

LOCATION OBJECT CODE LINE SOURCE LINE

```

5973 ;TRUE
5974 EQU 1
5975 ;EXT VRAM_WRITE,REG_WRITE,VRAM_READ
5976 ;EXT VDP_MODE_WORD
5977 ;EXT MUX_SPRITES
5978 ;EXT PARAM
5979 ;EXT LOCAL_SPR_TBL_SPRITE_ORDER
5980 GLB INIT_TABLE,GET_VRAM,PUT_VRAM,INIT_SPR_ORDER,WR_SPR_NH_TBL
5981 GLB INIT_TABLE,GET_VRAMQ,PUT_VRAMQ,INIT_SPR_ORDERQ,WR_SPR_NH_TBLQ
5982
5983 * PROCEDURE INIT_TABLE (TABLE_CODE:BYTE;TABLE_ADDRESS:INTEGER)
5984
5985 * THIS IS THE PASCAL ENTRY POINT TO INIT_TABLE
5986
5987 INIT_TABLE_P DEFM 2,1,2
5988
5989 * THIS IS THE PARAMETER DESCRIPTOR FOR INIT_TABLEQ
5990
5991 INIT_TABLEQ
5992 LD BC,INIT_TABLE_P
5993 LD DE,PARAM_AREA
5994 CALL PARAM
5995 LD A,[PARAM_AREA]
5996 HL,[PARAM_AREA+1]
5997
5998 INIT_TABLE
5999 ;INIT TABLE INITIALIZES THE TABLE ADDRESSES
6000 ;FOR VRAM TABLES. IT ALSO WRITES THE APPROPRIATE
6001 ;BASE ADDRESS INTO THE RESPECTIVE VDP REGISTER.
6002 ;
6003 ;IN: VRAM ADDRESS IN HL
6004 ;TABLE CODE IN 'A' : 0=SPRITE NAME TABLE
6005 ; 1=SPRITE GENERATOR TABLE, 2=PATTERN NAME
6006 ; 3= PATTERN GENERATOR TABLE, 4=
6007 ; COLOR TABLE
6008
6009 LD C,A
6010 LD B,0
6011 ;USE TABLE CODE AS INDEX
6012 LD IX,VRAM_ADDR_TABLE
6013 ADD IX,BC
6014 ADD IX,BC
6015 LD [IX+0],L
6016 LD [IX+1],H
6017 ;SAVE VRAM ADDRESS IN TABLE
6018 ;CHECK VDP GRAPHICS MODE
6019 LD A,[VDP_MODE_WORD]
6020 BIT 1,A
6021 JR Z,INIT_TABLE80
6022 ;** GRAPHICS MODE 2
6023 LD A,C
6024 CP 3
6025 ;GET TABLE CODE
6026 ;CHECK IF GENERATOR TABLE
6027 CP 4
6028 ;CHECK IF COLOR TABLE
6029 JR Z,CASE_OF_COLOR
6030 JR INIT_TABLE80
6031
6032 CASE_OF_GEN
6033 LD B,4
6034 LD A,L
6035 OR H
6036
6037
6038
6039
6040
6041
6042
6043
6044
6045
6046
6047
6048
6049
6050
6051
6052
6053
6054
6055
6056
6057
6058
6059
6060
6061
6062
6063
6064
6065
6066
6067
6068
6069
6070
6071
6072
6073
6074
6075
6076
6077
6078
6079
6080
6081
6082
6083
6084
6085
6086
6087
6088
6089
6090
6091
6092
6093
6094
6095
6096
6097
6098
6099
6100
6101
6102
6103
6104
6105
6106
6107
6108
6109
6110
6111
6112
6113
6114
6115
6116
6117
6118
6119
6120
6121
6122
6123
6124
6125
6126
6127
6128
6129
6130
6131
6132
6133
6134
6135
6136
6137
6138
6139
6140
6141
6142
6143
6144
6145
6146
6147
6148
6149
6150
6151
6152
6153
6154
6155
6156
6157
6158
6159
6160
6161
6162
6163
6164
6165
6166
6167
6168
6169
6170
6171
6172
6173
6174
6175
6176
6177
6178
6179
6180
6181
6182
6183
6184
6185
6186
6187
6188
6189
6190
6191
6192
6193
6194
6195
6196
6197
6198
6199
6200
6201
6202
6203
6204
6205
6206
6207
6208
6209
6210
6211
6212
6213
6214
6215
6216
6217
6218
6219
6220
6221
6222
6223
6224
6225
6226
6227
6228
6229
6230
6231
6232
6233
6234
6235
6236
6237
6238
6239
6240
6241
6242
6243
6244
6245
6246
6247
6248
6249
6250
6251
6252
6253
6254
6255
6256
6257
6258
6259
6260
6261
6262
6263
6264
6265
6266
6267
6268
6269
6270
6271
6272
6273
6274
6275
6276
6277
6278
6279
6280
6281
6282
6283
6284
6285
6286
6287
6288
6289
6290
6291
6292
6293
6294
6295
6296
6297
6298
6299
6300
6301
6302
6303
6304
6305
6306
6307
6308
6309
6310
6311
6312
6313
6314
6315
6316
6317
6318
6319
6320
6321
6322
6323
6324
6325
6326
6327
6328
6329
6330
6331
6332
6333
6334
6335
6336
6337
6338
6339
6340
6341
6342
6343
6344
6345
6346
6347
6348
6349
6350
6351
6352
6353
6354
6355
6356
6357
6358
6359
6360
6361
6362
6363
6364
6365
6366
6367
6368
6369
6370
6371
6372
6373
6374
6375
6376
6377
6378
6379
6380
6381
6382
6383
6384
6385
6386
6387
6388
6389
6390
6391
6392
6393
6394
6395
6396
6397
6398
6399
6400
6401
6402
6403
6404
6405
6406
6407
6408
6409
6410
6411
6412
6413
6414
6415
6416
6417
6418
6419
6420
6421
6422
6423
6424
6425
6426
6427
6428
6429
6430
6431
6432
6433
6434
6435
6436
6437
6438
6439
6440
6441
6442
6443
6444
6445
6446
6447
6448
6449
6450
6451
6452
6453
6454
6455
6456
6457
6458
6459
6460
6461
6462
6463
6464
6465
6466
6467
6468
6469
6470
6471
6472
6473
6474
6475
6476
6477
6478
6479
6480
6481
6482
6483
6484
6485
6486
6487
6488
6489
6490
6491
6492
6493
6494
6495
6496
6497
6498
6499
6500
6501
6502
6503
6504
6505
6506
6507
6508
6509
6510
6511
6512
6513
6514
6515
6516
6517
6518
6519
6520
6521
6522
6523
6524
6525
6526
6527
6528
6529
6530
6531
6532
6533
6534
6535
6536
6537
6538
6539
6540
6541
6542
6543
6544
6545
6546
6547
6548
6549
6550
6551
6552
6553
6554
6555
6556
6557
6558
6559
6560
6561
6562
6563
6564
6565
6566
6567
6568
6569
6570
6571
6572
6573
6574
6575
6576
6577
6578
6579
6580
6581
6582
6583
6584
6585
6586
6587
6588
6589
6590
6591
6592
6593
6594
6595
6596
6597
6598
6599
6600
6601
6602
6603
6604
6605
6606
6607
6608
6609
6610
6611
6612
6613
6614
6615
6616
6617
6618
6619
6620
6621
6622
6623
6624
6625
6626
6627
6628
6629
6630
6631
6632
6633
6634
6635
6636
6637
6638
6639
6640
6641
6642
6643
6644
6645
6646
6647
6648
6649
6650
6651
6652
6653
6654
6655
6656
6657
6658
6659
6660
6661
6662
6663
6664
6665
6666
6667
6668
6669
6670
6671
6672
6673
6674
6675
6676
6677
6678
6679
6680
6681
6682
6683
6684
6685
6686
6687
6688
6689
6690
6691
6692
6693
6694
6695
6696
6697
6698
6699
6700
6701
6702
6703
6704
6705
6706
6707
6708
6709
6710
6711
6712
6713
6714
6715
6716
6717
6718
6719
6720
6721
6722
6723
6724
6725
6726
6727
6728
6729
6730
6731
6732
6733
6734
6735
6736
6737
6738
6739
6740
6741
6742
6743
6744
6745
6746
6747
6748
6749
6750
6751
6752
6753
6754
6755
6756
6757
6758
6759
6760
6761
6762
6763
6764
6765
6766
6767
6768
6769
6770
6771
6772
6773
6774
6775
6776
6777
6778
6779
6780
6781
6782
6783
6784
6785
6786
6787
6788
6789
6790
6791
6792
6793
6794
6795
6796
6797
6798
6799
6800
6801
6802
6803
6804
6805
6806
6807
6808
6809
6810
6811
6812
6813
6814
6815
6816
6817
6818
6819
6820
6821
6822
6823
6824
6825
6826
6827
6828
6829
6830
6831
6832
6833
6834
6835
6836
6837
6838
6839
6840
6841
6842
6843
6844
6845
6846
6847
6848
6849
6850
6851
6852
6853
6854
6855
6856
6857
6858
6859
6860
6861
6862
6863
6864
6865
6866
6867
6868
6869
6870
6871
6872
6873
6874
6875
6876
6877
6878
6879
6880
6881
6882
6883
6884
6885
6886
6887
6888
6889
6890
6891
6892
6893
6894
6895
6896
6897
6898
6899
6900
6901
6902
6903
6904
6905
6906
6907
6908
6909
6910
6911
6912
6913
6914
6915
6916
6917
6918
6919
6920
6921
6922
6923
6924
6925
6926
6927
6928
6929
6930
6931
6932
6933
6934
6935
6936
6937
6938
6939
6940
6941
6942
6943
6944
6945
6946
6947
6948
6949
6950
6951
6952
6953
6954
6955
6956
6957
6958
6959
6960
6961
6962
6963
6964
6965
6966
6967
6968
6969
6970
6971
6972
6973
6974
6975
6976
6977
6978
6979
6980
6981
6982
6983
6984
6985
6986
6987
6988
6989
6990
6991
6992
6993
6994
6995
6996
6997
6998
6999
7000
7001
7002
7003
7004
7005
7006
7007
7008
7009
7010
7011
7012
7013
7014
7015
7016
7017
7018
7019
7020
7021
7022
7023
7024
7025
7026
7027
7028
7029
7030
7031
7032
7033
7034
7035
7036
7037
7038
7039
7040
7041
7042
7043
7044
7045
7046
7047
7048
7049
7050
7051
7052
7053
7054
7055
7056
7057
7058
7059
7060
7061
7062
7063
7064
7065
7066
7067
7068
7069
7070
7071
7072
7073
7074
7075
7076
7077
7078
7079
7080
7081
7082
7083
7084
7085
7086
7087
7088
7089
7090
7091
7092
7093
7094
7095
7096
7097
7098
7099
7100
7101
7102
7103
7104
7105
7106
7107
7108
7109
7110
7111
7112
7113
7114
7115
7116
7117
7118
7119
7120
7121
7122
7123
7124
7125
7126
7127
7128
7129
7130
7131
7132
7133
7134
7135
7136
7137
7138
7139
7140
7141
7142
7143
7144
7145
7146
7147
7148
7149
7150
7151
7152
7153
7154
7155
7156
7157
7158
7159
7160
7161
7162
7163
7164
7165
7166
7167
7168
7169
7170
7171
7172
7173
7174
7175
7176
7177
7178
7179
7180
7181
7182
7183
7184
7185
7186
7187
7188
7189
7190
7191
7192
7193
7194
7195
7196
7197
7198
7199
7200
7201
7202
7203
7204
7205
7206
7207
7208
7209
7210
7211
7212
7213
7214
7215
7216
7217
7218
7219
7220
7221
7222
7223
7224
7225
7226
7227
7228
7229
7230
7231
7232
7233
7234
7235
7236
7237
7238
7239
7240
7241
7242
7243
7244
7245
7246
7247
7248
7249
7250
7251
7252
7253
7254
7255
7256
7257
7258
7259
7260
7261
7262
7263
7264
7265
7266
7267
7268
7269
7270
7271
7272
7273
7274
7275
7276
7277
7278
7279
7280
7281
7282
7283
7284
7285
7286
7287
7288
7289
7290
7291
7292
7293
7294
7295
7296
7297
7298
7299
7300
7301
7302
7303
7304
7305
7306
7307
7308
7309
7310
7311
7312
7313
7314
7315
7316
7317
7318
7319
7320
7321
7322
7323
7324
7325
7326
7327
7328
7329
7330
7331
7332
7333
7334
7335
7336
7337
7338
7339
7340
7341
7342
7343
7344
7345
7346
7347
7348
7349
7350
7351
7352
7353
7354
7355
7356
7357
7358
7359
7360
7361
7362
7363
7364
7365
7366
7367
7368
7369
7370
7371
7372
7373
7374
7375
7376
7377
7378
7379
7380
7381
7382
7383
7384
7385
7386
7387
7388
7389
7390
7391
7392
7393
7394
7395
7396
7397
7398
7399
7400
7401
7402
7403
7404
7405
7406
7407
7408
7409
7410
7411
7412
7413
7414
7415
7416
7417
7418
7419
7420
7421
7422
7423
7424
7425
7426
7427
7428
7429
7430
7431
7432
7433
7434
7435
7436
7437
7438
7439
7440
7441
7442
7443
7444
7445
7446
7447
7448
7449
7450
7451
7452
7453
7454
7455
7456
7457
7458
7459
7460
7461
7462
7463
7464
7465
7466
7467
7468
7469
7470
7471
7472
7473
7474
7475
7476
7477
7478
7479
7480
7481
7482
7483
7484
7485
7486
7487
7488
7489
7490
7491
7492
7493
7494
7495
7496
7497
7498
7499
7500
7501
7502
7503
7504
7505
7506
7507
7508
7509
7510
7511
7512
7513
7514
7515
7516
7517
7518
7519
7520
7521
7522
7523
7524
7525
7526
7527
7528
7529
7530
7531
7532
7533
7534
7535
7536
7537
7538
7539
7540
7541
7542
7543
7544
7545
7546
7547
7548
7549
7550
7551
7552
7553
7554
7555
7556
7557
7558
7559
7560
7561
7562
7563
7564
7565
7566
7567
7568
7569
7570
7571
7572
7573
7574
7575
7576
7577
7578
7579
7580
7581
7582
7583
7584
7585
7586
7587
7588
7589
7590
7591
7592
7593
7594
7595
7596
7597
7598
7599
7600
7601
7602
7603
7604
7605
7606
7607
7608
7609
7610
7611
7612
7613
7614
7615
7616
7617
7618
7619
7620
7621
7622
7623
7624
7625
7626
7627
7628
7629
7630
7631
7632
7633
7634
7635
7636
7637
7638
7639
7640
7641
7642
7643
7644
7645
7646
7647
7648
7649
7650
7651
7652
7653
7654
7655
7656
7657
7658
7659
7660
7661
7662
7663
7664
7665
7666
7667
7668
7669
7670
7671
7672
7673
7674
7675
7676
7677
7678
7679
7680
7681
7682
7683
7684
7685
7686
7687
7688
7689
7690
7691
7692
7693
7694
7695
7696
7697
7698
7699
7700
7701
7702
7703
7704
7705
7706
7707
7708
7709
7710
7711
7712
7713
7714
7715
7716
7717
7718
7719
7720
7721
7722
7723
7724
7725
7726
7727
7728
7729
7730
7731
7732
7733
7734
7735
7736
7737
7738
7739
7740
7741
7742
7743
7744
7745
7746
7747
7748
7749
7750
7751
7752
7753
7754
7755
7756
7757
7758
7759
7760
7761
7762
7763
7764
7765
7766
7767
7768
7769
7770
7771
7772
7773
7774
7775
7776
7777
7778
7779
7780
7781
7782
7783
7784
7785
7786
7787
7788
7789
7790
7791
7792
7793
7794
7795
7796
7797
7798
7799
7800
7801
7802
7803
7804
7805
7806
7807
7808
7809
7810
7811
7812
7813
7814
7815
7816
7817
7818
7819
7820
7821
7822
7823
7824
7825
7826
7827
7828
7829
7830
7831
7832
7833
7834
7835
7836
7837
7838
7839
7840
7841
7842
7843
7844
7845
7846
7847
7848
7849
7850
7851
7852
7853
7854
7855
7856
7857
7858
7859
7860
7861
7862
7863
7864
7865
7866
7867
7868
7869
7870
7871
7872
7873
7874
7875
7876
7877
7878
7879
7880
7881
7882
7883
7884
7885
7886
7887
7888
7889
7890
7891
7892
7893
7894
7895
7896
7897
7898
7899
7900
7901
7902
7903
7904
7905
7906
7907
7908
7909
7910
7911
7912
7913
7914
7915
7916
7917
7918
7919
7920
7921
7922
7923
7924
7925
7926
7927
7928
7929
7930
7931
7932
7933
7934
7935
7936
7937
7938
7939
7940
7941
7942
7943
7944
7945
7946
7947
7948
7949
7950
7951
7952
7953
7954
7955
7956
7957
7958
7959
7960
7961
7962
7963
7964
7965
7966
7967
7968
7969
7970
7971
7972
7973
7974
7975
7976
7977
7978
7979
7980
7981
7982
7983
7984
7985
7986
7987
7988
7989
7990
7991
7992
7993
7994
7995
7996
7997
7998
7999
8000
8001
8002
8003
8004
8005
8006
8007
8008
8009
8010
8011
8012
8013
8014
8015
8016
8017
8018
8019
8020
8021
8022
8023
8024
8025
8026
8027
8028
8029
8030
8031
8032
8033
8034
8035
8036
8037
8038
8039
8040
8041
8042
8043
8044
8045
8046
8047
8048
8049
8050
8051
8052
8053
8054
8055
8056
8057
8058
8059
8060
8061
8062
8063
8064
8065
8066
8067
8068
8069
8070
8071
8072
8073
8074
8075
8076
8077
8078
8079
8080
8081
8082
8083
8084
8085
8086
8087
8088
8089
8090
8091
8092
8093
8094
8095
8096
8097
8098
8099
8100
8101
8102
8103
8104
8105
8106
8107
8108
8109
8110
8111
8112
8113
8114
8115
8116
8117
8118
8119
8120
8121
8122
8123
8124
8125
8126
8127
8128
8129
8130
8131
8132
8133
8134
8135
8136
8137
8138
8139
8140
8141
8142
8143
8144
8145
8146
8147
8148
8149
8150
8151
8152
8153
8154
8155
8156
8157
8158
8159
8160
8161
8162
8163
8164
8165
8166
8167
8168
8169
8170
8171
8172
8173
8174
8175
8176
8177
8178
8179
8180
8181
8182
8183
8184
8185
8186
8187
8188
8189
8190
8191
8192
8193
8194
8195
8196
8197
8198
8199
8200
8201
8202
8203
8204
8205
8206
8207
8208
8209
8210
8211
8212
8213
8214
8215
8216
8217
8218
8219
8220
8221
8222
8223
8224
8225
8226
8227
8228
8229
8230
8231
8232
8233
8234
8235
8236
8237
8238
8239
8240
8241
8242
8243
8244
8245
8246
8247
8248
8249
8250
8251
8252
8253
8254
8255
8256
8257
8258
8259
8260
8261
8262
8263
8264
8265
8266
8267
8268
8269
8270
8271
8272
8273
8274
8275
8276
8277
8278
8279
8280
8281
8282
8283
8284
8285
8286
8287
8288
8289
8290
8291
8292
8293
8294
8295
8296
8297
8298
8299
8300
8301
8302
8303
8304
8305
8306
8307
8308
8309
8310
8311
8312
8313
83
```

```

LOCATION OBJECT CODE LINE SOURCE LINE
1844 2004 6029 JR N2_CASE_OF_GEN10
1846 0E03 6030 LD C,3 ;VALUE TO WRITE FOR VRAM ADDRESS OF 00H
1848 1828 6031 JR INIT_TABLE90
184A 6032 CASE_OF_GEN10
184A 0E07 6033 LD C,7 ;VALUE TO WRITE FOR VRAM ADDRESS OF 2000H
184C 1824 6034 JR INIT_TABLE90
184E 6035 CASE_OF_COLOR
184E 0603 6036 LD B,3 ;REGISTER TO WRITE
1850 7D 6037 LD A,L
1851 84 6038 OR H
1852 2004 6039 JR N2_CASE_OF_CLR10
1854 0E7F 6040 LD C,7FH ;VALUE TO WRITE FOR VRAM ADDRESS OF 00H
1856 181A 6041 JR INIT_TABLE90
1858 6042 CASE_OF_CLR10
1858 0EFF 6043 LD C,OFFH ;VALUE TO WRITE FOR VRAM ADDRESS OF 2000H
185A 1816 6044 JR INIT_TABLE90
185C 6045
185C 6046 INIT_TABLE80
185C 6047 ;**
185C 6048 ;**
185C 6049 LD IY,BASE_FACTORS
1860 FD211876 6050 ADD IY,BC
1862 FD09 6051 ADD IY,BC
1864 FD7E00 6052 LD A,[IY+0] ;SHIFT COUNT NOW IN 'A'
1867 FD4601 6053 LD B,[IY+1] ;REGISTER NUMBER TO WRITE IN 'B'
186A 6054 DIVIDE ;**
186A C83C 6055 SRL H ;SHIFT HI BYTE
186C C81D 6056 RR L ;SHIFT LO BYTE
186E 3D 6057 DEC A ;DECREMENT SHIFT COUNT
186F 20F9 6058 JR NZ,DIVIDE
1871 4D 6059 ;**
1871 4D 6060 LD C,L ;WRITE TO VDP REGISTER
1872 6061 INIT_TABLE90 ;VALUE TO WRITE IN 'C'
1872 CD1CCA 6062 CALL REG_WRITE
1875 C9 6063 RET
1876 6064
1876 6065 BASE_FACTORS ;BASE_FACTOR,REGISTER_NUMBER
1876 6066
1876 6067 DEFB 7,5,11,6,10,2,11,4,6,3
1878 0208040603 6068
1880 00050001 6069 * PROCEDURE GET_VRAMQ (TABLE CODE:BYTE;START_INDEX:BYTE;SLICE:BYTE;
1884 00010001 6070 * VAR DATA:BUFFER;ITEM_COUNT:INTEGER);
1888 FFFE0002 6071
1888 00050001 6072 * THIS IS THE PASCAL ENTRY POINT TO INIT_TABLE
1888 00010001 6073
1888 00050001 6074 GET_VRAM_P DEFW 5,1,1,1,-2,2
1888 00010001 6075 * THIS IS THE PARAMETER DESCRIPTION FOR INIT_TABLEQ
1888 00050001 6076
1888 00010001 6077 GET_VRAMQ
1888 00050001 6078 LD BC,GET_VRAM_P
1888 00010001 6079 LD DE,PARAM_AREA
1888 00050001 6080 CALL PARAM
1888 00010001 6081 LD A,[PARAM_AREA]
1888 00050001 6082 DE,[PARAM_AREA+1]
1888 00010001 6083

```

LOCATION	OBJECT CODE LINE	SOURCE LINE
189C FD2A738F	LD	1Y, [PARAM_AREA+5]
18A0 2A738D	LD	HL, [PARAM_AREA+3]
18A3	6087 GET_VRAM	
	6088	;GETS A CERTAIN NUMBER OF BYTES FROM VRAM
	6089	;AND PUTS THEM IN A BUFFER
	6090	;IN: TABLE CODE IN A
	6091	0=SPRITE NAME TABLE
	6092	1=SPRITE GENERATOR TABLE, 2=PATTERN NAME
	6093	TABLE, 3= PATTERN GENERATOR TABLE, 4=
	6094	COLOR TABLE
	6095	START_INDEX IN DE,
	6096	DATA BUFFER IN HL, AND COUNT IN 1Y.
18A3 CD18AA	CALL SET_COUNT	
18A6 CD1D3E	CALL VRAM_READ	
18A9 C9	RET	
18AA	6100	
	6101	
	6102 SET_COUNT	
	6103	;CALLED BY PUT_VRAM AND GET_VRAM
	6104	;SETS BYTE COUNT AND INDEX FOR
	6105	WRITES AND READS TO AND FROM VRAM
	6106	TABLE BYTES/ITEM
	6107	SPRITE_NAME 4
	6108	SPRITE_GEN 8
	6109	PATTERN_NAME 1
	6110	PATTERN_GEN 8
	6111	COLOR (MODE 1) 1
	6112	COLOR (MODE 2) 8
18AA FD2273FE	LD	(SAVED_COUNT), 1Y
18AE D02173F2	LD	IX, VRAM_ADDR_TABLE
18B2 4F	LD	C, A
18B3 0600	LD	B, 0
18B5 FE04	CP	4
18B7 2007	JR	NZ, SET_COUNT10
	6119	..** COLOR TABLE, CHECK IF MODE 1
18B9 3A73C3	LD	A, [VDP_MODE_WORD]
18BC CB4F	BIT	1, A
18BE 282C	JR	Z, SET_COUNT20
	6123	..** NOT MODE 1, ADJUST ITEM COUNT AND INDEX
18C0	6124 SET_COUNT10	
18C0 FD211BFF	LD	1Y, SHIFT_CT
18C4 FD09	ADD	1Y, BC
18C6 FD7E00	LD	A, [1Y+0]
18C9 FE00	CP	0
18CB 281F	JR	Z, SET_COUNT20
18CD	6130 ADJUST_INDEX	
18CD CB23	SLA	E
18CF CB12	RL	D
18D1 30	DEC	A
18D2 20F9	JR	NZ, ADJUST_INDEX
18D4	6135 END_ADJ_INDEX	
18D4 C5	PUSH	BC
18D5 ED4B73FE	LD	BC, (SAVED_COUNT)
18D9 FD7E00	LD	A, [1Y+0]
18DC FE00	CP	0

LOCATION OBJECT CODE LINE SOURCE LINE

```

180E 2008      6140 JR      Z,END_ADJ_COUNT
180F          6141 ADJUST_COUNT
1810 CB21      6142 SLA C      ;MULTIPLY ITEM COUNT TO GET BYTE COUNT
1811 CB10      6143 RL B      ;
1812 30        6144 DEC A      ;
1813 20F9      6145 JR      NZ,ADJUST_COUNT
1814 ED4373FE  6146 LD      [SAVED_COUNT],BC ;SAVE ADJUSTED COUNT
1815          6147 END_ADJ_COUNT
1816          6148
1817 C1        6149 POP BC      ;RESTORE TABLE_CODE/INDEX
1818 E5         6150 SET_COUNT20
1819 D009      6151 PUSH HL
181A D009      6152 ADD IX,BC ;GET TABLE ADDRESS IN VRAM
181B D06E00    6153 ADD IX,BC ;
181C D06601    6154 LD      L,[IX+0] ;LOW ORDER OF VRAM ADDRESS
181D 19        6155 LD      H,[IX+1] ;HIGH ORDER OF VRAM ADDRESS
181E E8        6156 ADD HL,DE ;ADD BYTE INDEX TO GET VRAM START ADDRESS
181F E1        6157 EX DE,HL ;VRAM DESTINATION NOW IN DE
1820 ED4873FE  6158 POP HL ;RESTORE DATA POINTER
1821          6159 LD      BC,[SAVED_COUNT] ;RESTORE ADJUSTED COUNT
1822          6160 SET_COUNTX
1823 C9         6161 RET
1824          6162
1825          6163
1826          6164 SHIFT_CT
1827          6165 DEFB      2,3,0,3,3
1828          6166
1829          6167
1830          6168 * PROCEDURE PUT_VRAMQ (TABLE_CODE:BYTE;START_INDEX:BYTE;SLICE:BYTE;
1831          6169 * VAR DATA:BUFFER;ITEM_COUNT:INTEGER);
1832          6170
1833          6171 * THIS IS THE PASCAL ENTRY POINT TO INIT_TABLE
1834          6172
1835          6173 PUT_VRAM_P DEFV      5,1,1,1,-2,2
1836          6174 * THIS IS THE PARAMETER DESCRIPTOR FOR INIT_TABLEQ
1837          6175
1838          6176 PUT_VRAMQ
1839          6177 LD      BC,PUT_VRAM_P
1840          6178 LD      DE,PARAM_AREA
1841          6179 CALL PARAM
1842          6180 LD      A,[PARAM_AREA]
1843          6181 DE,[PARAM_AREA+1]
1844          6182 IY,[PARAM_AREA+5]
1845          6183 HL,[PARAM_AREA+3]
1846          6184
1847          6185 PUT_VRAM_
1848          6186 ;WRITES A CERTAIN NUMBER OF BYTES TO VRAM
1849          6187 ;FROM A BUFFER.
1850          6188 ;IN: TABLE CODE IN A
1851          6189 ; 0=SPRITE NAME TABLE
1852          6190 ; 1=SPRITE GENERATOR TABLE, 2=PATTERN NAME
1853          6191 ; TABLE, 3= PATTERN GENERATOR TABLE, 4=
1854          6192 ; COLOR TABLE
1855          6193 ; START INDEX IN DE,
1856          6194 ; DATA BUFFER IN HL, AND COUNT IN IY.

```

```

LOCATION OBJECT CODE LINE      SOURCE LINE

1C27 F5      6195      PUSH AF
1C28 FE00    6196 *    IF (TABLE_CODE = SPRITE_NAME_TABLE) AND (MUX_SPRITES = TRUE) THEN
1C2A 2022    6197      CP 0
1C2C 3A73C7  6198      JR NZ,ELSEZZ
1C2F FE01    6199      LD A,(MUX_SPRITES)
1C31 2018    6200      CP 1
1C31 2018    6201      JR NZ,ELSEZZ
1C31 2018    6202      JR NZ,ELSEZZ
1C33 F1      6203 *    WRITE ENTRY TO LOCAL TABLE
1C34 E5      6204      POP AF          ; CLEAR STACK
1C34 E5      6205      PUSH HL      ; [SP] = DATA BUFFER
1C35 2A8002  6206
1C38 7B      6207      LD HL,(LOCAL_SPR_TBL) ; CALCULATE ADDRESS IN TABLE
1C39 CB27    6208      LD A,E
1C39 CB27    6209      SLA A
1C39 CB27    6210      SLA A
1C39 CB27    6211      LD E,A
1C39 CB27    6212      ADD HL,DE
1C39 CB27    6213      EX DE,HL
1C39 CB27    6214
1C40 FDE5    6215      PUSH IV
1C42 C1      6216      POP BC
1C43 79      6217      LD A,C
1C44 CB27    6218      SLA A
1C44 CB27    6219      SLA A
1C44 CB27    6220      LD C,A
1C44 4F      6221
1C49 E1      6222      POP HL
1C4A ED80    6223      LDIR
1C4C 1807    6224      JR END_IFZZ
1C4C 1807    6225 *    ELSE
1C4E         6226      ELSEZZ
1C4E F1      6227      POP AF
1C4F CD1BAA  6228      CALL SET_COUNT
1C52 CD1D01  6229      CALL VRAM_WRITE
1C55         6230 *    END IF
1C55 C9      6231      END_IFZZ
1C55 C9      6232      RET
1C55 C9      6233
1C55 C9      6234
1C55 C9      6235
1C56 00010001 6236 *    PROCEDURE INIT_SPR_ORDERQ (SPRITE_COUNT:BYTE);
1C56 00010001 6237
1C56 00010001 6238 *    THIS IS THE PASCAL ENTRY POINT TO THE INIT_SPR_ORDER_ROUTINE.
1C56 00010001 6239
1C56 00010001 6240      INIT_SPR_P      DEFW      1,1
1C56 00010001 6241
1C56 00010001 6242      INIT_SPR_ORDERQ EQU
1C56 00010001 6243      LD
1C56 00010001 6244      LD
1C56 00010001 6245      CALL
1C56 00010001 6246      LD
1C56 00010001 6247
1C56 00010001 6248      INIT_SPR_ORDER_
1C56 00010001 6249
1C56 00010001 6250
1C56 00010001 6251

```

; INITIALIZES THE SPRITE DISPLAY ORDER
 ; LIST IN RAM TO DEFAULT ORDER (0...31)
 ; IN: NUMBER OF SPRITES TO ORDER IN 'A'

LOCATION OBJECT CODE LINE SOURCE LINE

```

1C66 47      6252      LD B,A          ;SAVE SPRITE COUNT
1C67 AF      6253      XOR A
1C68 2A8004   6254      LD HL,[SPRITE_ORDER]
1C68 77      6255      INIT_SPR10
1C68 77      6256      LD [HL],A
1C6C 23      6257      INC HL
1C6C 23      6258      INC A
1C6E B8      6259      CP B
1C6F 20FA     6260      JR NZ,INIT_SPR10
1C71 C9      6261      RET
1C72 00010001 6262
1C72 00010001 6263      * PROCEDURE WR_SPR_MM_TBLQ (SPRITE_COUNT:BYTE);
1C72 00010001 6264
1C72 00010001 6265      * THIS IS THE PASCAL ENTRY POINT TO THE WR_SPR_MM_TBL ROUTINE.
1C72 00010001 6266
1C72 00010001 6267      WR_SPR_P      DEFW 1,1
1C72 00010001 6268
1C72 00010001 6269      WR_SPR_MM_TBLQ EQU $
1C72 00010001 6270      LD BC,WR_SPR_P
1C72 00010001 6271      LD DE,PARAM_AREA
1C72 00010001 6272      CALL PARAM
1C72 00010001 6273      LD A,[PARAM_AREA]
1C72 00010001 6274
1C82         6275      WR_SPR_MM_TBL
1C82         6276      ;WRITES SPRITE NAME TABLE TO VRAM
1C82         6277      ;USING THE SPRITE ORDER LIST.
1C82         6278      ;
1C82         6279      ; NUMBER OF SPRITES TO WRITE IN 'A'
1C82         6280      ;MODE_0_PORT EQU 0BEH
1C82         6281      ;MODE_1_PORT EQU 0BFH
1C82         6282
1C82 002A8004 6283      LD IX,[SPRITE_ORDER] ;LIST OF DISPLAY ORDERS
1C82 002A8004 6284      LD [SPRITE_CT],A ;SAVE COUNT
1C82 002A8004 6285      PUSH AF ;SAVE COUNT
1C82 002A8004 6286      LD IY,VRAM_ADDR_TABLE
1C82 002A8004 6287      LD E,[IY+0]
1C82 002A8004 6288      LD D,[IY+1] ;VRAM SPRITE_NAME_TABLE ADDRESS NOW IN DE
1C82 002A8004 6289
1C82 002A8004 6290      ;** SET UP VDP TO RECEIVE DATA
1C82 002A8004 6291
1C82 002A8004 6292      LD A,E
1C82 002A8004 6293      OUT [MODE_1_PORT],A
1C82 002A8004 6294      LD A,D
1C82 002A8004 6295      OR 40H
1C82 002A8004 6296      OUT [MODE_1_PORT],A
1C82 002A8004 6297
1C82 002A8004 6298      POP AF ;RESTORE COUNT
1C82 002A8004 6299      OUTPUT_LOOP_TABLE MA
1C82 002A8004 6300      LD HL,[LOCAL_SPR_TBL] ;RESTORE RAM SPRITE_NAME_TABLE POINTER
1C82 002A8004 6301      LD C,[IX+0] ;DISPLAY ORDER
1C82 002A8004 6302      INC IX ;ADVANCE DISPLAY ORDER POINTER
1C82 002A8004 6303      LD B,0
1C82 002A8004 6304      ADD HL,BC
1C82 002A8004 6305      ADD HL,BC
1C82 002A8004 6306      ADD HL,BC
1C82 002A8004 6307      ADD HL,BC
1C82 002A8004 6308

```

LOCATION	OBJECT CODE	LINE	SOURCE	LINE
		6309	; **	OUTPUT TO VRAM THROUGH VDP
		6310		
1CAB 0604		6311	LD	B,4 ;ELEN,4) COUNT FOR ONE SPRITE
1CAA 0EBE		6312	LD	C,MODE_0_PORT ;OUTPUT PORT IN 'C'
1CAC		6313	OUTPUT_LOOP10	
1CAC EDA3		6314	OUTI	
1CAE 00		6315	MOP	
1CAF 00		6316	MOP	;DELAY
1CB0 20FA		6317	JR	NZ,OUTPUT_LOOP10
		6318	LD	A,[SPRITE_CT]
		6319	DEC	A
		6320	LD	[SPRITE_CT],A
1CB2 30		6321	DEC	A
1CB3 20E5		6322	JR	NZ,OUTPUT_LOOP_TABLE_MA
1CB5 C9		6323	RET	
		6324		
		6325	GLB	VRAM_ADDR_TABLE
		6326	GLB	SPRITE_NAME_TBL
		6327	GLB	SPRITE_GEN_TBL
		6328	GLB	PATRN_NAME_TBL
		6329	GLB	PATRN_GEN_TBL
		6330	GLB	COLOR_TABLE
		6331	DATA	
73F2		6332	VRAM_ADDR_TABLE	
73F3		6333	SPRITE_NAME_TBL	DEFS 2
73F4		6334	SPRITE_GEN_TBL	DEFS 2
73F5		6335	PATRN_NAME_TBL	DEFS 2
73F6		6336	PATRN_GEN_TBL	DEFS 2
73F7		6337	COLOR_TABLE	DEFS 2
73FA		6338	* THIS TABLE HOLDS THE BASE ADDRESSES OF ALL THE VRAM TABLES.	
		6339		
		6340		
73FC		6341	SAVE_TEMP	DEFS 2
73FE		6342	SAVED_COUNT	DEFS 2
		6343		
		6344	COMM	
		6345		
		6346	;PARAM AREA	DEFS 6
		6347	* THIS IS THE PARAMETER PASSING AREA FOR THE PASCAL ENTRY POINTS TO	
		6348	* ROUTINES IN THIS MODULE. IT IS HELD IN COMMON WITH OTHER SUCH ENTRY	
		6349	* POINTS FOR OTHER MODULES.	
		6350	PROG	

LOCATION OBJECT CODE LINE SOURCE LINE

6465 * THIS IS THE PARAMETER DESCRIPTOR FOR REG_WRITE

```

6466
6467 * BEGIN REG_WRITE
6468
6469 REG_WRITE EQU GLB
6470 LD BC,REG_WRITE_P
6471 LD DE,PARAM_AREA
6472 CALL PARAM
6473 LD HL,[PARAM_AREA]
6474 LD C,H
6475 LD B,L
6476
6477 REG_WRITE EQU GLB
6478
6479
6480 * VALUE =; CTRL_PORT
6481 LD A,C
6482 OUT CTRL_PORT,A
6483
6484 * REGISTER + 80H =; CTRL_PORT
6485 LD A,B
6486 ADD A,80H
6487 OUT CTRL_PORT,A
6488
6489 * IF REGISTER = 0 THEN VDP_MODE_WORD[0] := VALUE
6490 LD A,B
6491 CP 0
6492 JR NZ,NOT_REG_0
6493 LD A,C
6494 LD [VDP_MODE_WORD],A
6495 NOT_REG_0 EQU $
6496
6497 * IF REGISTER = 1 THEN VDP_MODE_WORD[1] := VALUE
6498 LD A,B
6499 CP 1
6500 JR NZ,NOT_REG_1
6501 LD A,C
6502 LD [VDP_MODE_WORD+1],A
6503 NOT_REG_1 EQU $
6504
6505 * END REG_WRITE
6506 RET
6507
6508 * PROCEDURE VRAM_WRITE (VAR DATA:BUFFER;DEST:INTEGER;COUNT:INTEGER)
6509
6510 * VAR DATA (POINTER TO DATA BUFFER) IS PASSED IN HL
6511 * DEST IS PASSED IN DE
6512 * COUNT IS PASSED IN BC
6513 * DESTROYS: ALL
6514
6515 VRAM_WRITE_P DEFW 3,-2,2,2
6516 * THIS IS THE PARAMETER DESCRIPTOR FOR VRAM_WRITE
6517
6518 * BEGIN VRAM_WRITE
6519 GLB EQU VRAM_WRITE
6520 VRAM_WRITE EQU $
6521
6522 <1CED>

```


LOCATION	OBJECT CODE	LINE	SOURCE	LINE
1CE0 011CE5	6521	LD	BC,VRAM_WRITE_P	
1CF0 11738A	6522	LD	DE,PARAM_AREA	
1CF3 CD0098	6523	CALL	PARAM	
1CF6 2A738A	6524	LD	HL,(PARAM_AREA)	
1CF9 ED58738C	6525	LD	DE,(PARAM_AREA+2)	
1CFD ED4B738E	6526	LD	BC,(PARAM_AREA+4)	
	6527			
	6528	GLB	VRAM_WRITE	
	6529	EQU	\$	
	6530			
	6531	* DEST := DEST + 4000H		
1D01 E5	6532	PUSH	HL	
1D02 D5	6533	PUSH	DE	
1D03 E1	6534	POP	HL	
1D04 114000	6535	LD	DE,4000H	
1D07 19	6536	ADD	HL,DE	
	6537			
	6538	* LOW BYTE OF DEST := CTRL_PORT		
1D08 7D	6539	LD	A,L	
1D09 D38F	6540	OUT	[CTRL_PORT],A	
	6541			
	6542	* HIGH BYTE OF DEST := CTRL_PORT		
1D08 7C	6543	LD	A,H	
1D0C D38F	6544	OUT	[CTRL_PORT],A	
	6545			
	6546	* DATA := DATA_PORT		
1D0E C5	6547	PUSH	BC	
1D0F D1	6548	POP	DE	
1D10 E1	6549	POP	HL	
1D11 DEBE	6550	LD	C,DATA_PORT	
1D13 43	6551	LD	B,E	
	6552	OUTPUT_LOOP		
1D14 EDA3	6553	EQU	\$	
1D16 00	6554	OUTI	NZ,OUTPUT_LOOP	
1D17 00	6555	MOP	D	
1D18 C21D14	6556	JP	M,END_OUTPUT	
1D1B 15	6557	DEC	NZ,OUTPUT_LOOP	
1D1C FA1D21	6558	JP		
1D1F 20F3	6559	JR		
	6560			
	6561	* END VRAM_WRITE		
	6562	EQU	\$	
	6563	RET		
	6564			
	6565	* PROCEDURE VRAM_READ (VAR DATA:BUFFER;SRCE:INTEGER;COUNT:INTEGER)		
	6566			
	6567	* VAR DATA (POINTER TO DATA BUFFER) IS PASSED IN HL		
	6568	* SRCE IS PASSED IN DE		
	6569	* COUNT IS PASSED IN BC		
	6570	* DESTROYS: ALL		
	6571			
	6572	VRAM_READ_P	DEFW	3,2,2,2
1D22 0003FFFE				
1D26 00020002				
	6573	* THIS IS THE PARAMETER DESCRIPTOR FOR VRAM_READ		
	6574			
	6575	* BEGIN VRAM_READ		
	6576	GLB	VRAM_READ	

LOCATION	OBJECT CODE	LINE	SOURCE LINE
102A	<102A>	6577	VRAM_READQ EQU \$
102A 011D22		6578	LD BC, VRAM_READ_P
102D 11738A		6579	LD DE, PARAM_AREA
1030 CD0098		6580	CALL PARAM
1033 2A738A		6581	HL, [PARAM_AREA]
1036 ED58738C		6582	DE, [PARAM_AREA+2]
103A ED48738E		6583	BC, [PARAM_AREA+4]
103E 7B		6584	LD VRAM_READ
103F D38F		6585	GLB EQU \$
1041 7A		6586	VRAM_READ EQU \$
1042 D38F		6587	
1044 C5		6588	* LOW BYTE OF SRCE =; CTRL_PORT
1045 D1		6589	LD A, E
1046 0E8E		6590	OUT [CTRL_PORT], A
1048 43		6591	
1049 EDA2		6592	* HIGH BYTE OF SRCE =; CTRL_PORT
104B 00		6593	LD A, D
104C 00		6594	OUT [CTRL_PORT], A
104D C21049		6595	
1050 15		6596	* DATA =; DATA_PORT
1051 FA1D56		6597	PUSH
1054 20F3		6598	POP
		6599	LD BC
		6600	DE
		6601	C, DATA_PORT
		6602	B, E
		6603	\$
		6604	
		6605	NZ, INPUT_LOOP
		6606	D
		6607	M, END_INPUT
		6608	NZ, INPUT_LOOP
		6609	
		6610	* END VRAM_READ
		6611	END_INPUT EQU \$
		6612	RET
		6613	
		6614	* FUNCTION REG_READ:BYTE
		6615	
		6616	* FUNCTION OUTPUT RETURNED IN A
		6617	* DESTROYS A ONLY
		6618	
		6619	* BEGIN REG_READ
		6620	GLB EQU \$
		6621	REG_READ EQU \$
		6622	
		6623	* REG_READ := CTRL_PORT
		6624	IN A, [CTRL_PORT]
		6625	
		6626	* END REG_READ
		6627	RET
		6628	
		6629	PROG

LOCATION OBJECT CODE LINE SOURCE LINE

```
6631 * THIS IS A PACKAGE OF ROUTINES THAT ALLOW APPLICATIONS PROGRAMMERS TO
6632 * OPERATE ON SHAPE GENERATORS. EACH OF THEM TAKES, AS INPUTS, AN AREA
6633 * IN ONE OF THE GENERATOR TABLES IN WHICH THE GENERATORS TO BE OPERATED
6634 * UPON RESIDE, A COUNT OF THE GENERATORS TO BE USED, AND AN AREA OF THE
6635 * SAME TABLE INTO WHICH THE RESULTS ARE TO BE PUT. THE ONLY RAM AREA THEY
6636 * IS IN THE WORK_BUFFER A POINTER TO WHICH IS DECLARED AS AN EXTERNAL
6637 * AND DEFINED IN THE CARTRIDGE.
6638
6639 ***** NOTE: *****
6640 ***** THESE ROUTINES WRITE TO AND READ *****
6641 ***** WITHOUT POSSIBILITY OF DEFERL *****
6642 ***** AND SHOULD NOT BE USED IN ANY *****
6643 ***** SITUATION WHERE THEY MAY BE *****
6644 ***** INTERRUPTED. *****
6645 *****
6646 *****
6647
6648 ; EXT WORK_BUFFER
6649 ; POINTER TO THE WORK_BUFFER DEFINED BY THE CARTRIDGE PROGRAMMER
6650
6651 ; EXT VDP_MODE_WORD
6652 ; THE WORD IN OS RAM THAT DESCRIBES THE CURRENT GRAPHICS MODE.
6653
6654 ; EXT GET_VRAM
6655 ; EXT PUT_VRAM
6656 ; EXTERNAL OS ROUTINES IN TABLE_MANAGER MODULE
6657
6658 ; EXT MIRROR_LR
6659 ; EXT MIRROR_UD
6660 ; EXT ROTATE
6661 ; EXT MAGNIFY
6662 ; EXT QUADRUPLER
6663 ; EXTERNAL ROUTINES THAT PERFORM BLOCK OPERATIONS
6664
6665 ; TRUE EQU 1
6666 ; FALSE EQU 0
6667 ; VALUES FOR BOOLEAN FLAGS
6668
6669 PATTERN_GEN EQU 3
6670 COLOR_TABLE EQU 4
6671 ; VALUES FOR TABLE CODE
6672
6673 ; PROCEDURE REFLECT_VERTICAL (TABLE_CODE(A),SOURCE(DE),DESTINATION(HL),COUNT(BC))
6674
6675 * REFLECT REFLECTS EACH OF A BLOCK OF GENERATORS FROM VRAM AROUND
6676 * THE VERTICAL AXIS. IF THE GENERATORS ARE FROM THE PATTERN PLANE
6677 * AND THE GRAPHICS MODE IS 2, THEN THE ROUTINE ALSO COPIES THE
6678 * CORRESPONDING COLOR GENERATORS., OTHERWISE IS ASSUMES THAT THE COLOR
6679 * DATA HAS ALREADY BEEN SET UP.
6680
6681 * BEGIN REFLECT_VERTICAL
6682 ; RFLCT_VERT
6683 ; ACTUAL ROUTINE NAME EXISTS IN OS
6684 ; JUMP TABLE ONLY
6685
6686 * SET OPERATION CODE
6687 LD IX,RFLCT_VERT
```

LOCATION	OBJECT CODE LINE	SOURCE LINE
105E 1810	6688	
	6689	* CONTINUE BELOW
	6690	JR
	6691	CONTINUE_GRAPHICS
	6692	
	6693	* PROCEDURE REFLECT_HORIZONTAL (TABLE_CODE(A),SOURCE(DE),DESTINATION(HL),COUNT(BC))
	6694	
	6695	* REFLECT_HORIZONTAL REFLECTS EACH OF A BLOCK OF GENERATORS FROM VRAM
	6696	* AROUND THE HORIZONTAL AXIS. IF THE GENERATORS ARE FROM THE PATTERN
	6697	* PLANE AND THE GRAPHICS MODE IS 2 THEN IT REFLECTS THE CORRESPONDING
	6698	* COLOR GENERATORS AS WELL.
	6699	
	6700	* BEGIN REFLECT_HORIZONTAL
	6701	REFLECT_HOR
	6702	; ACTUAL ROUTINE NAME EXISTS IN OS
	6703	; JUMP TABLE ONLY
	6704	
	6705	* SET OPERATION CODE
	6706	LD
	6707	
	6708	* CONTINUE BELOW
	6709	JR
	6710	
	6711	
	6712	
	6713	
	6714	* PROCEDURE ROTATE_90 (TABLE_CODE(A),SOURCE(DE),DESTINATION(HL),COUNT(BC))
	6715	
	6716	* ROTATE_90 ROTATES EACH OF A BLOCK OF GENERATORS FROM VRAM 90 DEGREES
	6717	* CLOCKWISE. IF THE GENERATORS ARE FROM THE PATTERN PLANE AND THE
	6718	* GRAPHICS MODE IS 2 THEN THE ROUTINE COPIES THE CORRESPONDING COLOR
	6719	* ENTRIES AS WELL.
	6720	
	6721	* BEGIN ROTATE_90
	6722	GLB
	6723	ROT_90
	6724	; ACTUAL ROUTINE NAME EXISTS IN OS
	6725	; JUMP TABLE ONLY
	6726	
	6727	* SET OPERATION CODE
	6728	LD
	6729	* CONTINUE BELOW
	6730	JR
	6731	
	6732	
	6733	
	6734	
	6735	* PROCEDURE ENLARGE (TABLE_CODE(A),SOURCE(DE),DESTINATION(HL),COUNT(BC))
	6736	
	6737	* ENLARGE TAKES EACH OF A BLOCK OF GENERATORS AND ENLARGES IT INTO
	6738	* A BLOCK OF FOUR GENERATORS WHEREIN EACH PIXEL OF THE ORIGINAL
	6739	* GENERATOR IS EXPANDED TO FOUR PIXELS IN THE NEW ONES. IF THE
	6740	* GENERATORS ARE FROM THE PATTERN PLANE AND THE GRAPHICS MODE IS 2
	6741	* THE ROUTINE ALSO QUADRUPLES EACH OF THE CORRESPONDING COLOR
	6742	* GENERATORS AS WELL.
	6743	
	6744	* BEGIN ENLARGE

LOCATION	OBJECT	CODE LINE	SOURCE LINE
106C		6745	GLB
		6746	ENLGR
		6747	ENLGR ; ACTUAL ROUTINE NAME EXISTS IN OS
		6748	ENLGR ; JUMP TABLE ONLY
		6749	
106C D0211E07		6750	* SET OPERATION CODE
		6751	LD IX,ENLGR
		6752	
		6753	
1070		6754	* CONTINUE EXECUTION HERE
		6755	CONTINUE_GRAPHICS
		6756	
		6757	* SAVE ALL ENTRY PARAMETERS
1070 D9		6758	EXX
1071 08		6759	EX
1072 D0E5		6760	PUSH
		6761	
		6762	* REPEAT
1074		6763	MAIN_LOOP
		6764	
		6765	* GET_VRAM_ (TABLE_CODE,SOURCE,WORK_BUFFER(0...7),1)
1074 08		6766	EX AF,AF
1075 F5		6767	PUSH
1076 08		6768	EX AF,AF
1077 F1		6769	POP AF
1078 D9		6770	EXX
1079 D5		6771	PUSH
107A D9		6772	EXX
107B D1		6773	POP
107C F0210001		6774	LD
1080 2A8006		6775	LD
1083 CD1BA3		6776	CALL
		6777	
		6778	* EXECUTE ENCODED OPERATION BELOW
1086 D0E1		6779	POP
1088 D0E5		6780	PUSH
108A D0E9		6781	JP [IX]
		6782	RETURN_HERE EQU \$
		6783	
		6784	* SOURCE : SUCC (SOURCE)
108C 13		6785	INC
		6786	
		6787	* COUNT := PRED (COUNT)
108D 08		6788	DEC
		6789	
		6790	* UNTIL COUNT = 0
108E 78		6791	LD
108F 81		6792	OR
1090 D9		6793	EXX
1091 20E1		6794	JR
		6795	
		6796	* END (ALL)
1093		6797	ALL_X
1093 D0E1		6798	POP
1095 C9		6799	RET
		6800	
1096		6801	RI LCT VERT

LOCATION OBJECT CODE LINE SOURCE LINE

```

6802 * OPERATIONS SPECIFIC TO REFLECT_VERTICAL ROUTINE
6803
1096 2A8006 6804 * MIRROR_L_R(WORK_BUFFER[0..7],WORK_BUFFER[8..15])
1099 010008 6805 LD HL,(WORK_BUFFER)
109C E5 6806 LD BC,B
109E 01 6807 PUSH HL
109F E8 6808 POP DE
10A0 CD1F00 6809 ADD HL,BC
6810 EX DE,HL
6811 CALL MIRROR_L_R
6812
10A3 CD1E72 6813 * PUT_VRAM (TABLE_CODE,DESTINATION,WORK_BUFFER[8..15],1)
6814 CALL PUT_TABLE
6815
10A6 CD1E50 6816 * IF COLOR_TEST THEN
10A9 FE01 6817 CALL COLOR_TEST
10AB 2006 6818 CP TRUE
6819 JR NZ,END_IF_1_GRAPHICS
6820
10AD CD1E89 6821 * GET_VRAM(COLOR_TABLE,SOURCE,WORK_BUFFER[0..7],1)
6822 CALL GET_COLOR
6823
10B0 CD1E9A 6824 * PUT_VRAM(COLOR_TABLE,DESTINATION,WORK_BUFFER[0..7],1)
6825 CALL PUT_COLOR
6826
10B3 6827 * END IF
6828 END_IF_1_GRAPHICS
6829
10B3 D9 6830 * DESTINATION := SUCC (DESTINATION)
10B4 23 6831 EXX
6832 INC HL
6833
10B5 1B05 6834 * END
6835 JR RETURN_HERE
6836
6837
10B7 6838
6839 RFLCT_HOR
6840 * OPERATIONS SPECIFIC TO REFLECT_HORIZONTAL ROUTINE
6841
10B7 2A8006 6842 * MIRROR_U_D(WORK_BUFFER[0..7],WORK_BUFFER[8..15])
10BA 010008 6843 LD HL,(WORK_BUFFER)
10B0 E5 6844 LD BC,B
10BE D1 6845 PUSH HL
10BF 09 6846 POP DE
10C0 E8 6847 ADD HL,BC
6848 EX DE,HL
6849 CALL MIRROR_U_D
6850
10C4 CD1E72 6851 * PUT_VRAM (TABLE_CODE,DESTINATION,WORK_BUFFER[8..15],1)
6852 CALL PUT_TABLE
6853
10C7 CD1E50 6854 * IF COLOR_TEST THEN
10CA FE01 6855 CALL COLOR_TEST
10CC 2013 6856 CP TRUE
6857 JR NZ,END_IF_2_GRAPHICS

```

```

LOCATION OBJECT CODE LINE SOURCE LINE
1DCE CD1E89 6859 * GET_VRAM(COLOR_TABLE,SOURCE,WORK_BUFFER[0..7],1)
6860 CALL GET_COLOR
6861
1D01 2A8006 6862 * MIRROR_U_D(WORK_BUFFER[0..7],WORK_BUFFER[8..15])
1D04 010008 6863 LD HL,(WORK_BUFFER)
1D07 E5 6864 LD BC,8
1D08 D1 6865 PUSH HL
1D09 09 6866 POP DE
1D0A E8 6867 ADD HL,BC
1D0B CD1F4E 6868 EX DE,HL
6869 CALL MIRROR_U_D
6870
1D0E CD1E9A 6871 * PUT_VRAM(COLOR_TABLE,DESTINATION,WORK_BUFFER[8..15],1)
6872 CALL PUT_COLOR
6873
1D01 6874 * END IF
6875 END_IF_2_GRAPHICS
6876
1D01 D9 6877 * DESTINATION := SUCC (DESTINATION)
1D02 23 6878 EXX
6879 INC HL
6880
1D03 18A7 6881 * END
6882 JR RETURN_HERE
6883
1D05 6884
6885
6886 ROT 90
6887 * OPERATIONS SPECIFIC TO THE ROTATE_90 ROUTINE
6888
6889 * ROTATE(WORK_BUFFER[0..7],WORK_BUFFER[8..15])
6891 LD HL,(WORK_BUFFER)
6892 LD BC,8
6893 LD HL
6894 POP DE
6895 ADD HL,BC
6896 EX DE,HL
6897 CALL ROTATE
6898
1D02 CD1E72 6899 * PUT_VRAM (TABLE_CODE,DESTINATION,WORK_BUFFER[8..15],1)
6900 CALL PUT_TABLE
6901
6902 * IF COLOR_TEST THEN
6903 CALL COLOR_TEST
6904 CP TRUE
6905 JR WZ,END_IF_3_GRAPHICS
6906
6907 * GET_VRAM(COLOR_TABLE,SOURCE,WORK_BUFFER[0..7],1)
6908 CALL GET_COLOR
6909
6910 * PUT_VRAM(COLOR_TABLE,DESTINATION,WORK_BUFFER[0..7],1)
6911 CALL PUT_COLOR
6912
6913 * END IF
6914 END_IF_3_GRAPHICS
6915

```

LOCATION	OBJECT CODE	LINE	SOURCE	LINE
		6916 *	DESTINATION := SUCC (DESTINATION)	
1E02 D9		6917	EXX	
1E03 23		6918	INC	HL
		6919		
1E04 C3108C		6920 *	END	
		6921	JP	RETURN_HERE
		6922		
		6923		
		6924		
1E07		6925	ENLGR	
		6926 *	OPERATIONS SPECIFIC TO THE ENLARGE ROUTINE	
		6927		
1E07 2A8006		6928 *	MAGNIFY(WORK_BUFFER[0..7],WORK_BUFFER[8..39])	
1E0A 010008		6929	LD	HL,(WORK_BUFFER)
1E0D E5		6930	LD	BC,0
1E0E D1		6931	PUSH	HL
1E0F D1		6932	POP	DE
1E10 09		6933	ADD	HL,BC
1E10 E8		6934	EX	DE,HL
1E11 CD1EAB		6935	CALL	MAGNIFY
		6936		
1E14 08		6937 *	PUT_VRAM (TABLE_CODE,DESTINATION,WORK_BUFFER[8..39],4)	
1E15 F5		6938	EX	AF,AF
1E16 08		6939	PUSH	AF
1E17 F1		6940	EX	AF,AF
1E18 D9		6941	POP	AF
1E19 E5		6942	EXX	
1E1A D9		6943	PUSH	HL
1E1B D1		6944	EXX	
1E1C 2A8006		6945	POP	DE
1E1F 010008		6946	LD	HL,(WORK_BUFFER)
1E22 09		6947	LD	BC,0
1E23 FD210004		6948	ADD	HL,BC
1E27 CD1C27		6949	LD	IX,4
		6950	CALL	PUT_VRAM
		6951		
1E2A CD1E50		6952 *	IF COLOR_TEST THEN	
1E2D FE01		6953	CALL	COLOR_TEST
1E2F 2024		6954	CP	TRUE
		6955	JR	NZ,END_IF_4_GRAPHICS
		6956		
1E31 CD1E89		6957 *	GET_VRAM(COLOR_TABLE,SOURCE,WORK_BUFFER[0..7],1)	
		6958	CALL	GET_COLOR
		6959		
1E34 2A8006		6960 *	QUADRUPLE(WORK_BUFFER[0..7],WORK_BUFFER[8..39])	
1E37 010008		6961	LD	HL,(WORK_BUFFER)
1E3A E5		6962	LD	BC,0
1E3B D1		6963	PUSH	HL
1E3C 09		6964	POP	DE
1E3D E8		6965	ADD	HL,BC
1E3E CD1EEA		6966	EX	DE,HL
		6967	CALL	QUADRUPLE
		6968		
1E41 3E04		6969 *	PUT_VRAM(COLOR_TABLE,DESTINATION,WORK_BUFFER[8..39],4)	
1E4:		6970	LD	A,COLOR_TABLE
1E44 E5		6971	EXX	
		6972	PUSH	HL