEIGHTNEW
=0CEF  PL1HEIGHTNEW
=0CF0  PL2HEIGHTNEW
=0CF1  PL3HEIGHTNEW
=0CF2  PL4HEIGHTNEW
=0D1F  TITLETXT
=0D35  GCPFMEM
```

```
=0DE9  MAPTO80
=0EE9  MAPTOBCD99
=1000  PFMEM
=1000  PFMEM.C0R0
=10C8  PFMEM.C0R5
=12A8  PFMEM.C0R17
=17BB  PFMEM.C76R49
=17BC  PFMEM.C80R49
=17E3  PFMEM.C76R50
=17E4  PFMEM.C80R50
=1B36  PFMEM.C120R71
=1BFE  PFMEM.C120R76
=1C04  PFMEM.C144R76
=1C9E  PFMEM.C120R80
=1CA4  PFMEM.C144R80
=1D40  PFMEM.C128R84
=1D42  PFMEM.C136R84
=1D68  PFMEM.C128R85
=1D6A  PFMEM.C136R85
 A000  CHARSET
 A0F8  LRSHEADER
 A109  AFTHEADER
 A11A  GCHEADER
 A12E  DLSTGC
 A14A  INITCOLD
 A15A  INITSELECT
 A15C  INITDEMO
 A15E  INITSTART
 A165  LOOP001
 A172  SKIP001
 A1F3  GAMELOOP
 A201  LOOP002
 A21F  SKIP002
 A227  LOOP003
 A250  SKIP003
 A262  SKIP004
 A26A  LOOP004
 A277  LOOP005
 A284  LOOP006
 A291  LOOP007
 A29E  LOOP008
 A2BA  LOOP009
 A2C2  SKIP005
 A2E0  LOOP010
 A2E8  SKIP006
 A306  LOOP011
 A30E  SKIP007
 A327  LOOP012
 A343  LOOP013
 A389  LOOP014
 A39E  SKIP008
 A3A6  LOOP015
 A3BB  SKIP009
 A3BD  LOOP016
 A3C6  SKIP010
 A3DF  SKIP011
 A3E4  LOOP017
 A3EA  SKIP012
 A3EB  LOOP018
```

A3FE SKIP013
A422 LOOP019
A428 LOOP020
A43C SKIP014
A43F SKIP015
A453 LOOP021
A473 SKIP016
A47D JUMP001
A49A SKIP017
A4A4 JUMP002
A4A7 SKIP018
A4AD SKIP019
A4C0 LOOP022
A4CA SKIP020
A4DB SKIP021
A4E5 SKIP022
A4E7 LOOP023
A4ED SKIP023
A4FC LOOP024
A503 SKIP024
A52A SKIP025
A52E SKIP026
A53E SKIP027
A548 SKIP028
A569 SKIP029
A579 JUMP003
A58D SKIP030
A593 LOOP025
A5A3 SKIP031
A5A5 SKIP032
A5AB SKIP033
A5D0 SKIP034
A600 SKIP035
A60C SKIP036
A61B SKIP037
A635 SKIP038
A687 SKIP039
A69B SKIP040
A6B7 SKIP041
A6C2 SKIP042
A6D1 VBIHNDLR
A6E9 SKIP043
A6EA SKIP044
A6F2 SKIP045
A6F6 LOOP026
A715 SKIP046
A718 DLSTHNDLR
A728 SKIP047
A730 LOOP027
A74B JUMP004
A751 IRQHNDLR
A765 LOOP028
A76F DRAWLINES
A77A SKIP048
A781 SKIP049
A782 DRAWLINE
A784 DRAWLINE2
A78E LOOP029
A7B8 SKIP050

```
A7BA  SKIP051
A7BF  UPDATTCOMP
A7CF  LOOP030
A7E1  SKIP052
A7E9  SKIP053
A7EC  SKIP054
A804  SKIP055
A80A  SKIP056
A821  SKIP057
A827  SKIP058
A830  SKIP059
A83A  LOOP031
A83C  LOOP032
A850  SKIP060
A85F  SKIP061
A898  SKIP062
A89B  HYPERWARP
A8AC  SKIP063
A8E8  SKIP064
A8EC  SKIP065
A900  SKIP066
A901  SKIP067
A915  SKIP068
A91E  SKIP069
A947  LOOP033
A96F  SKIP070
A97F  SKIP071
A980  ABORTWARP
A987  ENDWARP
A98D  CLEANUPWARP
A9A6  SKIP072
A9B4  INITTRAIL
A9E5  LOOP034
AA1A  SKIP073
AA20  SKIP074
AA21  PROJECTION
AA40  SKIP075
AA52  LOOP035
AA66  SKIP076
AA6F  SKIP077
AA78  SKIP078
AA79  MANEUVER
AA90  SKIP079
AAB3  SKIP080
AAB5  LOOP036
AAC8  SKIP081
AACF  SKIP082
AAD5  SKIP083
AADD  SKIP084
AAE0  SKIP085
AAF4  SKIP086
AB00  SKIP087
AB03  SKIP088
AB09  SKIP089
AB11  SKIP090
AB36  SKIP091
AB37  SKIP092
AB66  SKIP093
AB84  SKIP094
```

AB98 SKIP095
AB9C SKIP096
ABAE SKIP097
ABB3 LOOP037
ABBA SKIP098
ABC4 SKIP099
ABCA LOOP038
ABDD SKIP100
ABE1 SKIP101
ABE5 SKIP102
ABE9 SKIP103
ABEB SKIP104
ABFA SKIP105
ABFC LOOP039
AC08 SKIP106
AC0A SKIP107
AC31 SKIP108
AC32 SKIP109
AC4F SKIP110
AC6B INITEXPL
AC73 LOOP040
ACAF COPYPOSVEC
ACC1 COPYPOSXY
ACE5 SKIP111
ACE6 DOCKING
ACF3 SKIP112
AD12 SKIP113
AD26 SKIP114
AD35 SKIP115
AD61 SKIP116
AD6C SKIP117
AD70 SKIP118
AD71 SKIP119
AD82 SKIP120
ADB8 SKIP121
ADB9 SKIP122
ADCA LOOP041
ADD7 LOOP042
ADF1 MODDLST
ADF4 LOOP043
ADFB LOOP044
AE03 SKIP123
AE0D CLRPLAYFIELD
AE0F CLRMEM
AE1A LOOP045
AE29 TRIGGER
AE40 SKIP124
AE41 SKIP125
AE56 SKIP126
AE58 SKIP127
AE66 SKIP128
AEA8 NOISE
AEB1 SKIP129
AEB3 LOOP046
AEC9 SKIP130
AECA HOMINGVEL
AED2 SKIP131
AEDA SKIP132
AEE1 DAMAGE

```
AEE7  LOOP047
AF10  SKIP133
AF19  SKIP134
AF1E  SKIP135
AF32  SKIP136
AF3C  SKIP137
AF3D  COLLISION
AF3F  LOOP048
AF43  SKIP138
AF58  SKIP139
AF64  SKIP140
AF6F  SKIP141
AF94  SKIP142
AFC6  SKIP143
AFD5  LOOP049
AFE7  SKIP144
AFEC  LOOP050
AFF3  SKIP145
AFFD  SKIP146
AFFE  KEYBOARD
B011  LOOP051
B020  SKIP147
B02B  SKIP148
B036  SKIP149
B040  SKIP150
B041  SKIP151
B045  SETVIEW
B056  LOOP052
B060  SKIP152
B073  SKIP153
B07B  UPDSCREEN
B082  SKIP154
B096  SKIP155
B099  SKIP156
B0E6  SKIP157
B0ED  SKIP158
B0FB  SKIP159
B0FC  SKIP160
B106  SKIP161
B10A  GAMEOVER
B121  GAMEOVER2
B14A  SKIP162
B15A  SKIP163
B15D  SKIP164
B161  SKIP165
B162  SELECTWARP
B16A  SKIP166
B16B  SKIP167
B173  SKIP168
B1A7  CALCWARP
B1BE  SKIP169
B1C8  SKIP170
B1D3  SKIP171
B1E0  SKIP172
B1FE  LOOP053
B200  LOOP054
B212  SKIP173
B216  UPDTITLE
B21E  SKIP174
```

```
B21F  SKIP175
B223  SETTITLE
B22E  SKIP176
B234  LOOP055
B23A  SKIP177
B249  SKIP178
B25F  SKIP179
B268  SKIP180
B276  LOOP056
B27C  SKIP181
B286  LOOP057
B2A2  SKIP182
B2A8  SKIP183
B2AB  SOUND
B2C1  LOOP058
B2E1  SKIP184
B2E6  SKIP185
B2F3  SKIP186
B32B  SKIP187
B337  SKIP188
B349  SKIP189
B357  SKIP190
B369  SKIP191
B397  SKIP192
B39F  SKIP193
B3A6  BEEP
B3AF  LOOP059
B3B9  SKIP194
B3BA  INITIALIZE
B3BC  LOOP060
B3CA  SKIP195
B3EE  LOOP061
B41B  LOOP062
B441  LOOP063
B44C  LOOP064
B47C  SKIP196
B488  LOOP065
B492  LOOP066
B4B9  DRAWGC
B4BD  LOOP067
B4C6  SKIP197
B4E4  FLUSHGAMELOOP
B4F5  SKIP198
B50F  SKIP199
B511  SKIP200
B516  SKIP201
B51A  SKIP202
B51C  LOOP068
B527  SKIP203
B536  SKIP204
B53E  SKIP205
B544  SKIP206
B54E  LOOP069
B562  SKIP207
B565  SKIP208
B569  SKIP209
B56A  SKIP210
B574  SKIP211
B57C  LOOP070
```

```
B59A  SKIP212
B5B0  SKIP213
B5BB  SKIP214
B5C1  LOOP071
B5D1  LOOP072
B5DA  SKIP215
B5EA  LOOP073
B5EF  LOOP074
B601  LOOP075
B619  SKIP216
B61C  SKIP217
B61D  SKIP218
B632  LOOP076
B644  SKIP219
B655  SKIP220
B662  LOOP077
B664  LOOP078
B68D  SKIP221
B68F  SKIP222
B698  SKIP223
B69B  ROTATE
B6A4  SKIP224
B6E1  SKIP225
B6FB  SCREENCOLUMN
B709  SKIP226
B717  SKIP227
B71E  SCREENROW
B72E  SKIP228
B73E  SKIP229
B745  SKIP230
B74A  SKIP231
B753  SKIP232
B75A  SKIP233
B763  SKIP234
B764  INITPOSVEC
B785  SKIP235
B7A9  SKIP236
B7BE  RNDINVXY
B7D7  SKIP237
B7F0  SKIP238
B7F1  ISSURROUNDED
B803  SKIP239
B804  UPDPANEL
B810  SKIP240
B812  SKIP241
B822  SKIP242
B85C  SKIP243
B86F  DECENERGY
B88C  SKIP244
B88E  SKIP245
B896  LOOP079
B8A6  SKIP246
B8A7  SHOWCOORD
B8BD  SKIP247
B8CD  SHOWDIGITS
B8DF  PLCOLOROFFTAB
B8E4  PLSHAP1TAB
B9B1  PLSHAP2TAB
BA62  DLSTFRAG
```

```
 BA6A  DLSTFRAGGC
 BA6D  DLSTFRAGLRS
 BA75  DLSTFRAGAFT
 BA7D  DLSTFRAGFRONT
 BA8C  DLSTFRAGOFFTAB
 BA90  FOURCOLORPIXEL
 BAB0  PIXELMASKTAB
 BAB4  VELOCITYTAB
 BABE  KEYTAB
 BAD3  DRAINRATETAB
 BADD  WARPENERGYTAB
 BAF5  STICKINCTAB
 BAF9  DRAWLINESTAB
 BB3A  WARPSTARXTAB
 BB3E  WARPSTARYTAB
 BB42  PANELTXTTAB
 BBA6  SECTORTYPETAB
 BBAA  PHRASETAB
 BC2B  WORDTAB
 BE22  VIEWMODETAB
 BE26  MSGOFFTAB
 BE29  MSGBITTAB
 BE2C  MSGONTAB
 BE2F  PLSHAPOFFTAB
 BE7F  PLSHAPHEIGHTTAB
 BECF  TRACKKEYSTAB
 BED1  SECTORCHARTAB
 BED7  VEERMASKTAB
 BEDB  PLSTARBAOFFTAB
 BEDD  BONUSTAB
 BEE9  RANKTAB
 BEFC  CLASSTAB
 BF0C  MISSIONPHRTAB
 BF10  DAMAGEPROBTAB
 BF14  DAMAGEPHRTAB
 BF1A  DESTROYPHRTAB
 BF20  NOISEPATTAB
 BF3E  BEEPPATTAB
 BF5C  BEEPFRQTAB
 BF6E  BLIPSHAPTAB
 BF73  BARRELXTAB
 BF75  HITMAXZTAB
 BF7D  HITMINZTAB
 BF85  ZYLONHOMVELTAB
 BF89  ZYLONSHAPTAB
 BF91  ZYLONFLPATTAB
 BF99  ZYLONVELTAB
 BFA9  PFCOLORTAB
 BFB3  VICINITYMASKTAB
 BFBB  MOVEPROBTAB
 BFC0  COMPASSOFFTAB
 BFC9  HOMVELTAB
 BFD1  PLSHAPCOLORTAB
 BFDB  PLSHAPBRITTAB
 BFEB  NOISETORPVOLTAB
 BFF3  NOISETORPFRQTAB
=D000  HPOSP0
=D001  HPOSP1
=D002  HPOSP2
```

```
=D003  HPOSP3
=D004  HPOSM0
=D005  HPOSM1
=D006  HPOSM2
=D007  HPOSM3
=D008  M0PL
=D009  M1PL
=D00A  M2PL
=D00B  M3PL
=D00F  P3PL
=D010  TRIG0
=D012  COLPM0
=D016  COLPF0
=D01B  PRIOR
=D01D  GRACTL
=D01E  HITCLR
=D01F  CONSOL
=D200  AUDF1
=D202  AUDF2
=D203  AUDC2
=D204  AUDF3
=D205  AUDC3
=D206  AUDF4
=D207  AUDC4
=D208  AUDCTL
=D209  KBCODE
=D209  STIMER
=D20A  RANDOM
=D20E  IRQEN
=D20F  SKCTL
=D300  PORTA
=D302  PACTL
=D400  DMACTL
=D402  DLIST
=D407  PMBASE
=D409  CHBASE
=D40A  WSYNC
=D40B  VCOUNT
=D40E  NMIEN
=E000  ROMCHARSET
```