

```

LOCATION OBJECT CODE LINE SOURCE LINE
1248 ; .IDENT PLAYSON ;Includes TONE_OUT
1249 ; *****
1250 ; * PLAY_SONGS *****
1251 ; *****
1252 ; .COMMENT )
1253 ; see Users' Manual for description
1254 ; )
1255 ; GLB PLAY_SONGS_TONE_OUT
1256 ; EXT UPATMCTRL_UPFREQ
1257 ; INCLUDE OSSR EQU:OS ;equates
1258 ; *** Sound chip register code EQUATIES
1259 ; Tone generator frequency and attenuation formatted register codes
1260 ; SR1FRQ EQU 1000000008 ;BIT7 = 1, BIT6-4 = TONE GEN 1 FREQ CODE
1261 ; SR1ATN EQU 1001000008 ;BIT7 = 1, BIT6-4 = TONE GEN 1 ATTN CODE
1262 ; SR2FRQ EQU 1010000008 ;BIT7 = 1, BIT6-4 = TONE GEN 2 FREQ CODE
1263 ; SR2ATN EQU 1011000008 ;BIT7 = 1, BIT6-4 = TONE GEN 2 ATTN CODE
1264 ; SR3FRQ EQU 1100000008 ;BIT7 = 1, BIT6-4 = TONE GEN 3 FREQ CODE
1265 ; SR3ATN EQU 1101000008 ;BIT7 = 1, BIT6-4 = TONE GEN 3 ATTN CODE
1266 ; Noise generator control and attenuation formatted register codes
1267 ; SRNCTL EQU 1110000008 ;BIT7 = 1, BIT6-4 = NOISE GEN CONTROL CODE
1268 ; SRNATN EQU 1111000008 ;BIT7 = 1, BIT6-4 = NOISE GEN ATTN CODE
1269 ; Noise generator formatted control codes
1270 ; WHITE EQU 000001008 ;BIT2 = 1, WHITE NOISE CODE
1271 ; PERIOD EQU 000000008 ;BIT2 = 0, PERIODIC NOISE CODE
1272 ; MSRHI EQU 000000008 ;BIT0-1 SET FOR HIGHEST NOISE SHIFT RATE [N/512]
1273 ; MSRLW EQU 000000018 ;BIT0-1 SET FOR MEDIUM NOISE SHIFT RATE [N/1024]
1274 ; MSRLG EQU 000000108 ;BIT0-1 SET FOR LOWEST NOISE SHIFT RATE [N/2048]
1275 ; MSRTG3 EQU 000000118 ;BIT0-1 SET FOR SHIFT FROM TONE GEN 3 OUTPUT
1276 ; PLAY_SONGS
1277 ; * output CH1 attenuation and frequency
1278 ; LD A,OFF+SR1ATN ;format CH1 OFF byte into A
1279 ; LD C,SR1ATN ;format MSN C for CH1 attenuation
1280 ; LD D,SR1FRQ ;format MSN D for CH1 frequency
1281 ; LD IX,(PTR TO S_OM_1) ;point IX to byte 0 data area of song for CH1
1282 ; CALL TONE_OUT
1283 ; * output CH2 attenuation and frequency
1284 ; LD A,OFF+SR2ATN ;format CH2 OFF byte into A
1285 ; LD C,SR2ATN ;format MSN C for CH2 attenuation
1286 ; LD D,SR2FRQ ;format MSN D for CH2 frequency
1287 ; LD IX,(PTR TO S_OM_2) ;point IX to byte 0 data area of song for CH2
1288 ; CALL TONE_OUT
1289 ; * output CH3 attenuation and frequency
1290 ; LD A,OFF+SR3ATN ;format CH3 OFF byte into A
1291 ; LD C,SR3ATN ;format MSN C for CH3 attenuation
1292 ; LD D,SR3FRQ ;format MSN D for CH3 frequency
1293 ; LD IX,(PTR TO S_OM_3) ;point IX to byte 0 data area of song for CH3
1294 ; CALL TONE_OUT
1295 ; * output CH0 [noise] ATN [and CTRL, if different from last time]
1296 ; LD A,OFF+SRMATN ;format CH0 OFF byte into A
1297 ; LD C,SRMATN ;format MSN C for CH0 attenuation
1298 ; LD IX,(PTR TO S_OM_0) ;point IX to byte 0 data area of song for CH0
1299 ; LD E,(IX+0) ;look for inactive code, OFFH
1300 ; INC E ;this sets Z flag if E = OFFH
1301 ; IF (PSW,IS,ZERO)
1302 ; JR NZ,L5 ; song data area is inactive
1303 ; OUT (SOUND_PORT),A ; turn off CH0
1304 ; JR L6

```

```

LOCATION OBJECT CODE LINE SOURCE LINE
03A1 C30461 1391 JP M00B0 ;to load byte 0
1392 ; ENDF
1393 ; - test for special sound effect
03A4 E63C 1394 L14 AMD 00111100B ;mask irrelevant bits
03A6 FE04 1395 CP 00000100B ;test for B5 - B2 = 0001
1396 ; IF [PSW,IS,ZERO] ;note is a special effect
03A8 2028 1397 JR NZ,L15
1398 ;---CASE-- special effect
03AA FDE1 1399 POP IY ;IY := SONGNO
03AC FDE5 1400 PUSH IY ;put SONGNO back on stack
03AE C5 1401 PUSH BC ;save header on stack; NEXT_NOTE_PIR := SFX, DE := SFX
03AF 23 1402 IMC HL ;-pt HL to next byte [LSB addr SFX]
03B0 5E 1403 LD E,[HL] ;-E := LSB SFX
03B1 D07301 1404 LD [IX+1],E ;-put LSB of SFX in byte 1 of SxDATA [NEXT_NOTE_PIR]
03B4 23 1405 IMC HL ;-D := MSB SFX
03B5 56 1406 LD D,[HL] ;-D := MSB SFX
03B6 D07202 1407 LD [IX+2],D ;-put MSB SFX in byte 2 of SxDATA
03B9 23 1408 IMC HL ;point HL to next note [after this new note]
03BA FDE5 1409 POP AF ;A := SONGNO
03BC F1 1410 POP IY
03BD 05 1411 PUSH DE
03BE FDE1 1412 POP IY
03C0 1103C6 1413 LD DE,PASS1
03C3 05 1414 PUSH DE
03C4 FDE9 1415 JP [IY] ;create "CALL [IY]" with RET to PASS1 by storing
03C6 1600 1416 PASS1 ;PASS1 on the stack
03C8 1E07 1417 LD E,7 ;1st 7 bytes SFX will save addr next note & SONGNO
03CA FD19 1418 ADD IY,DE ;to allow SFX to load initial values
03CC 110461 1419 LD DE,M00B0 ;RET to M00B0
03CF 05 1420 PUSH DE
03D0 FDE9 1421 JP [IY]
1422 ; ENDF
1423 ; - if here, note is type 0 - 3
1424 L15 1424 L15 ;save header on stack
03D2 C5 1425 PUSH BC ;A := fresh copy header
03D3 78 1426 LD A,B ;mask all but type number
03D4 E603 1427 AMD 00000011B ;test for type 0
03D6 FE00 1428 CP 0 ;note is type 0: fixed freq and atrn
03D8 2020 1429 JR NZ,L16
1430 ; --CASE-- note type 0
1431 ; * set up NEXT_NOTE_PIR
03DA 23 1432 IMC HL ;next note [after this new note] is 4 bytes away,
03DB 23 1433 IMC HL ;point HL to it
03DC 23 1434 IMC HL
03DE 23 1435 IMC HL
03DF 23 1436 LD [IX+1],L ;put addr in NEXT_NOTE_PIR
03E0 D07501 1437 LD [IX+2],H ;* move new note data and fill in bytes where necessary
03E1 D07402 1438 ; DEC HL ;point HL back to 1st ROM data to move, NLEN
03E4 28 1439 LD DE,05 ;point DE to destination: bytes 5, 4, and 3
03E5 110005 1440 CALL DE,IO_DEST ;move 3 bytes
03E8 CD0478 1441 LDOR ;set for no freq sweep
03EB 010003 1442 LD [IX+FSSTEP],0 ;- [IX+FSSTEP],0
03EE ED88 1443 ;- [IX+FSSTEP],0
03F0 D0360700 1444 ;- [IX+FSSTEP],0
03F1 D0360700 1445 ;- [IX+FSSTEP],0
03F2 D0360700 1446 ;- [IX+FSSTEP],0
03F3 D0360700 1447 ;- [IX+FSSTEP],0
03F4 D0360700 1448 ;- [IX+FSSTEP],0
03F5 D0360700 1449 ;- [IX+FSSTEP],0
03F6 D0360700 1450 ;- [IX+FSSTEP],0
03F7 D0360700 1451 ;- [IX+FSSTEP],0
03F8 D0360700 1452 ;- [IX+FSSTEP],0
03F9 D0360700 1453 ;- [IX+FSSTEP],0
03FA D0360700 1454 ;- [IX+FSSTEP],0
03FB D0360700 1455 ;- [IX+FSSTEP],0
03FC D0360700 1456 ;- [IX+FSSTEP],0
03FD D0360700 1457 ;- [IX+FSSTEP],0
03FE D0360700 1458 ;- [IX+FSSTEP],0
03FF D0360700 1459 ;- [IX+FSSTEP],0

```

LOCATION OBJECT CODE LINE SOURCE LINE

```

03FA FE01      1448 L16      CP 1
03FC 201B      1449 ;          ;test for type 1
                1450 IF [PSM,IS,ZERO]
                ;note is type 1: swept freq, fixed attenuation
                1451 ;
                --CASE-- note type 1
                * set up NEXT_NOTE_PTR
                LD E,6
                LD D,0
                ADD HL,DE
                LD [IX+1],L
                LD [IX+2],H
                ;store in NEXT_NOTE_PTR
                * move new note data and fill in bytes where necessary
                DEC HL
                INC E
                CALL DE,TO_DEST
                LD BC,5
                LDDR
                LD [IX+ASTEP],0
                JR MOD80
                ENDDIF
                1466 ;
                1467 L17      CP 2
                1468 ;          ;test for type 2
                1469 IF [PSM,IS,ZERO]
                ;note is type 2: fixed freq, swept attenuation
                1470 ;
                --CASE-- note type 2
                * set up NEXT_NOTE_PTR
                LD E,6
                LD D,0
                ADD HL,DE
                POP AF
                PUSH AF
                AND 11000000B
                IF [PSM,IS,ZERO]
                JR NZ,L18
                DEC HL
                ENDDIF
                1481 ;
                1482 L18      LD [IX+1],L
                LD [IX+2],H
                * move new note data and fill in bytes where necessary
                DEC HL
                LD E,9
                CALL DE,TO_DEST
                LD BC,2
                LDDR
                LD A,0
                LD [DE],A
                DEC DE
                DEC DE
                LD C,3
                LDDR
                JR MOD80
                ENDDIF
                1497 ;
                1498 ;          ;if here, note is type 3: swept freq, swept attenuation
                1499 --CASE-- note type 3
                1500 * set up NEXT_NOTE_PTR
                LD E,8
                LD D,0
                ADD HL,DE
                LD [IX+1],L
                1504 ;          ;put addr in NEXT_NOTE_PTR

```

LOCATION	OBJECT CODE	LINE	SOURCE LINE
0440	D07402	1505	LD [IX+2],H
0450	2B	1506	* move new note data and fill in bytes where necessary
0451	D0E5	1507	DEC HL
0453	FDE1	1508	PUSH IX
0455	FD19	1509	POP IX
0457	FD19	1510	LD E,9
0459	FDE5	1511	ADD IX,DE
045B	D1	1512	PUSH IX
045C	010007	1513	POP DE
045F	EDB8	1514	LD BC,7
		1515	LDDR
		1516	* modify byte 0 basis header new note
0461	D0E5	1517	PUSH IX
0463	E1	1518	POP HL
0464	F1	1519	POP AF
0465	C1	1520	POP BC
0466	FEFF	1521	CP INACTIVE
0468	C8	1522	RET Z
0469	57	1523	LD D,A
046A	E63F	1524	AND 3FH
046C	FE04	1525	CP 04
046E	2002	1526	JR NZ,L20_LOAD_MEX
0470	063E	1527	LD B,62
0472		1528	L20_LOAD_MEX:
0472	7A	1529	LD A,D
0473	E6C0	1530	AND 0C0H
0475	80	1531	OR B
0476	77	1532	LD [HL],A
		1533	ENDIF
0477	C9	1534	RET
0478		1535	DE_TO_DEST
0478	D0E5	1536	PUSH IX
047A	FDE1	1537	POP IX
047C	FD19	1538	ADD IX,DE
047E	FDE5	1539	PUSH IX
0480	D1	1540	POP DE
0481	C9	1541	RET
		1542	END ;LOADNEX
		1543	PROG

```

;point HL back to 1st ROM data to move, APS
;point DE to destination: bytes 9 - 3
;IX := addr byte 0 (and DE = 6)
;DE := 9
;IX := addr byte 9 (APS)
;DE := addr APS
;move 7 bytes
;pt HL to byte 0
;A := header new note
;B := SONGNO
;test for inactive (song over, as detected above)
;save header in D
;Rid channel bits
;Special effect
;restore A to header
;A := CH# 0 0 0 0
;A := new CH# | SONGNO
;store back in byte 0
;DE passed = offset from byte 0, RETed w addr byte offset
;IX := addr byte 0 (and DE = offset)
;IX := addr byte 0 + offset
;DE := addr of destination byte in SxDATA

```

```

LOCATION OBJECT CODE LINE SOURCE LINE
1545 ; .IDENT ACTIVATE
1546 ; .ZOP
1547 ; .EPOP
1548 ; .COMMENT )
1549 ; ***** ACTIVATE *****
1550 ;
1551 ;
1552 ;
1553 ;
1554 ; THE FOLLOWING CHANGES/REVISIONS WERE MADE:
1555 ;
1556 ; 1. ELIMINATE CODE PLACING OLD SCREEN ADDRESS IN STATUS AREA
1557 ; 2. INIT X.PAT.POS IN OLD SCREEN WHEN IN VRAM AS WELL AS WHEN IN CRAM
1558 ; 3. USE VOP.MODE.WORD TO TEST GRAPHICS MODE
1559 ; 4. ADD CODE TO EXPAND ONE COLOR GENERATOR BYTE TO 8
1560 ; 5. ADDED C.BUFF.DEFS.8 FOR COLOR EXPANDING CODE
1561 ; 5/02 6. FIX COLOR GEN MOVE IN MODE 1
1562 ; 7. USE CONTROLER_MAP FOR BUFFER AREA
1563 ;
1564 ; ACTIVATE is used to initialize the RAM status area for the passed
1565 ; object and move its pattern and color generators to the PATTERN and
1566 ; COLOR GENERATOR tables in VRAM. The second function is enabled or
1567 ; disabled by setting or resetting the carry flag in the PSY. This is
1568 ; necessary to prevent sending the same graphics data to VRAM more than
1569 ; once when creating identical objects. The calling sequence for act-
1570 ; ivating an object is as follows:
1571 ;
1572 ; LD HL,OBJ_n ;->OBJ TO ACTIVATE
1573 ; SCF ;SIGNAL MV TO VRAM
1574 ; CALL ACTIVATE
1575 ;
1576 ;
1577 ;OR
1578 ;
1579 ; LD HL,OBJ_n ;->OBJ TO ACTIVATE
1580 ; OR A ;
1581 ; CALL ACTIVATE
1582 ;
1583 ; )
1584 ; EXT PUT VRAM_VRAM_WRITE,VOP_MODE_WORD
1585 ; EXT WORK_BUFFER
1586 ;
1587 ; GLB ACTIVATE_
1588 ;
1589 ; REGISTER USAGE: FOLLOWING WILL BE CHANGED BY ACTIVATE, ADDITIONAL
1590 ; MAY BE CHANGED BY CALLED SUBR
1591 ; AF,HL,DE,BC,IX
1592 ;
1593 ;
1594 ;
1595 ; PROCEDURE ACTIVATEQVAR OBJ:OBJECT;MOVE:BOOLEANJ;
1596 ;
1597 ; ACTIVATEQ IS THE PASCAL ENTRY POINT TO ACTIVATE
1598 ;
1599 ; EXT PARAM
1600 ; THE PASCAL PARAMETER PASSING PROCEDURE
1601 ; COMN

```

4/22/82
13:50:00


```

LOCATION OBJECT CODE LINE SOURCE LINE
1658 ; DE->OBJ GRAPHICS+0
1659 ; BC->OBJ STATUS+0
1660 ; A=0
1661 ;
<04C6> 1662 ACT CNPLX EQU $
1663 ; SUBCASE Complex
1664 LD A, [DE] ;GET COMP_CNT
1665 RRA
1666 RRA
1667 RRA
1668 RRA
1669 AND
1670 LD
1671 LD
1672 INC
1673 LD
1674 INC
1675 OR
1676 JR
1677 EQU
1678 POP
1679 AF
1680 PUSH
1681 PUSH
1682 EX
1683 CALL
1684 POP
1685 POP
1686 LD
1687 INC
1688 LD
1689 INC
1690 DJNZ
1691 POP
1692 RET
1693 ;
<04E7> 1694 ACT SEMI EQU $
1695 ; SUBCASE Semi_Mobile
1696 LD INIT_XP_OS
1697 LD A, [DE]
1698 LD L, A
1699 INC DE
1700 LD A, [DE]
1701 ADD A, L
1702 LD (1Y+5), A
1703 LD H, 0
1704 ; AT THIS POINT:
1705 ; STACK=OBJ_TYPE & SUP VRAM FLG
1706 ; HL=FIRST_GEN_NAME
1707 ; DE->NUMGEN
1708 ; BC:FREE
1709 ; SUP FOR VRAM IMIT
1710 POP
1711 JR
1712 PUSH
1713 LD
1714 BIT
1715 ; IF SUP VRAM FLG ON
1716 NC, SEMI_EXIT
1717 AF
1718 A, [VDP_MODE_WORD] ;SEE WHICH GRAPHICS MODE
1719 1, A ; IF GR 11 MODE
1720 ;
1721 ;
1722 ;
1723 ;
1724 ;
1725 ;
1726 ;
1727 ;
1728 ;
1729 ;
1730 ;
1731 ;
1732 ;
1733 ;
1734 ;
1735 ;
1736 ;
1737 ;
1738 ;
1739 ;
1740 ;
1741 ;
1742 ;
1743 ;
1744 ;
1745 ;
1746 ;
1747 ;
1748 ;
1749 ;
1750 ;
1751 ;
1752 ;
1753 ;
1754 ;
1755 ;
1756 ;
1757 ;
1758 ;
1759 ;
1760 ;
1761 ;
1762 ;
1763 ;
1764 ;
1765 ;
1766 ;
1767 ;
1768 ;
1769 ;
1770 ;
1771 ;
1772 ;
1773 ;
1774 ;
1775 ;
1776 ;
1777 ;
1778 ;
1779 ;
1780 ;
1781 ;
1782 ;
1783 ;
1784 ;
1785 ;
1786 ;
1787 ;
1788 ;
1789 ;
1790 ;
1791 ;
1792 ;
1793 ;
1794 ;
1795 ;
1796 ;
1797 ;
1798 ;
1799 ;
1800 ;
1801 ;
1802 ;
1803 ;
1804 ;
1805 ;
1806 ;
1807 ;
1808 ;
1809 ;
1810 ;
1811 ;
1812 ;
1813 ;
1814 ;
1815 ;
1816 ;
1817 ;
1818 ;
1819 ;
1820 ;
1821 ;
1822 ;
1823 ;
1824 ;
1825 ;
1826 ;
1827 ;
1828 ;
1829 ;
1830 ;
1831 ;
1832 ;
1833 ;
1834 ;
1835 ;
1836 ;
1837 ;
1838 ;
1839 ;
1840 ;
1841 ;
1842 ;
1843 ;
1844 ;
1845 ;
1846 ;
1847 ;
1848 ;
1849 ;
1850 ;
1851 ;
1852 ;
1853 ;
1854 ;
1855 ;
1856 ;
1857 ;
1858 ;
1859 ;
1860 ;
1861 ;
1862 ;
1863 ;
1864 ;
1865 ;
1866 ;
1867 ;
1868 ;
1869 ;
1870 ;
1871 ;
1872 ;
1873 ;
1874 ;
1875 ;
1876 ;
1877 ;
1878 ;
1879 ;
1880 ;
1881 ;
1882 ;
1883 ;
1884 ;
1885 ;
1886 ;
1887 ;
1888 ;
1889 ;
1890 ;
1891 ;
1892 ;
1893 ;
1894 ;
1895 ;
1896 ;
1897 ;
1898 ;
1899 ;
1900 ;
1901 ;
1902 ;
1903 ;
1904 ;
1905 ;
1906 ;
1907 ;
1908 ;
1909 ;
1910 ;
1911 ;
1912 ;
1913 ;
1914 ;
1915 ;
1916 ;
1917 ;
1918 ;
1919 ;
1920 ;
1921 ;
1922 ;
1923 ;
1924 ;
1925 ;
1926 ;
1927 ;
1928 ;
1929 ;
1930 ;
1931 ;
1932 ;
1933 ;
1934 ;
1935 ;
1936 ;
1937 ;
1938 ;
1939 ;
1940 ;
1941 ;
1942 ;
1943 ;
1944 ;
1945 ;
1946 ;
1947 ;
1948 ;
1949 ;
1950 ;
1951 ;
1952 ;
1953 ;
1954 ;
1955 ;
1956 ;
1957 ;
1958 ;
1959 ;
1960 ;
1961 ;
1962 ;
1963 ;
1964 ;
1965 ;
1966 ;
1967 ;
1968 ;
1969 ;
1970 ;
1971 ;
1972 ;
1973 ;
1974 ;
1975 ;
1976 ;
1977 ;
1978 ;
1979 ;
1980 ;
1981 ;
1982 ;
1983 ;
1984 ;
1985 ;
1986 ;
1987 ;
1988 ;
1989 ;
1990 ;
1991 ;
1992 ;
1993 ;
1994 ;
1995 ;
1996 ;
1997 ;
1998 ;
1999 ;
2000 ;

```

LOCATION OBJECT CODE LINE SOURCE LINE

```

04FD 2831      1715      JR      Z,SEMI_GRI
04FF EB       1716      EX      DE,HL
0500 44       1717      LD      B,H
0501 40       1718      LD      C,L
0502 6E       1719      LD      L,[HL]
0503 2600     1720      LD      H,0
0505 E5       1721      PUSH   HL,HL
0506 29       1722      ADD    ADD
0507 29       1723      ADD    HL,HL
0508 29       1724      ADD    HL,HL
0509 E5       1725      PUSH   HL,HL
050A 03       1726      INC    BC
050B 0A       1727      LD     LD A,[BC]
050C 6F       1728      LD     L,A
050D 03       1729      INC    BC
050E 0A       1730      LD     A,[BC]
050F 67       1731      LD     M,A
0510 C1       1732      POP   BC
0511 FDE1     1733      POP   IY
0513 F1       1734      POP   AF

1735      ;AT THIS POINT:
1736      HL->SOURCE BUFFER, PTRN_CNTRLS
1737      DE=INDEX TO START OF VRAM ENTRIES
1738      IY=NUMBER OF ITEMS TO READ FROM VRAM
1739      BC=OFFSET TO COLOR SOURCE BUFFER @
1740      AF=OBJ_TYPE (8 SUP VRAM FLG, UNNEEDED)
1741      ; FILL AS NEEDED TOP, MID, AND BOT PTRN_CNTRLS & DITTO FOR COLOR_CNTRLS
1742      BIT 7 A
1743      JR 7,A
1744      CALL SUP_GEN_CLR
1745      EQU SUP_UPDATE
1746      CALL 6,A
1747      BIT 6 A
1748      JR 7,A
1749      CALL SUP_GEN_CLR
1750      EQU SUP_UPDATE
1751      CALL 5,A
1752      BIT 5 A
1753      JR 7,A
1754      CALL SUP_GEN_CLR
1755      EQU SUP_EXIT
1756      RET
1757      ;
1758      ; Handle GRAPHICS MODE 1
1759      SEMI_GRI
1760      EQU
1761      EX DE,HL
1762      LD C,[HL]
1763      LD B,0
1764      PUSH BC
1765      POP  IY
1766      INC  HL,HL
1767      LD  A,[HL]
1768      INC  HL,HL
1769      LD  L,A
1770      PUSH HL
1771      PUSH BC

; GO GRI
;DE=FIRST_GEN_NAME
;SV -> NUMGEN
;CALC SOURCE OFFSET
;HL->SOURCE BUFFER
;HL->NUMGEN
;IY=NUMGEN
;HL->PTRN_CNTRLS
;SAVE FOR RESTORE

```

LOCATION OBJECT CODE LINE SOURCE LINE

```

053E D5 1772 DE
053F FDE5 1773 IY
0541 3E03 1774 A,3 ;SIGNAL PTRN GEN FILL
0543 CD1C27 1775 CALL PUT VRAM
0546 C1 1776 POP BC ;BC := MUMGEN
0547 E1 1777 POP HL ;HL := FIRST_GEN_NAME
0548 50 1778 LD E,L
0549 54 1779 LD D,H ;DE := FIRST_GEN_NAME
054A 09 1780 ADD HL,BC ;HL := FIRST_GEN_NAME + MUMGEN
054B 28 1781 DEC HL
054C CB3C 1782 SRL H
054E CB10 1783 RR L
0550 CB3C 1784 SRL H
0552 CB10 1785 RR L
0554 CB3C 1786 SRL H
0556 CB10 1787 RR L
0558 CB28 1788 SRA E
055A CB28 1789 SRA E
055C CB28 1790 SRA E ;DE := FIRST_GEN_NAME/B
055E B7 1791 OR A ;CLEAR CARRY
055F ED52 1792 SBC HL,DE
0561 23 1793 IMC HL ;HL := (F_G_M + MUMGEN - 1)/B - F_G_M/B + 1 = NUMBER COLR GENS
0562 E5 1794 PUSH HL
0563 FDE1 1795 POP IY
0565 E1 1796 POP
0566 29 1797 ADD HL,HL
0567 29 1798 ADD HL,HL
0568 29 1799 ADD HL,HL
0569 C1 1800 POP BC
056A 09 1801 HL,BC
056B 3E04 1802 LD A,4 ;HL->COLOR GNRTR SOURCE
056D CD1C27 1803 CALL PUT VRAM ;SIGNAL PTRN COLOR TBL
0570 F1 1804 POP AF ;FIX STACK
0571 C9 1805 RET
1806 ; Internal routine to initialize X_Pat_Pos in Old_Screen
1807 INIT_XP_OS:
1808 POP BC
1809 PUSH IY ;IY -> STATUS
1810 PUSH DE ;SAVE -> GRAPHICS
1811 LD E,(HL) ;DE := OLD_SCREEN ADDRESS
1812 LD D,(HL)
1813 LD 7,D ;? OLD SCRIN IN CROM
1814 JR NZ,SM_BY_OLD ;OLD_SCREEN IN VRAM?
1815 LD A,D
1816 CP 70H
1817 JR C,OS_IN_VRAM
1818 LD A,80H ;INIT X_PAT_POS = 80H
1819 LD (DE),A
1820 JR SM_BY_OLD
1821 INIT_80:
1822 OS_IN_VRAM:
1823 LD HL,INIT_80
1824 LD BC,1
1825 CALL VRAM_WRITE ;ONE BYTE TO MOVE TO VRAM
1826 EQU
1827 POP DE ;DE -> GRAPHICS
1828 INC DE ;DE -> FIRST_GEN_NAME
1829 RET

```

LOCATION OBJECT CODE LINE SOURCE LINE

```

1829 ;
1830 ; Internal rout to setup Ptrn Gen VRAM & Color Gen VRAM
1831 SUP_GEN_CLR EQU $
1832 PUSH AF ;SAVE FOR RESTORE
1833 PUSH BC
1834 PUSH IY
1835 PUSH DE
1836 PUSH HL
1837 LD A,3 ;SIGNAL PTRN GEN FILL
1838 CALL PUT_VRAM_
1839 POP HL ;RESTORE
1840 POP DE
1841 POP IY
1842 POP BC
1843 POP AF
1844 PUSH AF ;SAVE FOR RESTORE
1845 PUSH BC
1846 PUSH IY
1847 POP DE
1848 HL ;HOW MANY COLOR GEN BYTES?
1849 ;MZ,ONE_BYTE
1850 ;HL->COLOR GEN SOURCE
1851 ;A,4
1852 ;SIGNAL PTRN COLOR FILL
1853 CALL PUT_VRAM_
1854 HL
1855 O_B_RET: POP DE
1856 POP IY
1857 POP BC
1858 POP AF
1859 RET
1860 ; For each item to send, duplicate the color byte 8 times (in C_BUFFER)
1861 ; then send this generator to VRAM color table indexed by DE
1862 ONE_BYTE:
1863 ADD HL,BC ;HL -> COLOR BYTE
1864 LD C,L
1865 LD B,H ;BC -> COLOR BYTE
1866 PUSH IY
1867 POP HL ;HL = ITEM COUNT
1868 NEXT_COLOR:
1869 PUSH HL ;SAVE COUNTER
1870 LD A,(BC) ;GET COLOR BYTE
1871 PUSH BC ;SAVE POINTER TO COLOR
1872 LD BC,B ;CREATE 8 DUPLICATES
1873 LD HL,(WORK_BUFFER)
1874 ADD HL,BC ;PLACE THEM HERE, STARTING AT END OF BUFFER
1875 LD B,B
1876 DUPLI: DEC HL
1877 LD (HL),A
1878 DJNZ DUPLI
1879 PUSH DE
1880 LD IY,1
1881 LD A,4
1882 CALL PUT_VRAM_
1883 POP DE ;GET INDEX BACK
1884 POP BC ;POINTER TO COLOR BYTE
1885 INC DE ;INCREMENT INDEX
1886
1887
1888
1889
1890
1891
1892
1893
1894
1895
1896
1897
1898
1899
1900
1901
1902
1903
1904
1905
1906
1907
1908
1909
1910
1911
1912
1913
1914
1915
1916
1917
1918
1919
1920
1921
1922
1923
1924
1925
1926
1927
1928
1929
1930
1931
1932
1933
1934
1935
1936
1937
1938
1939
1940
1941
1942
1943
1944
1945
1946
1947
1948
1949
1950
1951
1952
1953
1954
1955
1956
1957
1958
1959
1960
1961
1962
1963
1964
1965
1966
1967
1968
1969
1970
1971
1972
1973
1974
1975
1976
1977
1978
1979
1980
1981
1982
1983
1984
1985
1986
1987
1988
1989
1990
1991
1992
1993
1994
1995
1996
1997
1998
1999
2000
2001
2002
2003
2004
2005
2006
2007
2008
2009
2010
2011
2012
2013
2014
2015
2016
2017
2018
2019
2020
2021
2022
2023
2024
2025
2026
2027
2028
2029
2030
2031
2032
2033
2034
2035
2036
2037
2038
2039
2040
2041
2042
2043
2044
2045
2046
2047
2048
2049
2050
2051
2052
2053
2054
2055
2056
2057
2058
2059
2060
2061
2062
2063
2064
2065
2066
2067
2068
2069
2070
2071
2072
2073
2074
2075
2076
2077
2078
2079
2080
2081
2082
2083
2084
2085
2086
2087
2088
2089
2090
2091
2092
2093
2094
2095
2096
2097
2098
2099
2100
2101
2102
2103
2104
2105
2106
2107
2108
2109
2110
2111
2112
2113
2114
2115
2116
2117
2118
2119
2120
2121
2122
2123
2124
2125
2126
2127
2128
2129
2130
2131
2132
2133
2134
2135
2136
2137
2138
2139
2140
2141
2142
2143
2144
2145
2146
2147
2148
2149
2150
2151
2152
2153
2154
2155
2156
2157
2158
2159
2160
2161
2162
2163
2164
2165
2166
2167
2168
2169
2170
2171
2172
2173
2174
2175
2176
2177
2178
2179
2180
2181
2182
2183
2184
2185
2186
2187
2188
2189
2190
2191
2192
2193
2194
2195
2196
2197
2198
2199
2200
2201
2202
2203
2204
2205
2206
2207
2208
2209
2210
2211
2212
2213
2214
2215
2216
2217
2218
2219
2220
2221
2222
2223
2224
2225
2226
2227
2228
2229
2230
2231
2232
2233
2234
2235
2236
2237
2238
2239
2240
2241
2242
2243
2244
2245
2246
2247
2248
2249
2250
2251
2252
2253
2254
2255
2256
2257
2258
2259
2260
2261
2262
2263
2264
2265
2266
2267
2268
2269
2270
2271
2272
2273
2274
2275
2276
2277
2278
2279
2280
2281
2282
2283
2284
2285
2286
2287
2288
2289
2290
2291
2292
2293
2294
2295
2296
2297
2298
2299
2300
2301
2302
2303
2304
2305
2306
2307
2308
2309
2310
2311
2312
2313
2314
2315
2316
2317
2318
2319
2320
2321
2322
2323
2324
2325
2326
2327
2328
2329
2330
2331
2332
2333
2334
2335
2336
2337
2338
2339
2340
2341
2342
2343
2344
2345
2346
2347
2348
2349
2350
2351
2352
2353
2354
2355
2356
2357
2358
2359
2360
2361
2362
2363
2364
2365
2366
2367
2368
2369
2370
2371
2372
2373
2374
2375
2376
2377
2378
2379
2380
2381
2382
2383
2384
2385
2386
2387
2388
2389
2390
2391
2392
2393
2394
2395
2396
2397
2398
2399
2400
2401
2402
2403
2404
2405
2406
2407
2408
2409
2410
2411
2412
2413
2414
2415
2416
2417
2418
2419
2420
2421
2422
2423
2424
2425
2426
2427
2428
2429
2430
2431
2432
2433
2434
2435
2436
2437
2438
2439
2440
2441
2442
2443
2444
2445
2446
2447
2448
2449
2450
2451
2452
2453
2454
2455
2456
2457
2458
2459
2460
2461
2462
2463
2464
2465
2466
2467
2468
2469
2470
2471
2472
2473
2474
2475
2476
2477
2478
2479
2480
2481
2482
2483
2484
2485
2486
2487
2488
2489
2490
2491
2492
2493
2494
2495
2496
2497
2498
2499
2500
2501
2502
2503
2504
2505
2506
2507
2508
2509
2510
2511
2512
2513
2514
2515
2516
2517
2518
2519
2520
2521
2522
2523
2524
2525
2526
2527
2528
2529
2530
2531
2532
2533
2534
2535
2536
2537
2538
2539
2540
2541
2542
2543
2544
2545
2546
2547
2548
2549
2550
2551
2552
2553
2554
2555
2556
2557
2558
2559
2560
2561
2562
2563
2564
2565
2566
2567
2568
2569
2570
2571
2572
2573
2574
2575
2576
2577
2578
2579
2580
2581
2582
2583
2584
2585
2586
2587
2588
2589
2590
2591
2592
2593
2594
2595
2596
2597
2598
2599
2600
2601
2602
2603
2604
2605
2606
2607
2608
2609
2610
2611
2612
2613
2614
2615
2616
2617
2618
2619
2620
2621
2622
2623
2624
2625
2626
2627
2628
2629
2630
2631
2632
2633
2634
2635
2636
2637
2638
2639
2640
2641
2642
2643
2644
2645
2646
2647
2648
2649
2650
2651
2652
2653
2654
2655
2656
2657
2658
2659
2660
2661
2662
2663
2664
2665
2666
2667
2668
2669
2670
2671
2672
2673
2674
2675
2676
2677
2678
2679
2680
2681
2682
2683
2684
2685
2686
2687
2688
2689
2690
2691
2692
2693
2694
2695
2696
2697
2698
2699
2700
2701
2702
2703
2704
2705
2706
2707
2708
2709
2710
2711
2712
2713
2714
2715
2716
2717
2718
2719
2720
2721
2722
2723
2724
2725
2726
2727
2728
2729
2730
2731
2732
2733
2734
2735
2736
2737
2738
2739
2740
2741
2742
2743
2744
2745
2746
2747
2748
2749
2750
2751
2752
2753
2754
2755
2756
2757
2758
2759
2760
2761
2762
2763
2764
2765
2766
2767
2768
2769
2770
2771
2772
2773
2774
2775
2776
2777
2778
2779
2780
2781
2782
2783
2784
2785
2786
2787
2788
2789
2790
2791
2792
2793
2794
2795
2796
2797
2798
2799
2800
2801
2802
2803
2804
2805
2806
2807
2808
2809
2810
2811
2812
2813
2814
2815
2816
2817
2818
2819
2820
2821
2822
2823
2824
2825
2826
2827
2828
2829
2830
2831
2832
2833
2834
2835
2836
2837
2838
2839
2840
2841
2842
2843
2844
2845
2846
2847
2848
2849
2850
2851
2852
2853
2854
2855
2856
2857
2858
2859
2860
2861
2862
2863
2864
2865
2866
2867
2868
2869
2870
2871
2872
2873
2874
2875
2876
2877
2878
2879
2880
2881
2882
2883
2884
2885
2886
2887
2888
2889
2890
2891
2892
2893
2894
2895
2896
2897
2898
2899
2900
2901
2902
2903
2904
2905
2906
2907
2908
2909
2910
2911
2912
2913
2914
2915
2916
2917
2918
2919
2920
2921
2922
2923
2924
2925
2926
2927
2928
2929
2930
2931
2932
2933
2934
2935
2936
2937
2938
2939
2940
2941
2942
2943
2944
2945
2946
2947
2948
2949
2950
2951
2952
2953
2954
2955
2956
2957
2958
2959
2960
2961
2962
2963
2964
2965
2966
2967
2968
2969
2970
2971
2972
2973
2974
2975
2976
2977
2978
2979
2980
2981
2982
2983
2984
2985
2986
2987
2988
2989
2990
2991
2992
2993
2994
2995
2996
2997
2998
2999
3000
3001
3002
3003
3004
3005
3006
3007
3008
3009
3010
3011
3012
3013
3014
3015
3016
3017
3018
3019
3020
3021
3022
3023
3024
3025
3026
3027
3028
3029
3030
3031
3032
3033
3034
3035
3036
3037
3038
3039
3040
3041
3042
3043
3044
3045
3046
3047
3048
3049
3050
3051
3052
3053
3054
3055
3056
3057
3058
3059
3060
3061
3062
3063
3064
3065
3066
3067
3068
3069
3070
3071
3072
3073
3074
3075
3076
3077
3078
3079
3080
3081
3082
3083
3084
3085
3086
3087
3088
3089
3090
3091
3092
3093
3094
3095
3096
3097
3098
3099
3100
3101
3102
3103
3104
3105
3106
3107
3108
3109
3110
3111
3112
3113
3114
3115
3116
3117
3118
3119
3120
3121
3122
3123
3124
3125
3126
3127
3128
3129
3130
3131
3132
3133
3134
3135
3136
3137
3138
3139
3140
3141
3142
3143
3144
3145
3146
3147
3148
3149
3150
3151
3152
3153
3154
3155
3156
3157
3158
3159
3160
3161
3162
3163
3164
3165
3166
3167
3168
3169
3170
3171
3172
3173
3174
3175
3176
3177
3178
3179
3180
3181
3182
3183
3184
3185
3186
3187
3188
3189
3190
3191
3192
3193
3194
3195
3196
3197
3198
3199
3200
3201
3202
3203
3204
3205
3206
3207
3208
3209
3210
3211
3212
3213
3214
3215
3216
3217
3218
3219
3220
3221
3222
3223
3224
3225
3226
3227
3228
3229
3230
3231
3232
3233
3234
3235
3236
3237
3238
3239
3240
3241
3242
3243
3244
3245
3246
3247
3248
3249
3250
3251
3252
3253
3254
3255
3256
3257
3258
3259
3260
3261
3262
3263
3264
3265
3266
3267
3268
3269
3270
3271
3272
3273
3274
3275
3276
3277
3278
3279
3280
3281
3282
3283
3284
3285
3286
3287
3288
3289
3290
3291
3292
3293
3294
3295
3296
3297
3298
3299
3300
3301
3302
3303
3304
3305
3306
3307
3308
3309
3310
3311
3312
3313
3314
3315
3316
3317
3318
3319
3320
3321
3322
3323
3324
3325
3326
3327
3328
3329
3330
3331
3332
3333
3334
3335
3336
3337
3338
3339
3340
3341
3342
3343
3344
3345
3346
3347
3348
3349
3350
3351
3352
3353
3354
3355
3356
3357
3358
3359
3360
3361
3362
3363
3364
3365
3366
3367
3368
3369
3370
3371
3372
3373
3374
3375
3376
3377
3378
3379
3380
3381
3382
3383
3384
3385
3386
3387
3388
3389
3390
3391
3392
3393
3394
3395
3396
3397
3398
3399
3400
3401
3402
3403
3404
3405
3406
3407
3408
3409
3410
3411
3412
3413
3414
3415
3416
3417
3418
3419
3420
3421
3422
3423
3424
3425
3426
3427
3428
3429
3430
3431
3432
3433
3434
3435
3436
3437
3438
3439
3440
3441
3442
3443
3444
3445
3446
3447
3448
3449
3450
3451
3452
3453
3454
3455
3456
3457
3458
3459
3460
3461
3462
3463
3464
3465
3466
3467
3468
3469
3470
3471
3472
3473
3474
3475
3476
3477
3478
3479
3480
3481
3482
3483
3484
3485
3486
3487
3488
3489
3490
3491
3492
3493
3494
3495
3496
3497
3498
3499
3500
3501
3502
3503
3504
3505
3506
3507
3508
3509
3510
3511
3512
3513
3514
3515
3516
3517
3518
3519
3520
3521
3522
3523
3524
3525
3526
3527
3528
3529
3530
3531
3532
3533
3534
3535
3536
3537
3538
3539
3540
3541
3542
3543
3544
3545
3546
3547
3548
3549
3550
3551
3552
3553
3554
3555
3556
3557
3558
3559
3560
3561
3562
3563
3564
3565
3566
3567
3568
3569
3570
3571
3572
3573
3574
3575
3576
3577
3578
3579
3580
3581
3582
3583
3584
3585
3586
3587
3588
3589
3590
3591
3592
3593
3594
3595
3596
3597
3598
3599
3600
3601
3602
3603
3604
3605
3606
3607
3608
3609
3610
3611
3612
3613
3614
3615
3616
3617
3618
3619
3620
3621
3622
3623
3624
3625
3626
3627
3628
3629
3630
3631
3632
3633
3634
3635
3636
3637
3638
3639
3640
3641
3642
3643
3644
3645
3646
3647
3648
3649
3650
3651
3652
3653
3654
3655
3656
3657
3658
3659
3660
3661
3662
3663
3664
3665
3666
3667
3668
3669
3670
3671
3672
3673
3674
3675
3676
3677
3678
3679
3680
3681
3682
3683
3684
3685
3686
3687
3688
3689
3690
3691
3692
3693
3694
3695
3696
3697
3698
3699
3700
3701
3702
3703
3704
3705
3706
3707
3708
3709
3710
3711
3712
3713
3714
3715
3716
3717
3718
3719
3720
3721
3722
3723
3724
3725
3726
3727
3728
3729
3730
3731
3732
3733
3734
3735
3736
3737
3738
3739
3740
3741
3742
3743
3744
3745
3746
3747
3748
3749
3750
3751
3752
3753
3754
3755
3756
3757
3758
3759
3760
3761
3762
3763
3764
3765
3766
3767
3768
3769
3770
3771
3772
3773
3774
3775
3776
3777
3778
3779
3780
3781
3782
3783
3784
3785
3786
3787
3788
3789
3790
3791
3792
3793
3794
3795
3796
3797
3798
3799
3800
3801
3802
3803
3804
3805
3806
3807
3808
3809
3810
3811
3812
3813
3814
3815
3816
3817
3818
3819
3820
3821
3822
3823
3824
3825
3826
3827
3828
3829
3830
3831
3832
3833
3834
3835
3836
3837
3838
3839
3840
3841
3842
3843
3844
3845
3846
3847
3848
3849
3850
3851
3852
3853
3854
3855
3856
3857
3858
3859
3860
3861
3862
3863
3864
3865
3866
3867
3868
3869
3870
3871
3872
3873
3874
3875
3876
3877
3878
3879
3880
3881
3882
3883
3884
3885
3886
3887
3888
3889
3890
3891
3892
3893
3894
3895
3896
3897
3898
3899
3900
3901
3902
3903
3904
3905
3906
3907
3908
3909
3910
3911
3912
3913
3914
3915
3916
3917
3918
3919
3920
3921
3922
3923
3924
3925
3926
3927
3928
3929
3930
3931
3932
3933
3934
3935
3936
3937
3938
3939
3940
3941
3942
3943
3944
3945
3946
3947
3948
3949
3950
3951
3952
3953
3954
3955
3956
3957
3958
3959
3960
3961
3962
3963
3964
3965
3966
3967
3968
3969
3970
3971
3972
3973
3974
3975
3976
3977
3978
3979
3980
3981
3982
3983
3984
3985
3986
3987
3988
3989
3990
3991
3992
3993
3994
3995
3996
3997
3998
3999
4000
4001
4002
4003
4004
4005
4006
4007
4008
4009
4010
4011
4012
4013
4014
4015
4016
4017
4018
4019
4020
4021
4022
4023
4024
4025
4026
4027
4028
4029
4030
4031
4032
4033
4034
4035
4036
4037
4038
4039
4040
4041
4042
4043
4044
4045
4046
4047
4048
4049
4050
4051
4052
4053
4054
4055
4056
4057
4058
4059
4060
4061
4062
4063
4064
4065
4066
4067
4068
4069
4070
4071
4072
4073
4074
4075
4076
4077
4078
4079
4080
4081
4082
4083
4084
4085
4086
4087
4088
4089
4090
4091
4092
4093
4094
4095
4096
4097
4098
4099
4100
4101
4102
4103
4104
4105
4106
4107
4108
4109
4110
4111
4112
4113
4114
4115
4116
4117
4118
4119
4120
4121
4122
4123
4124
4125
4126
4127
4128
4129
4130
4131
4132
4133
4134
4135
4136
4137
4138
4139
4140
4141
4142
4143
4144
4145
4146
4147
4148
4149
4150
4151
4152
4153
4154
4155
4156
4157
4158
4159
4160
4161
4162
4163
4164
4165
4166
4167
4168
4169
4170
4171
4172
4173
4174
4175
4176
4177
4178
4179
4180
4181
4182
4183
4184
4185
```

```

LOCATION OBJECT CODE LINE SOURCE LINE
05DF 03 IMC BC ;INCREMENT COLOR POINTER
05E0 E1 POP HL ;GET ITEM COUNTER
05E1 28 DEC HL
05E2 7C LD A,H
05E3 05 OR L
05E4 200C JR NZ,NEXT_COLOR
05E6 1BCD JR O B,RET
1893 ;Internal rout to update to next VRAM index screen area
1894 SUP_UPDATE EQU $
1895 C5 <05E0> PUSH BC
1896 010100 LD LD BC,100H
1897 EB EX DE,HL
1898 09 ADD HL,BC
1899 EB EX DE,HL
1900 C1 POP BC
1901 C9 RET
1902 ;
1903 ACT_MOBILE EQU $
1904 C00572 <05F1> SUBCASE Mobile INIT_XP_OS ;X_PAT_POS := BOH
1905 CALL
1906 ; INSERT NEW_GENERATOR ADDRESS IN OBJECT_CRAM
1907 IMC DE
1908 13 LD A,[DE]
1909 1A LD LD [(Y+5),A]
1910 13 IMC DE
1911 1A LD A,[DE]
1912 FB FD7706 LD LD [(Y+6),A]
1913 F1 POP AF
1914 C9 RET
1915 ACT_DSPRT EQU $
1916 ; SUBCASE Sprite size 0
1917 ACT_DSPRT EQU $
1918 ; SUBCASE Sprite size 1
1919 IMC BC
1920 03 IMC BC
1921 03 IMC BC
1922 03 IMC BC
1923 03 IMC BC
1924 EB EX DE,HL
1925 23 IMC HL
1926 7E LD A,[HL]
1927 5F LD E,A
1928 1600 LD D,0
1929 05 PUSH DE
1930 23 IMC HL
1931 5E LD E,[HL]
1932 23 IMC HL
1933 56 LD D,[HL]
1934 23 IMC HL
1935 06 ADD A,[HL]
1936 02 LD LD [(BC),A]
1937 4E LD C,[HL]
1938 06 LD B,0
1939 C5 POP BC
1940 EB POP
1941 EB EX DE,HL
1942 01 POP DE
0600 03 ;->NEXT_GEN IN CRAM
0601 03 IMC BC
0602 03 IMC BC
0603 03 IMC BC
0604 03 IMC BC
0605 EB EX DE,HL ;HL->FIRST_GEN_NAME
0606 23 IMC HL
0607 7E LD A,[HL]
0608 5F LD E,A ;SV INDEX TO VRAM
0609 1600 LD D,0
0608 05 PUSH DE ;DE=PTRN_PTR
060C 23 IMC HL
060D 5E LD E,[HL]
060E 23 IMC HL
060F 56 LD D,[HL]
0610 23 IMC HL
0611 06 ADD A,[HL]
0612 02 LD LD [(BC),A]
0613 4E LD C,[HL]
0614 0600 LD B,0 ;CALC & SET NEXT_GEN CRAM
0616 C5 POP BC
0617 FDE1 POP
0619 EB EX DE,HL ;HL->SOURCE PTRN GEN
061A 01 POP DE ;DE=INDEX TO PTRN GEN VRAM
    
```

LOCATION OBJECT CODE LINE SOURCE LINE

0618 F1	1943	POP	AF
061C D0	1944	RET	MC
061D 3E01	1945	LD	A,1
061F CD1C27	1946	CALL	PUT_VRAM_
0622 C9	1947	RET	
	1948		
	1949	PROG	

;SIGNAL SPRITE PRIM GEN FILL

LOCATION OBJECT CODE LINE SOURCE LINE

```

1951 ***** PUTOBJ *****
1952 ;DESCRIPTION: PUTOBJ VECTORS TO ONE OF 5 SPECIFIC ROUTINES FOR PLACING THE
1953 ; DIFFERENT OBJECT TYPES ON THE DISPLAY
1954 ; IX = ADDRESS OF OBJECT TO BE PROCESSED
1955 ; INPUT:
1956 ; B = PARAMETER TO BE PASSED SPECIFIC PUT ROUTINES
1957
1958 * IN ADDITION, THIS MODULE CONTAINS ROUTINES WHICH ALLOW VRAM OPERATIONS
1959 * TO BE DEFERRED, TYPICALLY UNTIL AN INTERRUPT OCCURS, AND PERFORMED
1960 * IN A BLOCK BY A CENTRAL WRITER ROUTINE.
1961 *****
1962
1963 DATA
1964 QUEUE_SIZE DEFS 1
1965 * THIS IS THE SIZE OF THE DEFERRED WRITE QUEUE. IT IS SET BY THE
1966 * CARTRIDGE PROGRAMMER. IT HAS RANGE 0 - 255.
1967
1968 QUEUE_HEAD DEFS 1
1969 QUEUE_TAIL DEFS 1
1970 * THESE ARE THE INDICES OF THE HEAD AND TAIL OF THE WRITE QUEUE.
1971
1972 HEAD_ADDRESS DEFS 2
1973 TAIL_ADDRESS DEFS 2
1974 * THESE ARE THE ADDRESSES OF THE QUEUE HEAD AND TAIL
1975
1976 ;TRUE EQU 1
1977 ;FALSE EQU 0
1978 * VALUES FOR BOOLEAN DEFERAL_FLAG
1979
1980 BUFFER DEFS 2
1981 * THIS IS A POINTER TO THE BEGINNING OF THE DEFERRED WRITE QUEUE. THE
1982 * CARTRIDGE PROGRAMMER IS RESPONSIBLE FOR PROVIDING A RAM AREA TO HOLD
1983 * THE QUEUE, AND PASSING ITS LOCATION AND SIZE TO INIT_QUEUE.
1984
1985 ; COMMON
1986 ;PARAM_AREA DEFS 3
1987 * PARAM_AREA IS THE COMMON PARAMETER PASSING AREA FOR PASCAL ENTRY PTS
1988
1989
1990 PROG
1991 SET_UP_WRITE EQU $
1992
1993 * SET_UP_WRITE SETS UP A DEFERRED VRAM OPERATION.
1994
1995 * PUT DATA AT QUEUE_HEAD
1996
1997 PUSH IX
1998 LD HL,(HEAD_ADDRESS)
1999 POP DE
2000 LD (HL),E ; PUT DATA POINTER
2001 HL
2002 LD (HL),D
2003 HL
2004 LD (HL),B ;STORE PUTOBJ PARAMETER
2005 HL
2006 EX DE,HL ; HEAD ADDRESS IN DE
2007 * INCREMENT QUEUE_HEAD

```

<0623>

0623 D0E5
0625 2A73CD
0628 D1
0629 73
062A 23
062B 72
062C 23
062D 70
062E 23
062F EB

```

LOCATION OBJECT CODE LINE SOURCE LINE
0630 3A73CB 2008 LD A,[QUEUE_HEAD]
0633 3C 2009 INC A ; NEW HEAD IN A
2010
2011 * IF QUEUE_HEAD = QUEUE_SIZE THEN
0634 2173CA 2012 LD HL,QUEUE_SIZE
0637 BE 2013 CP (HL)
0638 2000 2014 JR NZ,NOT_TOO_BIG
2015
2016 * QUEUE_HEAD := 0
063A 3E00 2017 LD A,0
063C 3273CB 2018 LD [QUEUE_HEAD],A
2019
2020 * HEAD_ADDRESS := BUFFER
063F 2A7301 2021 LD HL,[BUFFER]
0642 2273CD 2022 LD [HEAD_ADDRESS],HL
2023
0645 1807 2024 JR SET_UP_ENDIF
2025 * ELSE
2026 NOT_TOO_BIG EQU $
2027
2028 * STORE NEW QUEUE_HEAD
0647 3273CB 2029 LD [QUEUE_HEAD],A
2030
2031 * STORE HEAD ADDRESS
064A ED5373CD 2032 LD [HEAD_ADDRESS],DE
2033
2034 * END IF
2035 SET_UP_ENDIF
2036
064E 2037 * END SET_UP_WRITE
064E C9 2038 RET
2039
2040 * PROCEDURE INIT_QUEUE (SIZE:BYTE;VAR A_QUEUE:QUEUE)
2041
2042 * SIZE PASSED IN A, LOCATION PASSED IN HL
2043 * DESTROYS: A
2044
064F 00020001 2045 INIT_QUEUE_P DEFM 2,1,-2
0653 FFFE 2046 * THIS IS THE PARAMETER DESCRIPTOR FOR INIT_QUEUEQ
2047
2048 * BEGIN INIT_QUEUE
2049 INIT_QUEUEQ
2050 INIT_QUEUEQ EQU $
2051 BC,INIT_QUEUE_P
2052 DE,PARAM_AREA
2053 PARAM
2054 A,[PARAM_AREA]
2055 HL,[PARAM_AREA+1]
2056
2057 GLB INIT_QUEUE
2058 INIT_QUEUE EQU $
2059
2060 * QUEUE_SIZE := SIZE
2061 LD [QUEUE_SIZE],A
2062
2063 * OUTPUT HEAD := OUTPUT TAIL ;...

```

```

LOCATION OBJECT CODE LINE SOURCE LINE
0667 3E00 2064 LD A,0
0669 3273CB 2065 LD [QUEUE_HEAD],A
066C 3273CC 2066 LD [QUEUE_TAIL],A
066F 227301 2068 * BUFFER := TAIL_ADDRESS := HEAD_ADDRESS := LOCATION
0672 22730D 2069 LD [BUFFER],HL
0675 2273CF 2070 LD [HEAD_ADDRESS],HL
2071 LD [TAIL_ADDRESS],HL
2072
2073 * END INIT_QUEUE
2074 RET
2075
2076 * PROCEDURE WRITER_
2077
2078 * TAKES NO PARAMETERS
2079 * DESTROYS: ALL
2080
2081 * BEGIN WRITER_ GLB WRITER_
2082 WRITER_ $
2083 WRITER_ EQU
2084
2085 * SAVE DEFERAL FLAG
2066 LD A,[DEFER_WRITES]
2067 PUSH AF
2068
2069 * DEFER_WRITES := FALSE
2090 LD A,FALSE
2091 LD [DEFER_WRITES],A
2092
2093 * WHILE QUEUE_TAIL <> QUEUE_HEAD DO
2094 WRTR_WHILE EQU $
2095 LD A,[QUEUE_TAIL]
2096 HL,QUEUE_HEAD
2097 [HL]
2098 JR Z,WRTR_END_WHILE
2099
2100 * WRITE DATA AT QUEUE_TAIL TO VRAM
2101 HL,[TAIL_ADDRESS]
2102 LD E,[HL] ; GET OBJECT POINTER
2103 INC HL
2104 LD D,[HL]
2105 INC HL
2106 LD B,[HL] ; GET PARAMETER
2107 INC HL
2108
2109 * PROCESS OBJECT IN QUEUE
2110 DE
2111 PUSH IX
2112 POP HL
2113 CALL DO_PUTOBJ ; SAVE QUEUE TAIL ADDRESS
2114
2115 * INCREMENT QUEUE_TAIL
2116 LD A,[QUEUE_TAIL]
2117 INC A
2118
2119 * IF QUEUE_TAIL = QUEUE_SIZE THEN
2120 HL,QUEUE_SIZE

```

```

LOCATION OBJECT CODE LINE SOURCE LINE
06A2 BE 2121 CP (HL)
06A3 200E 2122 JR NZ,WRTR_ELSE
2123
2124 * QUEUE_TAIL := 0
2125 LD A,0
2126 LD (QUEUE_TAIL),A
2127
2128 * TAIL_ADDRESS := BUFFER
2129 LD HL,(BUFFER)
2130 LD (TAIL_ADDRESS),HL
06A4 E1 2131 POP HL ;RESTORE STACK POINTER
2132
2133 JR WRTR_END_IF
2134 * ELSE
2135 WRTR_ELSE EQU $
<06B3> 2136
2137 * STORE NEW QUEUE_TAIL
2138 LD (QUEUE_TAIL),A
2139
2140 * TAIL_ADDRESS := TAIL_ADDRESS + 3
2141 POP HL
06B6 E1 2142 LD (TAIL_ADDRESS),HL
06B7 2273CF 2143
2144 * END_IF
2145 WRTR_END_IF EQU $
<06BA> 2146
2147 JR WRTR_WHILE
06BA 18C6 2148 * END WHILE
2149 WRTR_END_WHILE EQU $
<06BC> 2150
2151 * RESTORE DEFERAL FLAG
2152 POP AF
06BC F1 2153 LD (DEFER_WRITES),A
06BD 3273C6 2154
2155 * END WRITER_
2156 RET
06C0 C9 2157
2158 GLB PUTOBJ_
2159
2160
2161 ;EXT PUTSEMI,PUT_MOBILE,PUTOSPRITE,PUT1SPRITE,PUTCOMPLEX
2162
2163 ;EXT DEFER_WRITES
2164 ;EXT PARAM_
2165 GLB PUTOBJJO DEFW 2,2,1
06C1 00020002 2166 PUTOBJ_PAR:
06C5 0001 2167
2168 * PROCEDURE PUT_OBJP (VAR DATA:BUFFER;PARAM:BYTE);
2169
2170 * THIS IS THE PASCAL ENTRY POINT TO THE PUTOBJ ROUTINE
2171
2172 PROG
2173 PUTOBJJO:
2174 LD BC,PUTOBJ_PAR
06C7 0106C1 2175 LD DE,PARAM_AREA
06CA 1173BA 2176 (A)
06CC 9B (A)

```

LOCATION	OBJECT CODE	LINE	SOURCE LINE
0600	D02A73BA	2177	LD IX, (PARAM_AREA1)
0604	3A73BC	2178	LD A, (PARAM_AREA+2)
0607	47	2179	LD B, A
2180			
2181	DEFER		EQU 1
2182	PUTOBJ		
2183	A, (DEFER_WRITES)		; CHECK IF DEFERRED WRITE IS DESIRED
2184	CP DEFER		
2185	JR NZ, DO_PUTOBJ		; IF NOT, PROCESS OBJECT
2186	CALL SET_UP_WRITE		; IF SO, SET UP FOR DEFERRED WRITE
2187	RET		
2188	DO_PUTOBJ		; GET ADDRESS OF GRAPHICS FOR OBJ_n
2189	LD L, (IX+0)		
2190	LD A, (HL)		; A = OBJ TYPE
2191	LD C, A		; SAVE COPY
2192	AND OFH		; MASK FOR OBJ TYPE NUMBER
2193	JP Z, PUTSEMI		; 0 = SEMI_MOBILE
2194	DEC A		
2195	JP Z, PUT_MOBILE		; 1 = MOBILE
2196	DEC A		
2197	JP Z, PUTSPRITE		; 2 = SPRITE
2198	DEC A		
2199	JP Z, PUT1SPRITE		; 3 = SPRITE1
2200	JP PUTCOMPLEX		; >3 = COMPLEX
2201	:		
2202	:		END ; pname
2203	:		PROG

```

LOCATION OBJECT CODE LINE SOURCE LINE
2205 ***** PUT_SEMI *****
2206 ;DESCRIPTION: PUTS SEMI_MOBILE OBJECTS ON SCREEN *****
2207 ;
2208 ;
2209 ; INPUT: IX = ADDRESS OF OBJECT TO BE PROCESSED
2210 ; HL = ADDRESS OF OBJECT'S GRAPHICS TABLES IN ROM
2211 *****
2212 GLB PUTSEMI
2213
2214
2215
2216 PUTSEMI: LD D, [IX+3] ;GET ADDRESS OF STATUS
2217 LD E, [IX+2] ;
2218 PUSH DE ; AND PUT INTO IX
2219 POP IX ;
2220 LD D, [IY+2] ;GET X_LOCATION
2221 LD E, [IY+1]
2222 CALL PX_TO_PTRN_POS
2223
2224 LD C,E ;C := PATTERN PLANE COL.
2225 LD D, [IY+4] ;GET Y_LOCATION
2226 LD E, [IY+3]
2227 CALL PX_TO_PTRN_POS
2228
2229 LD B,E ;B := PATTERN PLANE ROW
2230 LD E, [IY+0] ;GET FRAME NUMBER
2231
2232 ; HL = GRAPHICS_n, IX = OBJ_n, IY = STATUS_n, C = COL., B = ROW, E = FRAME
2233 ;
2234 LD D,0 ;DE HAS FRAME NUMBER
2235 ADD HL,DE ;2*FRAME NUMBER + ADDR OF GRAPHICS_n
2236 ADD HL,DE ;FRAME POINTER OFFSET
2237 LD E,5 ;HL NOW POINTS TO LOCATION HOLDING ADDRESS
2238 ADD HL,DE ;OF FRAME
2239 ;GET ADDRESS INTO DE
2240 LD E, [HL]
2241 TMC HL ;HL := ADDRESS OF FRAME
2242 LD D, [HL]
2243 EX DE, HL
2244 PUSH BC
2245 POP DE
2246 LD C, [HL]
2247 TMC HL ;DE := Y PAT POS & X PAT_POS
2248 LD B, [HL] ;C := X_EXTENT
2249 TMC HL ;B := Y_EXTENT
2250 ;HL POINTS TO FIRST NAME IN LIST
2251 ; TEST TO SEE IF OLD_SCREEN IS TO BE SAVED
2252 ;
2253 LD A, [IX+5] ;GET HIGH BYTE OF OLD_SCREEN ADDRESS
2254 BIT 7,A ;TEST BIT 15 OF OLD_SCREEN ADDRESS
2255 JR Z,S_OLD_SCRN
2256 ;
2257 CALL PUTFRAME
2258 RET
2259
2260 S_OLD_SCRN
0730
0731
0732
0733
0734
0735
0736
0737
0738
0739
0740
0741
0742
0743
0744
0745
0746
0747
0748
0749
0750
0751
0752
0753
0754
0755
0756
0757
0758
0759
0760
0761
0762
0763
0764
0765
0766
0767
0768
0769
0770
0771
0772
0773
0774
0775
0776
0777
0778
0779
0780
0781
0782
0783
0784
0785
0786
0787
0788
0789
0790
0791
0792
0793
0794
0795
0796
0797
0798
0799
0800
0801
0802
0803
0804
0805
0806
0807
0808
0809
0810
0811
0812
0813
0814
0815
0816
0817
0818
0819
0820
0821
0822
0823
0824
0825
0826
0827
0828
0829
0830
0831
0832
0833
0834
0835
0836
0837
0838
0839
0840
0841
0842
0843
0844
0845
0846
0847
0848
0849
0850
0851
0852
0853
0854
0855
0856
0857
0858
0859
0860
0861
0862
0863
0864
0865
0866
0867
0868
0869
0870
0871
0872
0873
0874
0875
0876
0877
0878
0879
0880
0881
0882
0883
0884
0885
0886
0887
0888
0889
0890
0891
0892
0893
0894
0895
0896
0897
0898
0899
0900
0901
0902
0903
0904
0905
0906
0907
0908
0909
0910
0911
0912
0913
0914
0915
0916
0917
0918
0919
0920
0921
0922
0923
0924
0925
0926
0927
0928
0929
0930
0931
0932
0933
0934
0935
0936
0937
0938
0939
0940
0941
0942
0943
0944
0945
0946
0947
0948
0949
0950
0951
0952
0953
0954
0955
0956
0957
0958
0959
0960
0961
0962
0963
0964
0965
0966
0967
0968
0969
0970
0971
0972
0973
0974
0975
0976
0977
0978
0979
0980
0981
0982
0983
0984
0985
0986
0987
0988
0989
0990
0991
0992
0993
0994
0995
0996
0997
0998
0999
1000

```