CTV COPY 6/9/2000

DECRYPT.L   April 23, 1984   1:29:06 pm   page 1.

Assembling DECRYPT.S
---------65xx Assembler V5.6 ---
                                                                PAGE   1
                                                        ----- Kontron -----

```
          *   MARIAOS         MARIA DATA LOCATION DEFINITIONS

          *  NOTE THE FOLLOWING WIERD THINGS ABOUT THE RAM:
          *          $00-$3F <=> $100-$13F
          *          $80-$FF <=> $180-$1FF
          *          $40-$FF <=> $2040-$20FF
          *          $140-$1FF <=> $2140-$21FF

          *   TIA REGISTERS
0001       INPTCTRL    EQU      $01          ;INPUT CONTROL
000C       INPT4       EQU      $0C          ;BITS 7        PLAYER 0 BUTTO
000D       INPT5       EQU      $0D          ;BITS 7        PLAYER 1 BUTTO
0015       AUDC0       EQU      $15          ;BITS    3210 AUDIO CONTROL
0016       AUDC1       EQU      $16          ;BITS    3210 AUDIO CONTROL
0017       AUDF0       EQU      $17          ;BITS   43210 AUDIO FREQUENC
0018       AUDF1       EQU      $18          ;BITS   43210 AUDIO FREQUENC
0019       AUDV0       EQU      $19          ;BITS    3210 AUDIO VOLUME 0
001A       AUDV1       EQU      $1A          ;BITS    3210 AUDIO VOLUME 1

          *   MARIA REGISTERS
0020       BACKGRND    EQU      $20          ;BACKGROUND COLOR
0021       P0C1        EQU      $21          ;PALETTE 0, COLOR 1
0022       P0C2        EQU      $22          ;PALETTE 0, COLOR 2
0023       P0C3        EQU      $23          ;PALETTE 0, COLOR 3
0024       WSYNC       EQU      $24          ;FAST MARIA WSYNC STROBE
0025       P1C1        EQU      $25          ;PALETTE 1, COLOR 1
0026       P1C2        EQU      $26          ;PALETTE 1, COLOR 2
0027       P1C3        EQU      $27          ;PALETTE 1, COLOR 3
0028       MSTAT       EQU      $28          ;BIT 6 IN VBLANK, BIT 7 IN DI
0029       P2C1        EQU      $29          ;PALETTE 2, COLOR 1
002A       P2C2        EQU      $2A          ;PALETTE 2, COLOR 2
002B       P2C3        EQU      $2B          ;PALETTE 2, COLOR 3
002C       DPPH        EQU      $2C          ;DISPLAY LIST POINTER HIGH
002D       P3C1        EQU      $2D          ;PALETTE 3, COLOR 1
002E       P3C2        EQU      $2E          ;PALETTE 3, COLOR 2
002F       P3C3        EQU      $2F          ;PALETTE 3, COLOR 3
0030       DPPL        EQU      $30          ;DISPLAY LIST POINTER LOW
0031       P4C1        EQU      $31          ;PALETTE 4, COLOR 1
0032       P4C2        EQU      $32          ;PALETTE 4, COLOR 2
0033       P4C3        EQU      $33          ;PALETTE 4, COLOR 3
0034       CHARBASE    EQU      $34          ;CHARACTER MODE HIGH POINTER
0035       P5C1        EQU      $35          ;PALETTE 5, COLOR 1
0036       P5C2        EQU      $36          ;PALETTE 5, COLOR 2
0037       P5C3        EQU      $37          ;PALETTE 5, COLOR 3
0038       OFFSET      EQU      $38          ;NOT USED *****
0039       P6C1        EQU      $39          ;PALETTE 6, COLOR 1
003A       P6C2        EQU      $3A          ;PALETTE 6, COLOR 2
003B       P6C3        EQU      $3B          ;PALETTE 6, COLOR 3
003C       CTRL        EQU      $3C          ;BIT 7 CHARACTER WIDTH, BIT 6
003D       P7C1        EQU      $3D          ;PALETTE 7, COLOR 1
003E       P7C2        EQU      $3E          ;PALETTE 7, COLOR 2
003F       P7C3        EQU      $3F          ;PALETTE 7, COLOR 3
```

```
           *   FREE RAM - $40-$FF

           *   ALIASED RAM - $100-$13F

           *   STACK - $140-$1FF

           *   6532 TIMERS AND PORTS

0280       SWCHA   ,  EQU      $280                          ;JOYSTICKS
           *              BIT 7    PLAYER 0 EAST IF CLEAR
           *              BIT 6             WEST
           *              BIT 5             SOUTH
           *              BIT 4             NORTH
           *              BIT 3    PLAYER 1 EAST IF CLEAR
           *              BIT 2             WEST
           *              BIT 1             SOUTH
           *              BIT 0             NORTH

0282       SWCHB      EQU      $282                          ;CONSOLE SWITCHES
           *              BIT 7    PLAYER 1 DIFFICULTY A IF SET, B IF CLEAR
           *              BIT 6    PLAYER 2 DIFFICULTY A IF SET, B IF CLEAR
           *              BIT 3    BLACK AND WHITE VS COLOR - COLOR WHEN SET
           *              BIT 1    GAME SELECT - CLEAR WHEN PRESSED
           *              BIT 0    GAME RESET - CLEAR WHEN PRESSED

0281       CTLSWA     EQU      $281
0283       CTLSWB     EQU      $283
0284       INTIM      EQU      $284                          ;INTERVAL TIMER IN
0295       TIM8T      EQU      $295                          ;TIMER 8T WRITE OUT
0296       TIM64T     EQU      $296                          ;TIMER 64T WRITE OUT
029E       TIM64TI    EQU      $29E                          ;INTERRUPT TIMER 64T


           *   ENDEF.S        ENCRYPTION SYMBOL DEFINITIONS
           *   ADDRESS DEFINITIONS

           *   PAGE 0 - $080-$0FF ($40-$7F TAKEN BY A REGISTER)

00C0       TEST0      EQU      $C0                           ;TEST DATA FOR CPU TEST
00C1       TEST1      EQU      $C1
00C2       TESTW0     EQU      $C2                           ;2 BYTES
00C4       TESTW1     EQU      $C4                           ;2 BYTES

00D0       TEMP0      EQU      $D0                           ;SCRATCH DATA FOR PROGRAM USE
00D1       TEMP1      EQU      $D1                           ;MORE SCRATCH DATA
00D2       TEMP2      EQU      $D2                           ;MORE SCRATCH DATA
00D3       TEMP3      EQU      $D3                           ;MORE SCRATCH DATA
00D4       TEMP4      EQU      $D4                           ;MORE SCRATCH DATA
00D5       TEMP5      EQU      $D5                           ;MORE SCRATCH DATA

00E0       STARTA     EQU      $E0                           ;WHERE ACCUMULATOR STARTS
00E1       OFFSETA    EQU      $E1                           ;OFFSET INTO ACCUMULATOR
00E2       OFFSETR    EQU      $E2                           ;OFFSET INTO A REGISTER
```

Assembling DECRYPT.S

------- 65xx Assembler V5.6 ------- KonLron -------

```
00E3        SIZEA     EQU      $E3        ;SIZE OF ACCUMULATOR
00E4        SIZER0    EQU      $E4        ;SIZE OF REGISTER 0
00E5        SIZER1    EQU      $E5        ;SIZE OF REGISTER 1
00E6        SIZER3    EQU      $E6        ;SIZE OF REGISTER 3
00E7        SIZER5    EQU      $E7        ;SIZE OF REGISTER 5

00EE        CARTBOTM  EQU      $EE        ;BOTTOM OF CARTRIDGE ADDRES
00EF        FUJICOLR  EQU      $EF        ;STARTING COLOR FOR FUJI-A

00F0        KNLSTATE  EQU      $F0        ;HOW MANY MORE ITERATIONS, T
00F1        KNLCOUNT  EQU      $F1        ;TIME FOR CHANGING FUJI COL
00F2        KNLTIME   EQU      $F2        ;TIME THAT COUNT IS GOOD FOR
00F3        KNLOFSET  EQU      $F3        ;HOW STAGGERED THE FUJI COLO

00F4        DLIADDR   EQU      $F4        ;SAME ADDRESS AS IN PACK-IN

            *    HIGH RAM - $1800-$27FF

1800        ACC       EQU      $1800      ;256 BYTE ACCUMULATOR
1900        REG0      EQU      $1900      ;128 BYTE REGISTER
1A00        REG2      EQU      $1A00      ;128 BYTE REGISTER
1B00        REG4      EQU      $1B00      ;128 BYTE REGISTER
1C00        REG6      EQU      $1C00      ;128 BYTE REGISTER
1D00        REG8      EQU      $1D00      ;128 BYTE REGISTER
1E00        REG10     EQU      $1E00      ;128 BYTE REGISTER
1F00        REG12     EQU      $1F00      ;128 BYTE REGISTER

1984        RAMGRAPH  EQU      $1984      ;GRAPHICS IN RAM, $19XX-$1F
1F84        RAMDLL    EQU      $1F84      ;DLL

2000        REG       EQU      $2000      ;         REGISTER

            *         ***** HOLE FROM $2         ;         IN PAGE 0 *****

2100        REG14     EQU      $2100      ;         REGISTER (OVERLAR

            *         ***** HOLE FROM $           ;         PAGE 1 *****

            *   DISPLAY LIST RAM

2200        RAMDLIST  EQU      $2200      ;LISTS FOR WORDS A

            *   MEMORY LOCATIONS

F400        ROMCODE   EQU      $         ;CODE LIVE
F880        ROMCODE2  EQU      $         ;PPED CODE
2300        RAMCODE   EQU      $         ;FOR CODE
0100        CODEDIF   EQU      $         ;EEN OLD AN
```

sembling DECRYPT.S
------- 65xx Assembler V5.6 --

```
00FE          REGION    EQU      $FE                      ;MASK FOR COUNTRY

0004          RANDBYTE  EQU      $04                      ;RANDOM BYTE IN CHECKSUM

              *NTGAME   EQU      $D804                    ;INTERNAL GAME ROM START LOCA
              *NTDLI    EQU      $D8C0                    ;INTERNAL GAME DLI HANDLER
000           INTDLI .  EQU      $F000                    ;OUR DLI

         *  SCAFFOLD.S
         *  THIS DOES THE DISPATCHING WHENEVER THE PACK-IN ISN'T AROUND

000           ORG      INTDLI

000 48        PHA
001 6CF400    JMP      (DLIADDR)

         *  CART.S        ROUTINES DEALING WITH CHECKING THE CARTRIDGE OUT

400           ORG      ROMCODE

400 4CB926    NOCART    JMP      LOCK2600-CODEDIF         ;NO INTERNAL CART
              *OCART    LDA      #$13                     ;TURN SECURITY ROM BACK ON -
              *         STA      INPTCTRL
              *         JSR      GRAPHON2
              *
              *         LDX      #$80
              *OCTLOOP  LDA      KNLSTATE                 ;WAIT A WHILE WITH THE DISPLA
              *         BEQ      NOCTLOOP
              *         LDA      #$00
              *         STA      KNLSTATE
              *         DEX
              *         BNE      NOCTLOOP
              *
              *         JMP      INTGAME                  ;JUMP TO INTERNAL GAME

403 4CB926    BADCART   JMP      LOCK2600-CODEDIF         ;CART DOES NOT CHECK, DO 2600

406 A916      CARTTEST  LDA      #$16                     ;TURN EXTERNAL CART ON
408 8501                STA      INPTCTRL

40A A0FF                LDY      #$FF
40C A27F                LDX      #$7F                     ;SEE IF A CART PLUGGED IN
40E BD00FE    CTSTLOOP  LDA      $FE00,Y
411 D980FD              CMP      $FD80,Y
414 D0EA                BNE      NOCART
416 88                  DEY
417 CA                  DEX
418 10F4                BPL      CTSTLOOP                 ;X LEFT = FF, Y LEFT = 7F

41A ADFCFF              LDA      $FFFC                    ;SEE IF START AT FFFF
```

Assembling DECRYPT.S

------------ 65xx Assembler V5.6 ------------
                                                             Kontron ------

```
F41D 2DFDFF            AND     $FFFD
F420 C9FF              CMP     #$FF
F422 F0DC              BEQ     NOCART              ;ALL LINES DRAWN HIGH, NO CA
F424 ADFCFF            LDA     $FFFC               ;SEE IF START AT 0000
F427 0DFDFF            ORA     $FFFD
F42A F0D4              BEQ     NOCART              ;ALL LINES DRAWN LOW, NO CART

F42C ADF8FF            LDA     $FFF8               ;CHECK FOR REGION VERIFICATI
F42F 09FE              ORA     #REGION
F431 C9FF              CMP     #$FF
F433 D0CE              BNE     BADCART
F435 ADF9FF            LDA     $FFF9               ;SEE IF DATA SIGNATURE EXIS
F438 290B              AND     #$0B                ;1X7 OR 1X3 VALID
F43A C903              CMP     #$03
F43C D0C5              BNE     BADCART
F43E ADF9FF            LDA     $FFF9               ;GET BOTTOM OF CART ADDRESS
F441 29F0              AND     #$F0
F443 85EE              STA     CARTBOTM
F445 8DFD23            STA     CSCMOD0+2-CODEDIF   ;SET UP FOR START OF CHECKSU
F446 C940              CMP     #$40                ;MAKE SURE IT IS NOT TOO LOW
F44A 90B7              BCC     BADCART
                       SEC
F44C E901              SBC     #$01                ;MAKE SURE WE GET FENCEPOST
F44E CDFDFF            CMP     $FFFD               ;MAKE SURE START VECTOR WITHI
F451 B080              BCS     BADCART

F453 202D25(F62D)      JSR     DECRYPT-CODEDIF     ;GET THE DECRYPTED CHECKSUM

F456 A900              LDA     #$00
F458 85F0  (F512)      STA     KNLSTATE            ;GET OUR STATE READY WITH DL

F45A 201224            JSR     CSCHKDLI-CODE       ;CHECK FOR COMING DLI

F45D A916              LDA     #$16                ;GENERATE THE CHECKSUM FOR C
F45F 8501              STA     INPTCTRL            ;FIRST, TURN CART BACK ON
F461 A200              LDX     #$00
F463 8A                TXA
F464 9D0018  CSOLOOP   STA     ACC,X               ;ZERO OUT THE CHECKSUM ACC
F467 CA                DEX
F468 D0FA              BNE     CSOLOOP

F46A 48                PHA                         ;PUT 0 ON STACK TO INIT CSCH
F46B A07F              LDY     #$7F                ;Y STARTS = 7F
F46D B900FF  CSALOOP   LDA     $FF00,Y             ;GET HI PAGE INTO ACC
F470 990018            STA     ACC,Y               ;$FF00-$FF7F AND $FFF9-$FFFF
F473 88                DEY
F474 C0F8              CPY     #$F8
F476 D0F5              BNE     CSALOOP

F478 A929              LDA     #L(S-CODEDIF)       ;SET UP FOR THE RETURN
F47A 8D0024            STA     CSCMOD1+1-CODEDIF
F47D A924              LDA     #H(S-CODEDIF)
F480 8D0022            STA     CSCMOD1+2-CODEDIF
```

```
F482 201224    CSCSLOOP  JSR     CSCHKDLI-CODEDIF      ;CHECK FOR COMING DLI
F485 68                  PLA                           ;SAVE LOWER STATE
F486 20F623              JSR     CSCHECK-CODEDIF        ;MARCH UP THE CODE
F489 48                  PHA                           ;PUT BACK LOWER STATE
F48A EEFD23              INC     CSCMOD0+2-CODEDIF
F48D ADFD23              LDA     CSCMOD0+2-CODEDIF
F490 C9FF                CMP     #$FF
F492 DOEE                BNE     CSCSLOOP

F494 201224              JSR     CSCHKDLI-CODEDIF       ;CHECK FOR COMING DLI
F497 200924              JSR     CSROTATE-CODEDIF       ;ROTATE THE BITS AROUND  A BIT
F49A 200924              JSR     CSROTATE-CODEDIF       ;ROTATE THE BITS AROUND  A BIT

F49D A92D                LDA     #L(T-CODEDIF)          ;SET UP FOR THE RETURN MARCH
F49F 8D0024              STA     CSCMOD1+1-CODEDIF
F4A2 A924                LDA     #H(T-CODEDIF)
F4A4 8D0124              STA     CSCMOD1+2-CODEDIF
F4A7 CEFD23              DEC     CSCMOD0+2-CODEDIF

F4AA 201224    CSCTLOOP  JSR     CSCHKDLI-CODEDIF       ;CHECK FOR COMING DLI
F4AD 68                  PLA                            ;GET LOWER STATE
F4AE 20F623              JSR     CSCHECK-CODEDIF        ;MARCH BACK DOWN CODE
F4B1 48                  PHA                            ;PUT BACK LOWER STATE
F4B2 CEFD23              DEC     CSCMOD0+2-CODEDIF
F4B5 ADFD23              LDA     CSCMOD0+2-CODEDIF
F4B8 C5EE                CMP     CARTROTM
F4BA 30EE                BCS     CSCTLOOP

F4BC A960                LDA     #$60                   ;DONE WITH DECRYPT, TURN OF G
F4BE 853C                STA     CTRL

F4C0 A277                LDX     #NLEN                  ;'FOLD' THE CHECKSUM TOGETHER
F4C2 BD0018    CSCFLOOP  LDA     ACC,X                  ;AND MOVE IT TO REG2
F4C5 5D5018              EOR     ACC+$50,X
F4C8 5D8818              EOR     ACC+$FF-NLEN,X
F4CB 9D001A              STA     REG2,X
F4CE CA                  DEX
F4CF 10F1                BPL     CSCFLOOP

F4D1 AD001A              LDA     REG2                   ;MAKE SURE IT IS LESS THAN N
F4D4 2907                AND     #NMASK
F4D6 8D001A              STA     REG2

F4D9 A900                LDA     #$00                   ;GET RID OF RANDOM BYTE
F4DB A204                LDX     #RANDBYTE
F4DD 9D001A              STA     REG2,X
F4E0 9D0020              STA     REG1,X

F4E3 A277                LDX     #NLEN                  ;SEE IF THEY CHECK!!
F4E5 BD0020    CSCCLOOP  LDA     REG1,X                 ;THE DECRYPTED SIGNATURE
F4E8 DD001A              CMP     REG2,X                 ;THE COMPUTED CHECKSUM
F4EB D00F3               BNE     NCCHECK
F4ED CA                  DEX
F4EE 10F5                BPL     CSCCLOOP
```

```
F4E0 4CB026                 JMP      SETMARIA-CODEDIF            ;EVERYTHING CHECK ;!!!!!

F4F3 4CB926     NOCHECK     JMP      LOCK2600-CODEDIF           ;DECRYPT FAILED, PUT HIM IN 2


F4F6 A200       CSCHECK     LDX      #$00                       ;ROUTINE TO CHECKSUM ONE PAGE
F4F8 7D0018     CSCLOOP     ADC      ACC,X
F4FB 7D00FF     CSCMOD0     ADC      $FF00,X
F4FE A8                     TAY
F4FF B9D52D     CSCMOD1     LDA      N-CODEDIF,Y
F502 9D0018                 STA      ACC,X
F505 E8                     INX
F506 D0F0                   BNE      CSCLOOP
F508 60                     RTS


F509 A200       CSROTATE    LDX      #$00                       ;ROUTINE TO ROTATE CHECKSUM A
F50B 3E0018     CSRLOOP     ROL      ACC,X
F50E E8                     INX
F50F D0FA                   BNE      CSRLOOP
F511 60                     RTS


F512 08         CSCHKDLI    PHP                                 ;SAVE PROCESSOR STATUS
F513 C6F0                   DEC      KNLSTATE                   ;SEE IF DLI ABOUT TO HAPPEN
F515 100C                   BPL      CSCDOUT                    ;COUNT DOWN OUR TIMER

F517 A902                   LDA      #$02                       ;KERNEL NOT HAPPENED, BUT ABO
F519 8501                   STA      INPTCTRL                   ;DISABLE CART, TURN ON SECURI
F51B A5F0       CSCDLOOP    LDA      KNLSTATE                   ;AND THEN WAIT FOR THE KERNEL
F51D 30FC                   BMI      CSCDLOOP
F51F A916                   LDA      #$16                       ;ACTIVATE SECURITY ROM, TURN O
F521 8501                   STA      INPTCTRL

F523 28         CSCDOUT     PLP                                 ;RESTORE PROCESSOR STATUS
F524 60                     RTS


*     THESE TABLES ARE USED FOR NON-LINEAR CONVERSION
F525 C7         S           DB       $C7,$65,$AB,$CA,$EE,$F7,$83,$09
F526 65
F527 AB
F528 CA
F529 EE
F52A F7
F52B 83
F52C 09
F52D E1                     DB       $E1,$DD,$72,$67
F52E DD
F52F 72
F530 67
```

ssembling DECRYPT.S
------- 65xx Assembler V5.6 --

```
F533 72
F534 55
F535 8E          DB          $8E,$91,$DC,$C5,$81,$BE,$78,$20
F536 91
F537 DC
F538 C5
F539 81
F53A BE
F53B 78
F53C 20
F53D 59          DB          $59,$E7,$E6,$3D,$06,$45,$AF,$C8
F53E 87
F53F E6
F540 3D
F541 06
F542 45
F543 AF
F544 C8
F545 08          DB          $08,$31,$38,$D1,$F3,$73,$64,$A9
F546 31
F547 38
F548 D1
F549 F3
F54A 73
F54B 64
F54C A9
F54D 17          DB          $17,$FC,$34,$87,$A3,$94,$FA,$90
F54E FC
F54F 34
F550 87
F551 A3
F552 94
F553 FA
F554 90
F555 B8          DB          $B8,$ED,$CE,$3B,$5B,$0A,$43,$D9
F556 ED
F557 CE
F558 3B
F559 5B
F55A 0A
F55B 43
F55C D9
F55D F3          DB          $F3,$53,$82,$83,$0D,$6D,$5A,$60
F55E 53
F55F 82
F560 83
F561 0D
F562 6D
F563 5A
F564 60
F565 9D          DB          $9D,$91,$A7,$39,$11,$10,$8C,$E4
F566 91
F567 A7
F568 39
```

ssembling DECRYPT.S   PAGE   9
------- 65xx Assembler V5.6 -------------------------------- Kontron -------

```
F569 11
F56A 10
F56B 8C
F56C E4
F56D 7F            DB      $7F,$80,$41,$E7,$E3,$F6,$56,$26
F56E 80
F56F 41
F570 E7
F571 E3
F572 F6
F573 56
F574 26
F575 35            DB      $35,$EC,$D6,$DF,$0C,$7F,$F4,$9E
F576 EC
F577 D6
F578 DF
F579 0C
F57A 7F
F57B F4
F57C 9E
F57D AC            DB      $AC,$52,$46,$EF,$CF,$BF,$A2,$3F
F57E 52
F57F 46
F580 EF
F581 CF
F582 BF
F583 A2
F584 3F
F585 A4            DB      $A4,$13,$15,$97,$4A,$1C,$80,$42
F586 13
F587 15
F588 97
F589 4A
F58A 1C
F58B 80
F58C 42
F58D 8C            DB      $8C,$81,$05,$58,$80,$18,$77,$28
F58E 81
F58F 05
F590 58
F591 80
F592 18
F593 77
F594 28
F595 02            DB      $02,$3E,$A8,$49,$1A,$6A,$CB,$6E
F596 3E
F597 A8
F598 49
F599 1A
F59A 6A
F59B CB
F59C 6E
F59D 08            DB      $08
```

F59F E8
F5A0 F1
F5A1 4F
F5A2 14
F5A3 79
F5A4 9B
F5A5 D8   DB $D8,$9F,$9B,$57,$19,$F5,$2A,$2D
F5A6 9F
F5A7 9B

F5A8 57
F5A9 19
F5AA F8
F5AB 2A
F5AC 2D
F5AD 76   DB $76,$0E,$E8,$2E,$4B,$F9,$07,$03
F5AE 0E
F5AF E8
F5B0 2E
F5B1 4B
F5B2 F9
F5B3 07
F5B4 03
F5B5 DE   DB $DE,$93,$16,$7E,$D4,$E5,$B2,$F0
F5B6 93
F5B7 16
F5B8 7E
F5B9 D4
F5BA E5
F5BB B2
F5BC F0
F5BD 7D   DB $7D,$7A,$DA,$D2,$A1,$CC,$1D,$E0
F5BE 7A
F5BF DA
F5C0 D2
F5C1 A1
F5C2 CC
F5C3 1D
F5C4 E0
F5C5 5E   DB $5E,$23,$A0,$95,$22,$1E,$36,$85
F5C6 23
F5C7 A0
F5C8 95
F5C9 22
F5CA 1E
F5CB 36
F5CC 85
F5CD FE   DB $FE,$1F,$39,$AA,$89,$96,$AD,$0F
F5CE 1F
F5CF 39
F5D0 AA
F5D1 89
F5D2 96
F5D3 AD
F5D4 0F

Assembling DECRYPT.S   PAGE 11
-------- 65xx Assembler V5.6 ---Kontron --------

```
F5D5 2F          DB      $2F,$C0,$47,$27,$5D,$24,$EA,$C3
F5D6 C0
F5D7 47
F5D8 27
F5D9 5D
F5DA 24
F5DB EA
F5CC C3
F5DD A5          DB      $A5,$F5,$21,$5F,$1B,$40,$8F,$AE
F5DE F5
F5DF 21
F5E0 5F
F5E1 1B
F5E2 40
F5E3 8F
F5E4 AE
F5E5 74          DB      $74,$25,$DD,$C1,$7C,$CD,$A6,$70
F5E6 25
F5E7 DD
F5E8 C1
F5E9 7C
F5EA CD
F5EB A6
F5EC 70
F5ED D7          DB      $D7,$33,$7B,$2C,$75,$BB,$86,$99
F5EE 33
F5EF 7B
F5F0 2C
F5F1 75
F5F2 BB
F5F3 86
F5F4 99
F5F5 BD          DB      $BD,$54,$9A,$6C,$63,$32,$48,$4C
F5F6 54
F5F7 9A
F5F8 6C
F5F9 63
F5FA 32
F5FB 48
F5FC 4C
F5FD 8D          DB      $8D,$BA,$5C,$61,$C4,$4E,$29,$37
F5FE BA
F5FF 5C
F600 61
F601 C4
F602 4E
F603 29
F604 37
F605 12          DB      $12,$C6,$98,$9C,$D5,$69,$6B,$E2
F606 C6
F607 98
F608 9C
```

Assembling DECRYPT.S
-------- 65xx Assembler V5.6 --

```
F60B 68
F60C E2
F60D 04              DB        $04,$4D,$E9,$C2,$88,$3A,$DB,$64
F60E 4D
F60F E9
F610 C2
F611 88
F612 3A
F613 DB
F614 64
F615 01              DB        $01,$44,$6F,$85,$F2,$30,$28,$FD
F616 44
F617 6F
F618 85
F619 F2
F61A 30
F61B 28
F61C FD
F61D 50              DB        $50,$71,$3C,$84,$66,$68,$C9,$03
F61E 71
F61F 3C
F620 B4
F621 66
F622 68
F623 C9
F624 D3
F625 CA              DB        $CA,$83,$C7,$AB,$F7,$65,$09,$EE
F626 83
F627 C7
F628 AB
F629 F7
F62A 65
F62B 09
F62C EE
```

```
       *  METHOD.S         THIS IS A PACKAGE OF ROUTINES THAT DO THE ACTUAL DE
       *
       *  THIS ROUTINE DECRYPTS A SIGNATURE.  THE KEY IS PLACED INTO REG1.
       *  MODULUS IS N.   EVERYTHING IS ASSUMED TO BE SIZE NLEN.
       *             INPUT:  SIG IN $FF80, N, NLEN
       *             OUTPUT: DECRYPTED SIG IN REG1 (LENGTH NLEN)
       *  REG1 = SIG * SIG MOD N   (DECRYPTION FUNCTION)
F62D A277   DECRYPT   LDX       #NLEN
F62F 86E4             STX       SIZE0
F631 86E5             STX       SIZE1

F633 BD80FF  DCLOOP   LDA       $FF80,X
F636 9D0119           STA       REG0+1,X
F639 9D0020           STA       REG1,X
F63C CA               DEX
F63D 10F4             BPL       DCLOOP

F63F A902             LDA       #$02              ;TURN SECURITY ROM BACK ON
```

```
F641 8501                    STA      INPTCTRL
F643 2084F8                  JSR      GRAPHON                 ;TURN GRAPHICS ON

F646 207225       .          JSR      MULTIPLY-CODE0I         ;START WITH THE MULTIPLY
F649 C6F2              :      DEC      KNLTIME                 ;SPEED UP FUJI ROLLING

F64B A277                    LDX      #NLEN                   ;SIZE OF REGISTER
F64D 86E4                    STX      SIZERO                  ;SET UP SIZE FOR DIVIDE
F64F BDD5FE    MDMRLOOP      LDA      N,X                     ;MOVE ACCMOD TO REG0
F652 9D0119                  STA      REG0+1,X                ;STORE IT
F655 CA                      DEX
F656 10F7                    BPL  ,    MDMRLOOP                ;KEEP GOING TILL REGISTER DO

F658 A5E1                    LDA      OFFSETA
F65A 85E3                    STA      SIZEA
F65C 200825                  JSR      DIVIDE-CODE0I           ;AND DO THE DIVIDE
F65F C6F2                    DEC      KNLTIME                 ;SPEED UP FUJI ROLLING

F661 A5E0                    LDA      STARTA                  ;SET UP FOR MOVE
F663 8D6925                  STA      DCMOD0+1-CODE0I
F666 A277                    LDX      #NLEN                   ;PUT ACC INTO REG1
F668           DCMLOOP
F668 BD0018    DCMOD0        LDA      ACC,X
F66B 900020                  STA      REG1,X
F66E CA                      DEX
F66F 10F7                    BPL      DCMLOOP

F671 60                      RTS                              ;ALL DONE, GET CHECKSUM TO C
```

```
*  MULTIPLY TWO NUMBERS -- THE TWO NUMBERS TO BE MULTIPLIED ARE PLACED
*  REG0 AND REG1.  THE RESULT IS LEFT IN ACC.  ACC WILL HAVE A LEADIN
*  (TO MAKE IT EASIER TO USE WITH A MODULO FOLLOWING IT).  REG0 SHOUL
*  AT REG0+1 (SO THE FIRST LOCATION CAN BE USED) THOUGH SIZERO SHO
*  REMAIN UNCHANGED.
*           INPUT:  REG0, REG1, SIZERO, SIZE
*           OUTPUT: ACC, STARTA, OFFSETA
*           USES:   REG0-REG14 EVEN
*  ACC = REG0 * REG1
```

```
F672 203026    MULTIPLY      JSR      SETREGS-CODE0I          ;INITIALIZE REGISTERS
F675 A4E5                    LDY      SIZER1                  ;PREPARE ACCUMULATOR
F677 C8                      INY
F678 84E1                    STY      OFFSETA                 ;OFFSET INTO REG0 AND ACC
F67A 98                      TYA
F67B 18                      CLC
F67C 65E2                    ADC      OFFSETR                 ;GET SIZE TO CLEAR
F67E 48                      PHA                              ;PUSH SIZE OF ACCUMULATOR ON
F67F AA                      TAX
F680 A900                    LDA      #$00
F682 8D6826                  STA      ADDMOD+1-CODE0I         ;WE ONLY USE EVEN REGISTERS
F685 9D0018    MULCALP       STA      ACC,X                   ;CLEAR ACC
F688 CA                      DEX
F689 D0FA                    BNE      MULCALP
F68B 8D0018                  STA      ACC                     ;CLEAR LEADING BYTE
```

```
ssembling DECRYPT.S
-------- 65xx Assembler V5.6 -------------------------------------------


F68E C8                       INY                                 ;Y STILL        FSETA, INC FOR
F68F 8C6526                   STY    ADACMOD0+1-CODEDIF           ;MODIFY          CUMULATOR INDEX
F692 8C6B26                   STY    ADACMOD1+1-CODEDIF
F695 8C7326                   STY    ADACMOD2+1-CODEDIF
F698 8C7826                   STY    ADACMOD3+1-CODEDIF

F69B A200        MULLOOP0     LDX    #$00                         ;RESET BIT OFFSET
F69D CE6526                   DEC    ADACMOD0+1-CODEDIF           ;MODIFY          CUMULATOR INDEX
F6A0 CE6B26                   DEC    ADACMOD1+1-CODEDIF
F6A3 CE7326                   DEC    ADACMOD2+1-CODEDIF
F6A6 CE7826                   DEC    ADACMOD3+1-CODEDIF
F6A9 C6E1                     DEC    OFFSETA                      ;GO TO NEXT REG1 BYTE
F6AB 301B                     BMI    MULOUT
F6AD A4E1        MULLOOP      LDY    OFFSETA                      ;GET OFFSET INTO REG0
F6AF B90020                   LDA    REG1,Y                       ;SEE WHAT OUR 'CURRENT BIT'
F6B2 3DD025                   AND    MULTMASK-CODEDIF,X
F6B5 F009                     BEQ    MULNEXT
F6B7 8D5926                   LDA    HSERDER-CODEDIF,Y            ;IT IS A 1, ADD IN APPROPRIA
F6BA 606926                   STA    ADDMOD+2-CODEDIF
F6BD 206126                   JSR    MULTADD-CODEDIF

F6C0 E8         MULNEXT      INX                                 ;GO TO NEXT BIT IN BYTE
F6C1 E008                    CPX    #$08
F6C3 30E8                    BMI    MULLOOP
F6C5 4C9B25                  JMP    MULLOOP0-CODEDIF

F6C8 68         MULOUT       PLA                                 ;GET SIZE OF ACCUMULATOR BAC
F6C9 85E1                    STA    OFFSETA
F6CB A901                    LDA    #$01                         ;STARTING BYTE OF ACCUMULATO
F6CD 85E0                    STA    STARTA
F6CF 60                      RTS

F6D0 01         MULTMASK     DB     $01,$02,$04,$08,$10,$20,$40,$80
F6D1 02
F6D2 04
F6D3 08
F6D4 10
F6D5 20
F6D6 40
F6D7 80
```

```
   *  DIVIDE TWO NUMBERS - THE ACCUMULATOR IS DIVIDED BY REG0.  THE REMA
   *  PLACED IN REG1.  THE ACCUMULATOR MUST START OUT WITH A NULL HI BYT
   *  (IE: INITIAL STARTA OF 1, THE FIRST BYTE OF ACC IS ZEROED HERE).
   *  AS IN MULTIPLY, REG0 SHOULD START AT REG0+1.
   *        INPUT:  ACC, REG0, SIZERO, SIZEA
   *        OUTPUT: ACC, STARTA, OFFSETA
   *        USES:   REG2-REG14 EVEN
   *  REG1 = REM(ACC/REG1)
F6D8 203026     DIVIDE       JSR    SETREGS-CODEDIF              ;SET SHIFTED REGISTERS

F6DB A5E3                    LDA    SIZEA
```

ssembling DECRYPT.S
------- 65xx Assembler V5.6 ---------------------------------------

```
F6DD 38                    SEC
F6DE C5E4                  SBC        SIZERO
F6E0 85E0                  STA        STARTA
F6E2 85E1                  STA        OFFSETA              ;THIS IS THE START OF WHAT IS
                                                           ;THIS IS NUMBER OF TIMES LOOP

F6F4 A200                  LDX        #$00
F6F6 8E0018                STX        ACC                  ;MAKE SURE LEADING BYTE IS Z
F6E9 8E8626                STX        SUBMOD+1-CODEDIF      ;THE ONLY USE EVEN REGISTER
F6EC 8EA326                STX        CMPMOD+1-CODEDIF
F6EF CA                    DEX                             ;THESE WILL BE INCREMENTED ON
F6F0 8EA026                STX        CMACMOD+1-CODEDIF
F6F3 8E8326                STX        SBACMOD0+1-CODEDIF
F6F6 8E8926                STX        SBACMOD1+1-CODEDIF
F6F9 8E9126                STX        SBACMOD2+1-CODEDIF
F6FC 8E9626                STX        SBACMOD3+1-CODEDIF

6FF A207     DIVLOOP0      LDX        #$07                 ;RESET BIT OFFSET
701 EEA026                 INC        CMACMOD+1-CODEDIF
704 EE8326                 INC        SBACMOD0+1-CODEDIF
707 EE8926                 INC        SBACMOD1+1-CODEDIF
70A EE9126                 INC        SBACMOD2+1-CODEDIF
70D EE9626                 INC        SBACMOD3+1-CODEDIF
710 C6E1                   DEC        OFFSETA
712 3017                   BMI        DIVOUT
714 BD5926    DIVLOOP       LDA        HSEROFF-CODEDIF,X     ;DO MODIFICATIONS FOR COMPARE
717 8D8726                 STA        SUBMOD+2-CODEDIF
71A 8DA426                 STA        CMPMOD+2-CODEDIF
71D 209D26                 JSR        DIVCOMP-CODEDIF       ;SEE IF REGISTER IS LARGER
720 9003                   BCC        DIVNEXT              ;  IF SO, DO NOTHING
722 207F26                 JSR        DIVSUB-CODEDIF        ;AND DO THE SUBTRACT

725 CA       DIVNEXT       DEX                             ;GO TO NEXT BIT IN BYTE
726 10EC                   BPL        DIVLOOP
728 4CFF25                 JMP        DIVLOOP0-CODEDIF

72B A5E3     DIVOUT        LDA        SIZEA
72D 85E1                   STA        OFFSETA              ;SET LAST RETURN VALUE
72F 60                     RTS                             ;THIS IS THE END OF WHAT IS L
```

```
             *  THIS ROUTINE SETS UP REGISTERS FOR MULTIPLY AND DIVIDE.  THE REGIST
             *  IS SHIFTED LEFT 7 TIMES, WITH THE INTERMEDIATE FORMS LEFT IN REG2-R
             *  (IN ORDER OF HOW MUCH THEY HAVE BEEN SHIFTED).  IT IS ASSUMED THAT
             *  A LEADING ZERO.  THE LEADING BYTE IS ZEROED SO WHOEVER LOADS REG0 I
             *  HAVE TO ZERO IT.  NOTE THAT SIZERO SHOULD NOT INCLUDE THE LEADING Z
             *          INPUT:  REG0, SIZERO
             *          OUTPUT: REG0-14 EVEN, OFFSETR (INDEX INTO ABOVE)
30 A6E4      SETREGS       LDX        SIZERO               ;OFFSETR = SIZERO+1
32 E8                     INX
33 86E2                   STX        OFFSETR

  A000                    LDY        #$00
  8C0019                  STY        REG0                 ;CLEAR THE EXTRA BYTE
```

```
ssembling DECRYPT.S
------- 65xx Assembler V5.6 ---------------------------------------------------------

F73D 8D4C26              STA      SERSMOD1+2-CODEDIF
F740 C8                  INY                                    ;GO TO NEXT
F741 895926              LDA      HSEROFF-CODEDIF,Y
F744 8D5026              STA      SERSMOD2+2-CODEDIF

F747 A6E2                LDX      OFFSETR                        ;DO A LEFT SHIFT OF THE REGIS
F749 13                  CLC
F74A            SERSLOOP
F74A 8D0019    SERSMOD1  LDA      REGO,X                         ;GET DATA
F74D 2A                  ROL      A                              ;ROTATE IT
F74E 9D0019    SERSMOD2  STA      REGO,X                         ;STORE IT
F751 CA                  DEX
F752 10F6                BPL      SERSLOOP

F754 C007                CPY      #$07
F756 30E2                BMI      SERLOOP

F758 60        SEROUT    RTS


               *  OFFSET TABLES AND MASK BYTES USED BY SETREGS
F759 19        HSEROFF   DB       H(REGO),H(REG2),H(REG4),H(REG6)
F75A 1A
F75B 1B
F75C 1C
F75D 1D                  DB       H(REG8),H(REG10),H(REG12),H(REG14)
F75E 1E
F75F 1F
F760 21
```

```
               *  ADD TWO NUMBERS - CALLER MODIFIES ADDMOD+1,ADDMOD+2 TO THE ADDRESS
               *  REGISTER TO BE ADDED IN.   CALLER MODIFIES ADACMOD0..3+1 FOR OFFSET
               *  ACCUMULATOR.
               *           INPUT:   ACC, REGN, OFFSETR, X
               *           OUTPUT:  ACC
               *       ACC = ACC + REGN
F761 A4E2      MULTADD   LDY      OFFSETR                        ;START AT THE END OF THE REGIS
F763 18                  CLC
F764           ADDLOOP
F764 B90018    ADACMOD0  LDA      ACC,Y                          ;ADD THE REGISTER TO THE ACCU
F767 790019    ADDMOD    ADC      REGO,Y
F76A 990018    ADACMOD1  STA      ACC,Y
F76D 88                  DEY
F76E 10F4                BPL      ADDLOOP                        ;KEEP GOING TILL REGISTER EXHA

F770 900C      ADDLOOP2  BCC      ADDOUT                         ;IF CARRY CLEAR, ALL DONE
F772 B90017    ADACMOD2  LDA      ACC-$100,Y                     ;PROPAGATE CARRY (Y IS WRAPPED
F775 6900                ADC      #$00
F777 990017    ADACMOD3  STA      ACC-$100,Y
F77A 88                  DEY
F77B 4C7026              JMP      ADDLOOP2-CODEDIF
F77E 60        ADDOUT    RTS
```

Assembling DECRYPT.S   PAGE 17
------- 65xx Assembler V5.6 ----------------------------- Kontron ---------

```
            *  SUBTRACT TWO NUMBERS - CALLER MODIFIES SUBMOD+1,SUBMOD+2 TO THE AD
            *  THE REGISTER TO BE SUBTRACTED.  CALLER MODIFIES SBACMOD0..3+1 TO T
            *  INTO THE ACCUMULATOR.
            *          INPUT:  ACC, REGN, OFFSETR, X
            *          OUTPUT: ACC
            *  ACC = ACC - REGN
F77F A4E2   DIVSUB   LDY      OFFSETR                    ;START AT THE END OF THE REG
F781 38              SEC
F782        SUBLOOP
F782 B90018 SBACMOD0 LDA      ACC,Y                      ;ADD THE REGISTER TO THE ACC
F785 F90019 SUBMOD   SBC      REG0,Y
F788 990018 SBACMOD1 STA      ACC,Y
F78B 88              DEY
F78C 10F4            BPL      SUBLOOP                    ;KEEP GOING TILL REGISTER EX

F78E B00C   SUBLOOP2 BCS      SUBOUT                     ;IF CARRY CLEAR, ALL DONE
F790 B90017 SBACMOD2 LDA      ACC-$100,Y                 ;PROPAGATE CARRY (Y IS WRAPP
F793 E900            SBC      #$00
F795 990017 SBACMOD3 STA      ACC-$100,Y
F798 88              DEY
F799 4C8E26          JMP      SUBLOOP2-CODEDIF
F79C 60     SUBOUT   RTS

            *  COMPARE TWO NUMBERS - CALLER MODIFIES CMPMOD+1,CMPMOD+2 TO THE ADD
            *  THE REGISTER TO BE SUBTRACTED.  CMACMOD+1 IS MODIFIED FOR THE OFFS
            *  THE ACCUMULATOR.
            *          INPUT:  ACC, REGN, OFFSETR
            *          OUTPUT: CARRY SET IF REGISTER LESS THAN ACCUMULATOR
F79D A000   DIVCOMP  LDY      #$00                       ;START AT THE TOP OF THE REG
F79F        CMPLOOP
F79F B90018 CMACMOD  LDA      ACC,Y                      ;ADD THE REGISTER TO THE ACC
F7A2 D90019 CMPMOD   CMP      REG0,Y
F7A5 F001            BEQ      CMPNEXT
F7A7 60     CMPOUT   RTS                                 ;OUT

F7A8 C4E2   CMPNEXT  CPY      OFFSETR                    ;WE HAVE TO LOOK AT ANOTHER
F7AA F0FB            BEQ      CMPOUT                     ;HAVE MADE IT TO THE END O
F7AC C8              INY                                 ;NO, GO ANOTHER BYTE
F7AD 4C9F26          JMP      CMPLOOP-CODEDIF

            *  VECTOR.S           WE HAVE DETERMINED VALIDITY, VECTOR TO CART IN 2600
            *                     3600 MODE

F7B0 A216   SETMARIA LDX      #$16                                      ;IN 3600 MODE, CART
F7B2 8601            STX      INPTCTRL
F7B4 9A              TXS
F7B5 F8              SED
F7B6 6CFCFF          JMP      ($FFFC)                    ;VECTOR INTO THE CART IN 360
```

ssembling DECRYPT.S
------- 65xx Assembler V5.6 --

```
F7B9 A902      LOCK2600    LDA     #$02
F7BB 8501                  STA     INPTCTRL                    ;TURN SECU      UN ON
F7BD A27F                  LDX     #$7F                        ;LOCK CAR        00 MODE, CART
F7BF BDCBF7    L2LOOP      LDA     SYNC,X                      ;MOVE CODE        RAM
F7C2 9D8004                STA     $480,X                      ;MOVE INTO 6532 RAM
F7C5 CA                    DEX
F7C6 10F7                  BPL     L2LOOP
F7C8 4C8004                JMP     $480                        ;AND EXECUTE OUT OF RAM

F7CB A900      SYNC    .   LDA     #0
F7CD AA                    TAX
F7CE 9501      ZEROLP      STA     1,X
F7D0 E8                    INX
F7D1 E02C                  CPX     #$2C
F7D3 D0F9                  BNE     ZEROLP
F7D5 A904                  LDA     #4
F7D7 8502                  STA     2
F7D9 A904                  LDA     #4
F7DB EA                    NOP
F7DC 301F                  BMI     
F7DE A204                  LDX     #4
F7E0 CA        DEX         DEX
F7E1 10FD                  BPL     DEX
F7E3 9A                    TXS
F7E4 8D1001                STA     $110
F7E7 20C704                JSR     DUMMY+1-SYNC+$480
F7EA 20C704                JSR     DUMMY+1-SYNC+$480
F7ED 8511                  STA     $11
F7EF 8502                  STA     2
F7F1 851B                  STA     $1B
F7F3 851C                  STA     $1C
F7F5 850F                  STA     $F
F7F7 8502                  STA     2
F7F9 2403                  BIT     3
F7FB 3009                  BMI     OUT
F7FD A902      E           LDA     #2
F7FF 8509                  STA     9
F801 8D12F1                STA     $F112
F804 D00E                  BNE     DONE
F806 2402      OUT         BIT     2
F808 300A                  BMI     DONE
F80A A902                  LDA     #2
F80C 8506                  STA     6
F80E 8D18F1                STA     $F118
F811 8D60F4    DUMMY       STA     $F460
F814 A9FD      DONE        LDA     #$FD
F816 8509                  STA     9
F818 6CFCFF                JMP     ($FFFC)


F81B CA        ENDROM      NOP                                 ;END OF FIRST PART OF ROM

               *   TESTS.S          RAM AND CPU TESTS - IF EITHER ARE BAD, DECRYPTION WI
               *                    THUS, THESE MUST BE TESTED FIRST
```

```
                   * ***** NOTE: *****  THE FOLLOWING INSTRUCTIONS ARE NOT TESTED BY THE
                   *  AND THUS SHOULD NOT BE USED IN THE VALIDATION/DECRYPTION CODE:
                   *        BRK, RTI, PLP, PHP, CLV, SEV, BVC, BVS, CLD, SED, BIT, SEI,


                   *  TEST FAILURE MODES:
0000   BADCPU    EQU     $00                              ;CPU ERROR
0001   BAD6116A  EQU     $01                              ;ERROR IN RAM $2000-$27FF
0002   BAD6116B  EQU     $02                              ;ERROR IN RAM $1800-$1FFF
0003   BADRAM    EQU     $03                              ;CAN'T GET TO ANY OF THE RAM
0004   BADMARIA  EQU     $04                              ;MARIA SHADOWING NOT WORKING
0005   BADVALID  EQU     $05                              ;BAD VALIDATION OR DECRYPTION


F880                     ORG     ROMCODE2

F880 A91D   FATAL$  LDA     #$1D                          ;THERE HAS BEEN SOMETHING BAD
F882 8501           STA     INPTCTRL                      ;LOCK IN 2600 MODE, TEST CART

F884 78     MAIN    SEI                                   ;INITIALIZE
F885 D8             CLD

F886 A902           LDA     #$02                          ;PUT BASE UNIT INTO MARIA ENA
F888 8501           STA     INPTCTRL

F88A A9FB           LDA     #H(ENDDLI)                    ;WESTBERG SUX
F88C 85F5           STA     DLIADDR+1                     ;WESTBERG SUX
F88E A912           LDA     #L(ENDDLI)                    ;WESTBERG SUX
F890 85F4           STA     DLIADDR                       ;WESTBERG SUX

F892 A97F           LDA     #$7F
F894 853C           STA     CTRL                          ;TURN OFF DMA
F896 A900           LDA     #$00
F898 8520           STA     BACKGRND                      ;BACKGROUND COLOR TO BLACK


                   *  ***** RAM TESTS *****

                   *  A SIMPLE RAM TEST TO CHECK PAGES $2000 AND $2100 IS DONE FIRST TO (
                   *  ENOUGH RAM TO TEST OUT THE CPU (SHADOWED TO PAGES $0000 AND $0100).
                   *  THE CPU TEST, A FULL RAM TEST IS DONE.

                   *  EARLY RAM TEST, JUST CHECK OUT OUR TWO PAGES USING MINIMAL INSTRUCT
                                                          ;TEST OUT 4 PATTERNS OF RAM
F89A A205   RAMCHECK LDX    #$05
F89C BD1DF9 RCAGAIN  LDA    RAMPAT,X
F89F A000            LDY    #$00
F8A1 990020 RCLOOP   STA    $2000,Y                       ;CHECK ZERO PAGE
F8A4 D90020          CMP    $2000,Y
F8A7 D027            BNE    CHKRAMB
F8A9 990021          STA    $2100,Y                       ;CHECK PAGE 1
F8AC D90021          CMP    $2100,Y
F8AF D01F            BNE    CHKRAMB
F8B1 88             DEY
```

ssembling DECRYPT.S
-------- 65xx Assembler V5.6 ---

```
F8B2 DOED                    BNE      RCLOOP
F8B4 CA                      DEX
F8B5 10E5                    BPL      RCAGAIN
```

```
              * SEE IF MARIA SHADOWING WORKS                      ;A SIMPL    TO SEE IF SHAD

F8B7 A943                    LDA      #$43
F8B9 8D8C20                  STA      $208C
F8BC C580                    CMP      $0080
F8BE D00B                    BNE      MARIAERR
F8C0 8D8021                  STA      $2180
F8C3 CD8001                  CMP      $0180
F8C6 D003                    BNE      MARIAERR

F8C8 4C38F9                  JMP      CPUTEST                      ;IF SHAD       AD RAM WORKS,
```

```
              * RAM FAILURE ROUTINES

F8CB A004       MARIAERR     LDY      #BADMARIA                    ;MARIA SHADOWING BAD
F8CD 4C80F8                  JMP      FATAL$

F8D0 8D0018     CHKRAMB      STA      $1800                        ;RAMA HAS FAILED IN SIMPLE TE
F8D3 CD0018                  CMP      $1800                        ;   TO SEE IF ANY RAM WORKS
F8D6 D00A                    BNE      RAMERR

F8D8 A001       RAMAERR      LDY      #BAD5116A                    ;BAD RAM CHIP - $2000-$27FF
F8DA 4C80F8                  JMP      FATAL$

F8DD A002       RAMBERR      LDY      #BAD5116B                    ;BAD RAM CHIP - $1800-$1FFF
F8DF 4C80F8                  JMP      FATAL$

F8E2 A003       RAMERR       LDY      #BADRAM                      ;ALL RAM BAD - COULD BE ANOTH
F8E4 4C80F8                  JMP      FATAL$


              * A FULL RAM TEST, TO BE DONE AFTER THE CPU TEST SUCCEEDS

F8E7 A900       RAMTEST      LDA      #$00                         ;SET UP STATE TO MARCH THROUG
F8E9 85F0                    STA      $F0                          ;(F0) = $2000
F8EB 85F2                    STA      $F2                          ;(F2) = $1800
F8ED A007                    LDY      #$07                         ;NUMBER OF PAGES TO CHECK
F8EF 84F4                    STY      $F4

F8F1 B923F9     RTPAGE       LDA      RAMAPAGE,Y                   ;SET UP RAM A PAGE TO CHECK
F8F4 85F1                    STA      $F1
F8F6 B923F9                  LDA      RAMBPAGE,Y                   ;SET UP RAM B PAGE TO CHECK
F8F9 85F3                    STA      $F3
F8FB A205                    LDX      #$05                         ;NUMBER OF RAM PATTERNS TO C

F8FD BD1DF9     RTPAT        LDA      RAMPAT,X                     ;GET RAM PATTERN
F900 A000                    LDY      #$00                         ;INITIALIZE INDEX

F902 91F0       RTLOOP       STA      ($F0),Y                      ;CHECK RAM A
F904 D1F0                    CMP      ($F0),Y
```

```
 F906 D0D0          BNE     RAMAERR
 F908 91F2          STA     ($F2),Y              ;CHECK RAM B
 F90A D1F2          CMP     ($F2),Y
 F90C D0CF          BNE     RAMBERR
 F90E 88            DEY
 F90F D0F1          BNE     RTLOOP

 F911 CA            DEX
 F912 10E9          BPL     RTPAT

 F914 C6F4          DEC     $F4                  ;ONE LESS PAGE
 F916 A4F4          LDY     $F4
 F918 10D7          BPL     RTPAGE
 F91A 4C17F8        JMP     STARTVND             ;START THE VALIDATION AND DE


 F91D 00    RAMPAT    DB    $00,$FF,$55,$AA,$69,$0F           ;PATTERNS FOR RA
 F91E FF
 F91F 55
 F920 AA
 F921 69
 F922 0F
 F923 22    RAMAPAGE  DB    $22,$23,$24,$25,$26,$27,$22,$23   ;HI BYTES OF RAM
 F924 23
 F925 24
 F926 25
 F927 26
 F928 27
 F929 22
 F92A 23

 F92B 18    RAMBPAGE  DB    $18,$19,$1A,$1B,$1C,$1D,$1E,$1F   ;HI BYTES OF RAM
 F92C 19
 F92D 1A
 F92E 1B
 F92F 1C
 F930 1D
 F931 1E
 F932 1F
```

```
            *  ***** CPU TESTS *****


 F933      IRQINT                              ;IF WE GET AN IRQ, IT IS A C
 F933 A000 CPUERR    LDY     #BADCPU           ;CPU ERROR
 F935 4C80F8         JMP     FATALS

            * CPU TEST, METHODICALLY CHECK ALL INSTRUCTIONS, ADDRESSING MODES, AN
            *  BITS THAT THE DECRYPTION WILL BE USING

 F938 A9AA CPUTEST   LDA     #$AA              ;FIRST TEST OUT LDA AND BRA
 F93A F0F7           BEQ     CPUERR            ;CHECK BEQ FAIL
 F93C 10F5           BPL     CPUERR            ;CHECK BPL FAIL
 F93E 3003           BMI     CTA               ;CHECK BMI SUCCEED
```

Assembling DECRYPT.S
-------- 65xx Assembler V5.6 ----------------------------------

```
F940  4C33F9              JMP      CPUERR
F943  0003       CTA      BNE      CTB              ;CHECK BNE SUCCEED
F945  4C33F9              JMP      CPUERR
F948  85AA       CTB      STA      $AA              ;STORE TO ADDRESS $AA = AA
F94A  C5AA                CMP      $AA              ;SEE IF IT ADDRESSES AND COMP
F94C  D0E5                BNE      CPUERR

F94E  A900                LDA      #$00             ;TEST ALTERNATE POLARITY
F950  D0E1                BNE      CPUERR           ;CHECK BNE FAIL
F952  30DF                BMI      CPUERR           ;CHECK BMI FAIL
F954  1003                BPL      CTC              ;CHECK BPL SUCCEED
F956  4C33F9              JMP      CPUERR
F959  F003       CTC      BEQ      CTD              ;CHECK BEQ SUCCEED
F95B  4C33F9              JMP      CPUERR
F95E  C900       CTD      CMP      #$00
F960  D001                BNE      CPUERR           ;CHECK CMP EQ
F962  90CF                BCC      CPUERR           ;CHECK BCC FAIL
F964  B003                BCS      CTE              ;CHECK BCS SUCCEED
F966  4C33F9              JMP      CPUERR
F969  C901       CTE      CMP      #$01             ;CHECK BCS FAIL
F96B  30C6                BCS      CPUERR
F96D  9003                BCC      CTF              ;CHECK BCC SUCCEED
F96F  4C33F9              JMP      CPUERR


F972  A255       CTF      LDX      #$55             ;TEST X AND Y, LOADS, STORES
F974  E056                CPX      #$56             ;CHECK CPX
F976  F0BB                BEQ      CPUERR
F978  8EAA01              STX      $1AA             ;CHECK STX - $1AA = 55
F97B  ECAA01              CPX      $1AA             ;CHECK CPX
F97E  D0B3                BNE      CPUERR
F980  A4AA                LDY      $AA              ;CHECK LDY, - Y = AA
F982  C0AB                CPY      #$AB
F984  F0AD                BEQ      CPUERR
F986  8C5501              STY      $155             ;CHECK STY - $155 = AA
F989  CC5501              CPY      $155             ;CHECK CPY
F98C  D0A5                BNE      CPUERR

F98E  CA                  DEX                       ;CHECK TRANSFER DATA PATHS A
F98F  9A                  TXS                       ;TO POINT TO $155, S MUST BE
F990  E8                  INX
F991  68                  PLA                       ;S HAS 55, A = $155 (= AA)
F992  C9AA                CMP      #$AA             ;TEST TXS AND PLA
F994  D055                BNE      CPUERRO
F996  8A                  TXA                       ;A = 55
F997  48                  PHA                       ;$155 = 55
F998  EC5501              CPX      $155             ;TEST TXA AND PHA
F99B  D04E                BNE      CPUERRO
F99D  98                  TYA                       ;A = AA
F99E  C9AA                CMP      #$AA             ;TEST TYA
F9A0  D049                BNE      CPUERRO
F9A2  AA                  TAX                       ;X = AA
F9A3  BD0001              LDA      $100,X           ;NORM,X - A = $1AA (= 55)
```

Assembling DECRYPT.S   PAGE 23
---------- 65xx Assembler V5.6 --------------------------------------- Kontron --------

```
 F9A6 A8                TAY                              ;Y = 55
 F9A7 C055              CPY     #$55                     ; TEST NORM,X, TAX, TAY
 F9A9 D040              BNE     CPUERRO


                                                        ;TEST ADDRESSING MODES (NORM,
 F9AB B500              LDA     $00,X                    ;ZP,X   A = $AA (= AA)
 F9AD C5AA              CMP     $AA                      ;ZP, TEST ZP AND ZP,X
 F9AF D03A              BNE     CPUERRO
 F9B1 C9AA              CMP     #$AA                     ; TEST ZP AND ZP,X
 F9B3 D036              BNE     CPUERRO
 F9B5 49FF              EOR     #$FF                     ;A = 55
 F9B7 990000            STA     $00,Y                    ;ZP,Y = $55 = 55
 F9BA C555              CMP     $55
 F9BC D02D              BNE     CPUERRO
 F9BE D90001            CMP     $100,Y                   ;NORM,Y ($155)
 F9C1 D028              BNE     CPUERRO
 F9C3 DDAB20            CMP     $20AB,X                  ;NORM,Y W/WRAP ($155)
 F9C6 D023              BNE     CPUERRO

 F9C8 A920              LDA     #$20                     ;SET UP ADDR, TEST (IND,X), (
 F9CA 85F1              STA     $F1
 F9CC A9CC              LDA     #$CC
 F9CE 85F0            . STA     $F0                      ;($F0) = $20CC (WHICH IS $CC)
 F9D0 8146   :         STA     ($F0-$AA,X)              ;( IND,X) - $CC = CC
 F9D2 C5CC              CMP     $CC
 F9D4 D015              BNE     CPUERRO
 F9D6 91F0              STA     ($F0),Y                  ;(IND),Y - $2121 = CC
 F9D8 CD2121            CMP     $2121
 F9DB D00E              BNE     CPUERRO
 F9DD A9EE              LDA     #L(CTCONT)               ;TEST (IND), ONLY JMP USES
 F9DF 85F0              STA     $F0
 F9E1 A9F9              LDA     #H(CTCONT)
 F9E3 85F1              STA     $F1
 F9E5 6CF000            JMP     ($F0)                    ;(IND)
 F9E8 4CEBF9            JMP     CPUERRO


 F9EB 4C33F9   CPUERRO  JMP    CPUERR                    ;ANOTHER CPUERR


 F9EL A955     CTCONT   LDA    #$55                      ;TEST ADDER
 F9F0 18                CLC
 F9F1 6955              ADC    #$55                      ;55 + 55 = $AA
 F9F3 EA                NOP                              ;NOP, MAKE SURE IT DOESN'T A
 F9F4 B0F5              BCS    CPUERRO
 F9F6 10F3              BPL    CPUERRO
 F9F8 F0F1              BEQ    CPUERRO
 F9FA C9AA              CMP    #$AA
 F9FC D0ED              BNE    CPUERRO
                 ;     SEC
 F9FE 6955              ADC    #$55                      ;AA + 55 + C = 0 + C
 FA00 EA                NOP                              ;NOP, MAKE SURE IT DOESN'T A
 FA01 90E8              BCC    CPUERRO
```

Assembling DECRYPT.S
------- 65xx Assembler V5.6 --------------------------------

```
FA03 30E6              BMI     CPUERRO
FA05 D0E4              BNE     CPUERRO

        ;             SEC                      ; TEST S
FA07 E955             SBC     #$55             ; 0 - 55
FA09 B0E0             BCS     CPUERRO
FA0B 10DE             BPL     CPUERRO
FA0D F0DC             BEQ     CPUERRO
FA0F C9AB             CMP     #$AB
FA11 D0D8             BNE     CPUERRO
FA13 18               CLC
FA14 E9AA             SBC     #$AA             ; AB - AA
FA16 90D3             BCC     CPUERRO
FA18 3001             BMI     CPUERRO
FA1A D0CF             BNE     CPUERRO


FA1C A9FF             LDA     #$FF             ; TEST OUT INCREMENTS AND DEC
FA1E AA               TAX                      ; X = FF
FA1F E8               INX                      ; TEST INX - X = 0
FA20 D036             BNE     CPUERR1
FA22 CA               DEX                      ; TEST DEX - X = FF
FA23 F033             BEQ     CPUERR1
FA25 1031             BPL     CPUERR1
FA27 E0FF             CPX     #$FF
FA29 D02D             BNE     CPUERR1
FA2B A8               TAY                      ; Y = FF
FA2C C8               INY                      ; TEST INY - Y = 0
FA2D D029             BNE     CPUERR1
FA2F 88               DEY                      ; TEST DEY - Y = FF
FA30 F026             BEQ     CPUERR1
FA32 C8               INY                      ; Y = 0
FA33 D023             BNE     CPUERR1
FA35 85F0             STA     $F0              ; $F0 = FF
FA37 E6F0             INC     $F0              ; TEST INC - $F0 = 0
FA39 D01D             BNE     CPUERR1
FA3B C4F0             CPY     $F0
FA3D D019             BNE     CPUERR1
FA3F C6F0             DEC     $F0              ; TEST DEC - $F0 - FF
FA41 F015             BEQ     CPUERR1
FA43 C5F0             CMP     $F0
FA45 D011             BNE     CPUERR1


FA47 A94A             LDA     #$AA             ; TEST SHIFTS AND ROTATES - 1
FA49 18               CLC                      ; C = 0
FA4A 2A               ROL     A                ; 01010100, C=1
FA4B 2A               ROL     A                ; 10101001, C=0
FA4C 2A               ROL     A                ; 01010010, C=1
FA4D C952             CMP     #$52             ; 01010010
FA4F D0C7             BNE     CPUERR1

        ;             SEC                      ; C = 1
FA51 6A               ROR     A                ; 10101001, C=0
```

Assembling DECRYPT.S   PAGE   25
-------- 65xx Assembler V5.6 ---------------------------------- Kontron --------

```
 FA52 6A                    ROR      A                 ;01010100, C=1
 FA53 6A                    ROR      A                 ;10101010, C=0
 FA54 C9AA                  CMP      #$AA              ;10101010
 FA56 F003                  BEQ      CTSHIFT

 FA58 4C33F9    CPUERR1     JMP      CPUERR            ;ANOTHER CPUERR

 FA5B 0A        CTSHIFT     ASL      A                 ;01010100, C=1
 FA5C 90FA                  BCC      CPUERR1
 FA5E 0A                    ASL      A                 ;10101000, C=0
 FA5F B0F7                  BCS      CPUERR1
 FA61 0A                    ASL      A                 ;01010000, C=1
 FA62 C950                  CMP      #$50
 FA64 D0F2                  BNE      CPUERR1

 FA66 4905                  EOR      #$05              ;01010101
 FA68 4A                    LSR      A                 ;00101010, C=1
 FA69 90ED                  BCC      CPUERR1
 FA6B 4A                    LSR      A                 ;00010101, C=0
 FA6C B0EA                  BCS      CPUERR1
 FA6E 4A                    LSR      A                 ;00001010, C=1
 FA6F C90A                  CMP      #$0A
 FA71 D0E5                  BNE      CPUERR1

 FA73 A955                  LDA      #$55              ;TEST LOGICAL OPERATIONS
 FA75 091B                  ORA      #$1B              ;TEST OR - A = 5F
 FA77 C95F                  CMP      #$5F
 FA79 D0DD                  BNE      CPUERR1
 FA7B 2955                  AND      #$55              ;A = 55
 FA7D 291B                  AND      #$1B              ;TEST AND - A = 11
 FA7F C911                  CMP      #$11
 FA81 D0D5                  BNE      CPUERR1
 FA83 0955                  ORA      #$55              ;A = 55
 FA85 491B                  EOR      #$1B              ;TEST EOR
 FA87 C94E                  CMP      #$4E
 FA89 D0CD                  BNE      CPUERR1

 FA8B 2091FA                JSR      CTJSR             ;GRAND FINALE, TEST JSR, S =
 FA8E 4C58FA    CTJSRRET    JMP      CPUERR1           ;NO GOOD IF WE DIDN'T JSR

 FA91 BA        CTJSR       TSX                        ;SEE WHERE STACK IS
 FA92 E052                  CPX      #$52
 FA94 D0C2                  BNE      CPUERR1
 FA96 68                    PLA                        ;GET RETURN ADDRESS
 FA97 C98D                  CMP      #L(CTJSRRET-1)
 FA99 D0BD                  BNE      CPUERR1
 FA9B 68                    PLA
 FA9C C9FA                  CMP      #H(CTJSRRET-1)
 FA9E D0B8                  BNE      CPUERR1
 FAA0 A9F8                  LDA      #H(RAMTEST-1)     ;PUT START OF CODE AS RETURN
```

Assembling DECRYPT.S
-------- 65xx Assembler V5.6 ---

```
     FAA2 48                PHA
     FAA3 A9E6              LDA      #L(RAMTEST-1)
     FAA5 48                PHA
     FAA6 60                RTS                              ;DO IT
     FAA7 4C58FA            JMP      CPUERR1                 ;AGAIN, NO GOOD IF WE DI


            *  KERNEL.S        DLI ROUTINES FOR THE SECURITY ROM

            *  OUR DLI HANDLER
     FAAA 8A      DLI       TXA                              ;STACK REGISTERS, A ALRE
     FAAB 48                PHA

     FAAC A943              LDA      #$43
     FAAE 853C              STA      CTRL
     FAB0 A20F              LDX      #$0F
     FAB2 A5EF              LDA      FUJICOLR
     FAB4 8522              STA      POC2                    ;INITIALIZE COLOR
     FAB6 24F3              BIT      KNLOFSET                ;FIGURE OUT STAGGERING
     FAB8 5006              BVC      DFJMP1
     FABA 1002              BPL      DFJMP0
     FABC 8524    DFLOOP    STA      WSYNC                   ;CHANGE COLOR ONCE PER T
     FABE 8524    DFJMP0    STA      WSYNC                   ;SECOND LINE
     FAC0 8524    DFJMP1    STA      WSYNC                   ;THIRD LINE
     FAC2 38                SEC                              ;GET A NEW COLOR FOR THE
     FAC3 E910              SBC      #$10
     FAC5 C910              CMP      #$10
     FAC7 B002              BCS      DFNEXT
     FAC9 E90F              SBC      #$0F
     FACB 8522    DFNEXT    STA      POC2                    ;CHANGE COLORS
     FACD CA                DEX
     FACE 10EC              BPL      DFLOOP

     FAD0 A240    DLIATARI  LDX      #$40                    ;SET UP CTRL FOR ATARI
     FAD2 863C              STX      CTRL

     FAD4 29F0              AND      #$F0
     FAD6 090E              ORA      #$0E
     FAD8 8527              STA      P1C3

     FADA A5EF              LDA      FUJICOLR
     FADC 29F0              AND      #$F0
     FADE 0906              ORA      #$06
     FAE0 8525              STA      P1C1
     FAE2 29F0              AND      #$F0
     FAE4 18                CLC                              ;ROTATE BAR COLOR
     FAE5 6940              ADC      #$40
     FAE7 9002              BCC      DLAJMP
     FAE9 690F              ADC      #$0F
     FAEB 0903    DLAJMP    ORA      #$03
     FAED 8526              STA      P1C2

     FAEF C6F1              DEC      KNLCOUNT                ;SEE IF TIME FOR A COLOR
     FAF1 1019              BPL      DLIDONE
```

```
FAF3 A5F3              LDA     KNLOFSET                    ;SEE IF TIME TO STAGGER, OR (
FAF5 6960              ADC     #$60
FAF7 9011              BCC     OLIOFSET
FAF9 A5EF              LDA     FUJICOLR                    ;ROTATE FUJI COLOR
FAFB 18               CLC
FAFC 6910              ADC     #$10
FAFE 9002              BCC     DLJMPO
FB00 690F              ADC     #$0F
FB02 85EF     DLJMPO   STA     FUJICOLR
FB04 A5F2              LDA     KNLTIME                     ;RESET TIMER
FB06 85F1              STA     KNLCOUNT

FB08 A900              LDA     #$00
FB0A 85F3     OLIOFSET STA     KNLOFSET                    ;UPDATE KERNAL STAGGERING CON
FB0C A902     OLIDONE  LDA     #$02                        ;NOTE THAT WE HAVE DONE KERNE
FB0E 85F0              STA     KNLSTATE
FB10 68               PLA                                  ;UNSTACK REGISTERS
FB11 AA               TAX
FB12 68      ENDOLI    PLA
FB13 40               RTI

FB14 4C14FB   INFLOOP  JMP     INFLOOP

            *   MAIN.S          MAIN ROUTINE FOR DECRYPTION CODE
            *   CALLED FROM ROUTINES IN TESTS.S

FB17 A2FF    STARTVND  LDX     #STACKPTR
FB19 9A               TXS                                  ;SET STACK POINTER

FB1A A900              LDA     #0                          ;ZERO THE TIA REGISTERS OUT
FB1C AA               TAX
FB1D 9501    TIAOLOOP  STA     1,X
FB1F E8               INX
FB20 E02C             CPX     #$2C
FB22 D0F9             BNE     TIAOLOOP
FB24 A902             LDA     #$02                        ;BACK INTO MARIA MODE
FB26 8501             STA     INPTCTRL

            *   THIS ROUTINE DROPS OUR CODE INTO RAM
FB28 A200    DROPRAM   LDX     #$00                        ;X = 0, DROP CODE AND GRAPHIC
FB2A 9620             STX     BACKGRND                    ;PUT BACKGROUND TO BLACK

FB2C 8D00F4   DRLOOP   LDA     ROMCODE+$000,X              ;DROP CODE
FB2F 9D0023            STA     RAMCODE+$000,X
FB32 8D00F5            LDA     ROMCODE+$100,X
FB35 9D0024            STA     RAMCODE+$100,X
FB38 8D00F6            LDA     ROMCODE+$200,X
FB3B 9D0025            STA     RAMCODE+$200,X
FB3E 8D00F7            LDA     ROMCODE+$300,X
FB41 9D0026            STA     RAMCODE+$300,X
FB44 8D00F8            LDA     ROMCODE+$400,X
FB47 9D0027            STA     RAMCODE+$400,X
FB4A BDBEFB            LDA     ROMDLIST,X                  ;DROP DISPLAY LISTS
```

```
sembling DECRYPT.S
------- 65xx Assembler V5.6 ------------------------------

840 9D0022              STA     RAMDLIST,X
850 E000               CPX     #$00
852 302A               BMI     DRLJMP0
854 BD4BFC              LDA     ROMDLL,X              ;DROP D
857 9D841F              STA     RAMDLL,X
85A BDC6FC              LDA     ROMGRPH6,X            ;DROP GR          TO HALF PAGE
85D 9D8419         ..   STA     RAMGRAPH+$000,X
860 BD3DFD      '       LDA     ROMGRPH5,X
863 9D841A              STA     RAMGRAPH+$100,X
866 BDB4FD        .     LDA     ROMGRPH4,X
869 9D841B              STA     RAMGRAPH+$200,X
86C BD18FE              LDA     ROMGRPH3,X    J
86F 9D841C              STA     RAMGRAPH+$300,X
872 BD57FE              LDA     ROMGRPH2,X
875 9D841D              STA     RAMGRAPH+$400,X
878 BD96FE              LDA     ROMGRPH1,X
87B 9D841E              STA     RAMGRAPH+$500,X
87E CA      DRLJMP0     DEX
87F D0AB                BNE     DRLOOP

881 4C0623              JMP     CARTTEST-CODEDIF      ;START THE DECRYPTION


                *  TURN THE GRAPHICS ON
884 ADF9FF : GRAPHON    LDA     $FFF9                 ;SEE IF DISPLAY IS TO BE START
887 2904                AND     #$04
889 F032                BEQ     STRTCRPT

88B A903     GRAPHON2   LDA     #$03                  ;SET UP KERNEL
88D 85F1                STA     KNLCOUNT
88F 85F2                STA     KNLTIME

891 A949                LDA     #$49                  ;SET COLOR
893 85EF                STA     FUJICOLR
895 A966                LDA     #$66
897 8525                STA     P1C1
899 A956                LDA     #$56
89B 8526                STA     P1C2
89D A92E                LDA     #$2E
89F 8527                STA     P1C3

8A1 A9AA                LDA     #L(DLI)               ;SET DLI
8A3 85F4                STA     DLIADDR
8A5 A9FA                LDA     #H(DLI)
8A7 85F5                STA     DLIADDR+1

8A9 2428     SCREENOF   BIT     MSTAT                 ;IS VBLANK ENDED YET?
8AB 30FC                BMI     SCREENOF
8AD 2428     SCREENON   BIT     MSTAT                 ;IS VBLANK STARTED YET?
8AF 10FC                BPL     SCREENON

8B1 A984                LDA     #L(RAMDLL)
8B3 8530                STA     DPPL                  ;SET DPPL AND DPPH TO DLLIST
8B5 A91F                LDA     #H(RAMDLL)
```

ssembling DECRYPT.S   PAGE   29
------- 65xx Assembler V5.6 -------------------------------------- Kontron -------


```
FBB7 852C                      STA     OPF4
FB89 A943                      LDA     #$43
FB8B 853C                      STA     CTRL                    ;TURN GRAPHICS ON

FBBD 60          STRTCRPT RTS


                 *  DISPLAY LISTS
FBBE 84          ROMDLIST DB     L(RAMGRAPH),$1F,H(RAMGRAPH),$BB,$00,$00      ;4 BY
FBBF 1F
FBC0 19
FBC1 BB
FBC2 00
FBC3 00
FBC4 84          RDL5BYTE DB     L(RAMGRAPH),$40,H(RAMGRAPH),$1F,$BB,$00,$00   ;5 BY
FBC5 40
FBC6 19
FBC7 1F
FBC8 BB
FBC9 00
FBCA 00
FBCB 85          RDLFUJI1 DB     L(RAMGRAPH+ROMFUJI1-ROMGRAPH),$1C,H(RAMGRAPH),$4A,$0
FBCC 1C
FBCD 19
FBCE 4A
FBCF 00
FBD0 00
FBD1 89          RDLFUJI2 DB     L(RAMGRAPH+ROMFUJI2-ROMGRAPH),$1C,H(RAMGRAPH),$4A,$0
FBD2 1C
FBD3 19
FBD4 4A
FBD5 00
FBD6 00
FBD7 8D          RDLFUJI3 DB     L(RAMGRAPH+ROMFUJI3-ROMGRAPH),$1C,H(RAMGRAPH),$48,$0
FBD8 1C
FBD9 19
FBDA 48
FBDB 00
FBDC 00
FBDD 91          RDLFUJI4 DB     L(RAMGRAPH+ROMFUJI4-ROMGRAPH),$18,H(RAMGRAPH),$46,$0
FBDE 18
FBDF 19
FBE0 46
FBE1 00
FBE2 00
FBE3 96          RDLFUJI5 DB     L(RAMGRAPH+ROMFUJI5-ROMGRAPH),$19,H(RAMGRAPH),$42,$0
FBE4 19
FBE5 19
FBE6 42
FBE7 00
FBE8 00
FBE9 9D          RDLFUJI6 DB     L(RAMGRAPH+ROMFUJI6-ROMGRAPH),$17,H(RAMGRAPH),$3E,$0
FBEA 17
```

Assembling DECRYPT.S
------- 65xx Assembler V5.6 ---------------------------------

```
FBEB 19
FBEC 3E
FBED 00
FBEE 00
FBEF A6        RDLFUJI7   DB      L(RAMGRAPH+ROMFUJI7-ROMGRAPH)                    (H),$3E,
FBF0 17
FBF1 19
FBF2 3E
FBF3 00
FBF4 00
FBF5 AF        RDLRACE    DB      L(RAMGRAPH+ROMSTRIP-ROMGRPH3)           RAPH+$300)
FBF6 2C
FBF7 1C
FBF8 00
FBF9 AF                   DB      L(RAMGRAPH+ROMSTRIP-ROMGRPH3)           GRAPH+$300)
FBFA 2C
FBFB 1C
FBFC 50
F3FD 00
FBFE 00
FBFF AF        RDLRACEL   DB    - L(RAMGRAPH+ROMSTRIP-ROMGRPH3),$2  (RAMGRAPH+$400)
FC00 2C
FC01 1D
FC02 00
FC03 AF                   DB      L(RAMGRAPH+ROMSTRIP-ROMGRPH3),$2   RAMGRAPH+$400)
FC04 2C
FC05 1D
FC06 50
FC07 00
FC08 00
FC09 AF        RDLINE01   DB      L(RAMGRAPH+ROMLINE1-             (RAMGRAPH+$000)
FC0A 2D
FC0B 19
FC0C 28
FC0D 00
FC0E 00
FC0F C2        RDLINE02   DB      L(RAMGRAPH+ROMLINE2-ROMGRAPH),$2   (RAMGRAPH+$000)
FC10 2D
FC11 19
FC12 28
FC13 00
FC14 00
FC15 D5        RDLINE03   DB      L(RAMGRAPH+ROMLINE3-ROMGRAPH),$20,H(RAMGRAPH+$000)
FC16 2D
FC17 19
FC18 28
FC19 00
FC1A 00
FC1B E8        RDLINE04   DB      L(RAMGRAPH+ROMLINE4-ROMGRAPH),$2D,H(RAMGRAPH+$000)
FC1C 2D
FC1D 19
FC1E 28
FC1F 00
FC20 00
```

ssembling DECRYPT.S   PAGE   31
-------- 65xx Assembler V5.6 ----------------------------------Kontron --------

```
FC21 AF        RDLINE05   DB       L(RAMGRAPH+ROMLINE1-ROMGRAPH),$20,H(RAMGRAPH+$100),1
FC22 2D
FC23 1A
FC24 28
FC25 00
FC26 00
FC27 C2        RDLINE06   DB       L(RAMGRAPH+ROMLINE2-ROMGRAPH),$20,H(RAMGRAPH+$100),1
FC28 2D
FC29 1A
FC2A 28
FC2B 00
FC2C 00
FC2D D5        RDLINE07   DB       L(RAMGRAPH+ROMLINE3-ROMGRAPH),$20,H(RAMGRAPH+$100),1
FC2E 2D
FC2F 1A
FC30 28
FC31 00
FC32 00
FC33 E8        RDLINE08   DB       L(RAMGRAPH+ROMLINE4-ROMGRAPH),$20,H(RAMGRAPH+$100),1
FC34 2D
FC35 1A
FC36 28
FC37 00
FC38 00
FC39 AF        RDLINE09   DB       L(RAMGRAPH+ROMLINE1-ROMGRAPH),$20,H(RAMGRAPH+$200),
FC3A 2D
FC3B 1B
FC3C 28
FC3D 00
FC3E 00
FC3F C2        RDLINE10   DB       L(RAMGRAPH+ROMLINE2-ROMGRAPH),$20,H(RAMGRAPH+$200),
FC40 2D
FC41 1B
FC42 28
FC43 00
FC44 00
FC45 D5        RDLINE11   DB       L(RAMGRAPH+ROMLINE3-ROMGRAPH),$20,H(RAMGRAPH+$200),1
FC46 2D
FC47 1B
FC48 28
FC49 00
FC4A 00
```

```
               * DISPLAY LIST LIST
FC4B 0F        ROMDLL     DB       $0F,H(RAMDLIST),L(RAMDLIST+RDL5BYTE-ROMDLIST)   ;5 BY
FC4C 22
FC4D 06
FC4E 0F                   DB       $0F,H(RAMDLIST),L(RAMDLIST+$00)
FC4F 22
FC50 00
FC51 0F                   DB       $0F,H(RAMDLIST),L(RAMDLIST+$00)
FC52 22
FC53 00
```

Assembling DECRYPT.S
------- 65xx Assembler V5.6 --   PAGE   32

```
FC54  0F                    DB      $0F,H(RAMDLIST),L(RAMDLIST+
FC55  22
FC56  00
FC57  03                    DB      $03,H(RAMDLIST),L(RAMDLIST+
FC58  22
FC59  00
FC5A  85                    DB      $85,H(RAMDLIST),L(RAMDLIST+RD
FC5B  22
FC5C  0D
FC5D  05                    DB      $05,H(RAMDLIST),L(RAMDLIST+RD
FC5E  22
FC5F  13
FC60  05                    DB      $05,H(RAMDLIST),L(RAMDLIST+RD
FC61  22
FC62  19
FC63  05                    DB      $05,H(RAMDLIST),L(RAMDLIST+RD
FC64  22
FC65  1F
FC66  05                    DB      $05,H(RAMDLIST),L(RAMDLIST+RD
FC67  22
FC68  25
FC69  05                    DB      $05,H(RAMDLIST),L(RAMDLIST+RD
FC6A  22
FC6B  2B
FC6C  05                    DB      $05,H(RAMDLIST),L(RAMDLIST+RD
FC6D  22
FC6E  31
FC6F  0F                    DB      $0F,H(RAMDLIST),L(RAMDLIST+$00
FC70  22
FC71  00
FC72  01                    DB      $01,H(RAMDLIST),L(RAMDLIST+RD
FC73  22
FC74  37
FC75  00                    DB      $00,H(RAMDLIST),L(RAMDLIST+RDLINE
FC76  22
FC77  48
FC78  02                    DB      $02,H(RAMDLIST),L(RAMDLIST+RD
FC79  22
FC7A  37
FC7B  00                    DB      $00,H(RAMDLIST),L(RAMDLIST+RDLINE02-ROMDLIST)
FC7C  22
FC7D  51
FC7E  02                    DB      $02,H(RAMDLIST),L(RAMDLIST+RDLRACE-ROMDLIST)
FC7F  22
FC80  37
FC81  00                    DB      $00,H(RAMDLIST),L(RAMDLIST+RDLINE03-ROMDLIST)
FC82  22
FC83  57
FC84  02                    DB      $02,H(RAMDLIST),L(RAMDLIST+RDLRACE-ROMDLIST)
FC85  22
FC86  37
FC87  00                    DB      $00,H(RAMDLIST),L(RAMDLIST+RDLINE04-ROMDLIST)
FC88  22
FC89  5D
```

ssembling DECRYPT.S   PAGE   33
------- 65xx Assembler V5.5 ----------------------------------- Kontron --------

```
C8A 02              DB      $02,H(RAMDLIS   ,L(RAMDLIST+RDLRACE-ROMDLIST)
C8B 22
C8C 37
C8D 00              DB      $00,H(RAMDLIST),L(RAMDLIST+RDLINE05-ROMDLIST)
C8E 22
C8F 63
C90 02              DB      $02,H(RAMDLIST),L(RAMDLIST+RDLRACE-ROMDLIST)
C91 22
C92 37
C93 00              DB      $00,H(RAMDLIST),L(RAMDLIST+RDLINE06-ROMDLIST)
C94 22
C95 69
C96 02              DB      $02,H(RAMDLIST),L(RAMDLIST+RDLRACE-ROMDLIST)
C97 22
C98 37
C99 00              DB      $00,H(RAMDLIST),L(RAMDLIST+RDLINE07-ROMDLIST)
C9A 22
C9B 6F
C9C 02              DB      $02,H(RAMDLIST),L(RAMDLIST+RDLRACE-ROMDLIST)
C9D 22
C9E 37
C9F 00              DB      $00,H(RAMDLIST),L(RAMDLIST+RDLINE08-ROMDLIST)
CA0 22
CA1 75
CA2 02              DB      $02,H(RAMDLIST),L(RAMDLIST+RDLRACE-ROMDLIST)
CA3 22
CA4 37
CA5 00              DB      $00,H(RAMDLIST),L(RAMDLIST+RDLINE09-ROMDLIST)
CA6 22
CA7 7B
CA8 02              DB      $02,H(RAMDLIST),L(RAMDLIST+RDLRACE-ROMDLIST)
CA9 22
CAA 37
CAB 00              DB      $00,H(RAMDLIST),L(RAMDLIST+RDLINE10-ROMDLIST)
CAC 22
CAD 81
CAE 02              DB      $02,H(RAMDLIST),L(RAMDLIST+RDLRACE-ROMDLIST)
CAF 22
CB0 37
CB1 00              DB      $00,H(RAMDLIST),L(RAMDLIST+RDLINE11-ROMDLIST)
CB2 22
CB3 87
CB4 01              DB      $01,H(RAMDLIST),L(RAMDLIST+RDLRACEL-ROMDLIST)
CB5 22
CB6 41
CB7 0F              DB      $0F,H(RAMDLIST),L(RAMDLIST+$00)        ;TRAILING SPA
CB8 22
CB9 00
CBA 0F              DB      $0F,H(RAMDLIST),L(RAMDLIST+$00)
CBB 22
CBC 00
CBD 0F              DB      $0F,H(RAMDLIST),L(RAMDLIST+$00)
CBE 22
CBF 00
```

```
  FCC0 0F                   DB       $0F,H(RAMDLIST),L(RAMDLIST+$00)
  FCC1 22
  FCC2 00
  FCC3 0F                   DB       $0F,H(RAMDLIST),L(RAMDLIST+$00)
  FCC4 22
  FCC5 00


                    *  ROM GRAPHICS FOR THE FUJI-A AND WORDS
  FCC6             ROMGRAPH
  FCC6 00          ROMGRPH6  DB      $00                        ;NULL INFO
  FCC7 7C          ROMFUJI1  DB      $7C,$7F,$8F,$80            ;LINE 6
  FCC8 7F
  FCC9 8F
  FCCA 80
  FCCB FC          ROMFUJI2  DB      $FC,$7F,$8F,$C0
  FCCC 7F
  FCCD 8F
  FCCE C0
  FCCF 1F          ROMFUJI3  DB      $1F,$87,$F8,$7F
  FCD0 87
  FCD1 F8
  FCD2 7E
  FCD3 0F          ROMFUJI4  DB      $0F,$E0,$7F,$81,$FC
  FCD4 E0
  FCD5 7F
  FCD6 81
  FCD7 FC
  FCD8 07          ROMFUJI5  DB      $07,$FF,$80,$7F,$80,$7F,$F8
  FCD9 FF
  FCDA 80
  FCDB 7F
  FCDC 80
  FCDD 7F
  FCDE F8
  FCDF 1F          ROMFUJI6  DB      $1F,$FF,$F0,$00,$7F,$80,$03,$FF,$FE
  FCE0 FF
  FCE1 F0
  FCE2 00
  FCE3 7F
  FCE4 80
  FCE5 03
  FCE6 FF
  FCE7 FE
  FCE8 1F          ROMFUJI7  DB      $1F,$00,$00,$00,$7F,$80,$00,$00,$3E
  FCE9 00
  FCEA 00
  FCEB 00
  FCEC 7F
  FCED 80
  FCEE 00
  FCEF 00
  FCF0 3E
```

```
OCE9
FCF1  00      ROMLINE1  DI      $00,$00,$00,$00,$3F,$FF,$FF,$FF      ;LINE 1
FCF2  00
FCF3  00
FCF4  00
FCF5  3F
FCF6  FF
FCF7  FF
FCF8  FF
FCF9  00                           F   .  ,$00,$00,$00,$3F,$FF,$FF
FCFA  00
FCFB  00
FCFC  00
F  FD  00
F  FE  3F
F  FF  FF
F  00  FF
FD01  00      DB      $00,$03,$FC
FD02  03
FD03  FC
FD04  00      ROMLINE2  DI      $00,$00,$3F,$00,$3F,$FF,$FF,$FF     ;LINE 2
FD05  00
FD06  3F
FD07  00
FD0   FF
F  A  FF
FD0B  FF
FD0C  00                           $F0,$02,$F0,$00,$00,$3F,$FF,$FF
FD0D  03
F
FD10  00
FD11  3F
FD12  FF
FD13  FF
FD14  FC      DB      $FC,$03,$FC
FD15  03
FD16  FC
FD17  00      ROMLINE3  DI      $00,$00,$FF,$00,$00,$03,$FF,$00     ;LINE 3
```

CRYPT.L   April 23, 1984   1:29:06 pm   page 36

                                                                      PAGE   36
Assembling DECRYPT.S --------------------------------------- Kontron --------
-------- 65xx Assembler V5.6 ---

```
D27 FF              DB          $FF,$C3,$FC

D28 C3
D29 FC
D2A 00   ROMLINE4   DB          $00,$03,$FF,$F0,$00,$03,$FF,$00              ;LINE 4 OF AT
D2B 03
D2C FF
D2D F0
C2E 00
D2F 03
D30 FF
D31 00              DB          $00,$3F,$FF,$00,$00,$3F,$F0,$00
D32 00
D33 3F
D34 FF
D35 00
D36 00
D37 3F
D38 F0
D39 00
D3A 3F              DB          $3F,$C3,$FC
D3B C3
D3C FC

FD3D 00  ROMGRPH5.  DB          $00                           ;NULL INFO
FD3E 7C             DB          $7C,$7F,$8F,$80               ;LINE 5
FD3F 7F
FD40 8F
FD41 80
FD42 7C             DB          $7C,$7F,$8F,$80
FD43 7F
FD44 8F
FD45 80
FD46 1F             DB          $1F,$87,$F8,$7E
FD47 87
FD48 F8
FD49 7E
FD4A 0F             DB          $0F,$F0,$7F,$83,$FC
FD4B F0
FD4C 7F
FD4D 83
FD4E FC
FD4F 01             DB          $01,$FF,$80,$7F,$80,$7F,$80
FD50 FF
FD51 80
FD52 7F
FD53 80
FD54 7F
FD55 80
FD56 1F             DB          $1F,$FF,$F8,$00,$7F,$80,$07,$FF,$FE
FD57 FF
FD58 F8
FD59 00
FD5A 7F
FD5B 80
```

CONFIDENTIAL

```
FD5C  07
FD5D  FF
FD5E  FE
FD5F  1F            DB      $1F,$F0,$00,$00,$7F,$80,$00,$03,$FF
FD60  F0
FD61  00
FD62  00
FD63  7F
FD64  80
FD65  00
FD66  03
FD67  FE

FD68  00            DB      $00,$0F,$F3,$FC,$00,$03,$FF,$00    ;LINE 5 OF
FD69  0F
FD6A  F3
FD6B  FC
FD6C  00
FD6D  03
FD6E  FF
FD6F  00
FD70  00            DB      $00,$FF,$3F,$C0,$00,$3F,$F0,$00
FD71  FF
FD72  3F
FD73  C0
FD74  00
FD75  3F
FD76  F0
FD77  00
FD78  FF            DB      $FF,$C3,$FC
FD79  C3
FD7A  FC
FD7B  00            DB      $00,$3F,$C0,$FF,$00,$03,$FF,$00    ;LINE 6 OF
FD7C  3F
FD7D  C0
FD7E  FF
FD7F  00
FD80  03
FD81  FF
FD82  00
FD83  03            DB      $03,$FC,$0F,$F0,$00,$3F,$F0,$3F
FD84  FC
FD85  0F
FD86  F0
FD87  C0
FD88  3F
FD89  F0
FD8A  3F
FD8B  FC            DB      $FC,$03,$FC
FD8C  03
FD8D  FC
FD8E  00            DB      $00,$FF,$00,$3F,$C0,$03,$FF,$00    ;LINE 7 OF
FD8F  FF
FD90  00
```

```
FD91  3F
FD92  CO
FD93  03
FD94  FF
FD95  00
FD96  OF          DB      $0F,$F0,$03,$FC,$00,$3F,$FF
FD97  FO
FD98  03
FD99  FC
FD9A  00
FD9B  3F
FD9C  FO
FD9D  FF
FD9E  CO          DB      $C0,$03,$FC
FD9F  03
FDAO  FC
FDA1  03          DB      $03,$FF,$FF,$FF,$F0,$03,$FF
FDA2  FF
FDA3  FF
FDA4  FF
FDA5  FO
FDA6  03
FDA7  FF
FDA8  00
FDA9  3F          DB      $3F,$FF,$FF,$FF,$00,$3F,$F0
FDAA  FF
FDAB  FF
FDAC  FF
FDAD  00
FDAE  3F
FDAF  FO
FDB0  3F
FDB1  FO          DB      $F0,$03,$FC
FDB2  03
FDB3  FC

FDB4  00    ROMGRPH4  DB   $00                    ;NULL INE
FDB5  7C          DB      $7C,$7F,$8F,$80         ;LINE 6
FDB6  7F
FDB7  8F
FDB8  80
FDB9  7C          DB      $7C,$7F,$8F,$80
FDBA  7F
FDBB  8F
FDBC  80
FDBD  1F          DB      $1F,$87,$F8,$7E
FDBE  87
FDBF  F8
FDC0  7E
FDC1  07          DB      $07,$F0,$7F,$03,$F8
FDC2  FO
FDC3  7F
FDC4  03
FDC5  F8
```

Assembling DECRYPT.S
-------- 65xx Assembler V5.6 --------------------------------------- Kontron --------

```
FDC6  00          DB          $00,$FF,$C0,$7F,$80,$FF,$C0
FDC7  FF
FDC8  C0
FDC9  7F
FDCA  80
FDCB  FF
FDCC  C0
FDCD  1F          DB          $1F,$FF,$FC,$00,$7F,$80,$0F,$FF,$FE
FDCE  FF
FDCF  FC
FDD0  00
FDD1  7F
FDD2  80
FDD3  0F
FDD4  FF
FDD5  FE
FDD6  1F          DB          $1F,$FC,$00,$00,$7F,$80,$00,$0F,$FE
FDD7  FC
FDD8  00
FDD9  00
FDDA  7F
FDDB  80
FDDC  00
FDDD  0F
FDDE  FE

FDDF  0F          DB          $0F,$FF,$FF,$FF,$FC,$03,$FF,$00          ;LINE 9 OF
FDE0  FF
FDE1  FF
FDE2  FF
FDE3  FC
FDE4  03
FDE5  FF
FDE6  00
FDE7  FF          DB          $FF,$FF,$FF,$FF,$C0,$3F,$F0,$0F
FDE8  FF
FDE9  FF
FDEA  FF
FDEB  C0
FDEC  3F
FDED  F0
FDEE  0F
FDEF  FC          DB          $FC,$03,$FC
FDF0  03
FDF1  FC
FDF2  3F          DB          $3F,$F0,$00,$03,$FF,$03,$FF,$03          ;LINE 10 OF
FDF3  F0
FDF4  00
FDF5  03
FDF6  FF
FDF7  03
FDF8  FF
FDF9  03
FDFA  FF          DB          $FF,$00,$00,$3F,$F0,$3F,$F0,$03
```

```
FDFB 00
FDFC 00
FDFD 3F
FDFE F0
FDFF 3F
FE00 F0
FE01 03
FE02 FF        DB        $FF,$03,$FC
FE03 03
FE04 FC
FE05 FF        DB        $FF,$C0,$00,$00,$FF,$C3,$FF,$C0
FE06 C0
FE07 00
FE08 00
FE09 FF
FE0A C3
FE0B FF
FE0C 0F
FE0D FC        DB        $FC,$00,$00,$0F,$FC,$3F,$F0,$C0
FE0E 00
FE0F 00
FE10 0F
FE11 FC
FE12 3F
FE13 F0
FE14 00
FE15 FF        DB        $FF,$C3,$FC
FE16 C3
FE17 FC
FE18 00        ROMGRPH3  DB   $00              ;NULL LINE
FE19 7C                  DB   $7C,$7F,$8F,$80  ;LINE 3
FE1A 7F
FE1B 8F
FE1C 80
FE1D 7C        DB        $7C,$7F,$8F,$80
FE1E 7F
FE1F 8F
FE20 80
FE21 0F        DB        $0F,$87,$F8,$7C
FE22 87
FE23 F8
FE24 7C
FE25 07        DB        $07,$F0,$7F,$83,$F8
FE26 F0
FE27 7F
FE28 83
FE29 F8
FE2A 00        DB        $00,$7F,$C0,$7F,$80,$FF,$80
FE2B 7F
FE2C C0
FE2D 7F
FE2E 80
FE2F FF
```

Assembling DECRYPT.S   PAGE   41
-------- 65xx Assembler V5.6 --------------------------------------- Kontron --------

```
FE30  80
FE31  1F              DB      $1F,$FF,$FE,$00,$7F,$60,$1F,$FF,$FE
FE32  FF
FE33  FE
FE34  00
FE35  7F
FE36  60
FE37  1F
FE38  FF
FE39  FE
FE3A  1F              DB      $1F,$FF,$00,$00,$7F,$80,$00,$3F,$FE
FE3B  FF
FE3C  00
FE3D  00
FE3E  7F
FE3F  80
FE40  00
FE41  3F
FE42  FE

FE43  55    ROMSTRIP  DB      $55,$55,$55,$55,$55,$55,$55,$55        ;RACING STR
FE44  55
FE45  55
FE46  55
FE47  55
FE48  55
FE49  55
FE4A  55
FE4B  55              DB      $55,$55,$55,$55,$55,$55,$55,$55
FE4C  55
FE4D  55
FE4E  55
FE4F  55
FE50  55
FE51  55
FE52  55
FE53  55              DB      $55,$55,$55,$55
FE54  55
FE55  55
FE56  55

FE57  00    ROMGRPH2  DB      $00                                   ;NULL INFO
FE58  7C         .    DB      $7C,$7F,$8F,$80                       ;LINE 2
FE59  7F
FE5A  8F
FE5B  80
FE5C  7C              DB      $7C,$7F,$8F,$80
FE5D  7F
FE5E  8F
FE5F  80
FE60  0F              DB      $0F,$C7,$F8,$FC
FE61  C7
FE62  F8
FE63  FC
```

Assembling DECRYPT.S
-------- 65xx Assembler V5.6 --------------------------------

```
FE64 03                    DB      $03,$F0,$7F,$83,$F0
FE65 F0
FE66 7F
FE67 83
FE68 F0
FE69 00                    DB      $00,$3F,$E0,$7F,$81,$FF,$00
FE6A 3F
FE6B E0
FE6C 7F
FE6D 81
FE6E FF
FE6F 00
FE70 01                    DB      $01,$FF,$FE,$00,$7F,$80,$1F,$FF
FE71 FF
FE72 FE
FE73 00
FE74 7F
FE75 80
FE76 1F
FE77 FF
FE78 E0
FE79 1F                    DB      $1F,$FF,$C0,$00,$7F,$80,$00,$FF,$FF
FE7A FF
FE7B C0
FE7C 00
FE7D 7F
FE7E 80
FE7F 00
FE80 FF
FE81 FE
```

```
FE82 AA                    DB      $AA,$AA,$AA,$AA,$AA,$AA,$AA,$AA      ;RACING STRI
FE83 AA
FE84 AA
FE85 AA
FE86 AA
FE87 AA
FE88 AA
FE89 AA
FE8A AA                    DB      $AA,$AA,$AA,$AA,$AA,$AA,$AA,$AA
FE8B AA
FE8C AA
FE8D AA
FE8E AA
FE8F AA
FE90 AA
FE91 AA
FE92 AA                    DB      $AA,$AA,$AA,$AA
FE93 AA
FE94 AA
FE95 AA

FE96 00          ROMGRPH1  DB      $00                  ;NULL INFO
FE97 7C                    DB      $7C,$7F,$8F,$80      ;LINE 1
```

```
FE98 7F
FE99 8F
FE9A 80
FE9B 7C           DB        $7C,$7F,$8F,$80
FE9C 7F
FE9D 8F
FE9E 80
FE9F 0F           DB        $0F,$C7,$F8,$FC
FEA0 C7
FEA1 F8
FEA2 FC
FEA3 03           DB        $03,$F8,$7F,$87,$F0
FEA4 F8
FEA5 7F
FEA6 87
FEA7 F0
FEA8 00           DB        $00,$1F,$E0,$7F,$81,$FE,$00
FEA9 1F
FEAA E0
FEAB 7F
FEAC 81
FEAD FE
FEAE 00
FEAF 00           DB        $00,$1F,$FF,$00,$7F,$80,$3F,$FE,$00
FEB0 1F
FEB1 FF
FEB2 00
FEB3 7F
FEB4 80
FEB5 3F
FEB6 FE
FEB7 00
FEB8 1F           DB        $1F,$FF,$E0,$00,$7F,$80,$01,$FF,$FE
FEB9 FF
FEBA E0
FEBB 00
FEBC 7F
FEBD 80
FEBE 01
FEBF FF
FEC0 FE
```

```
FEC1 55           DB        $55,$55,$55,$55,$55,$55,$55,$55       ;RACING STR
FEC2 55
FEC3 55
FEC4 55
FEC5 55
FEC6 55
FEC7 55
FEC8 55
FEC9 55           DB        $55,$55,$55,$55,$55,$55,$55,$55
FECA 55
FECB 55
FECC 55
```

Assembling DECRYPT.S
--------- 65xx Assembler V5.6 ---------

```
FECD 55
FECE 55
FECF 55
FED0 55
FED1 55              DB        $55,$55,$55,$55
FED2 55
FED3 55
FED4 55


         *  NUMBERS.S      NUMBERS FOR THE ENCRYPTION


         *  THIS IS A MASK APPLIED TO THE HI BYTE OF              SURE
         *  LESS THAN N
0007     NMASK      EQU       $07

         *  N = P*Q, THE BASIC MODULO OF DECRYPTION
0077     NLEN       EQU       $77
FED5 09  N          DB        $09,$CA,$C9,$C6,$54,$12,$08
FED6 CA
FED7 C9
FED8 C6
FED9 54
FEDA 12
FEDB 08
FEDC 1B
FEDD 60             DB        $60,$58,$81,$48,$86,$01,$08
FEDE 58
FEDF 81
FEE0 48
FEE1 86
FEE2 01
FEE3 08
FEE4 3F
FEE5 09             DB        $D9,$25,$A0,$78,$DC,$32,$79
FEE6 25
FEE7 A0
FEE8 73
FEE9 DC
FEEA 32
FEEB 79
FEEC 84
FEED 38             DB        $38,$7C,$8C,$2F,$E2,$E2,$FA,$8D
FEEE 7C
FEEF 8C
FEF0 2F
FEF1 E2
FEF2 E2
FEF3 FA
FEF4 8D
FEF5 0A             DB        $0A,$00,$38,$C5,$EC,$AF,$2D,$8A
FEF6 00
FEF7 38
```

```
FEF8 C5
FEF9 EC
FEFA AF
FEFB 2D
FEFC 3A
FEFD CD        DB        $CD,$05,$93,$6A,$A5,$14,$45,$77
FEFE 06
FEFF 93
FF00 6A
FF01 A5
FF02 14
FF03 46
FF04 77
FF05 C4        DB        $C4,$6A,$82,$53,$36,$EF,$8C,$CE
FF06 6A
FF07 82
FF08 53
FF09 36
FF0A EF
FF0B 8C
FF0C CE
FF0D 0C        DB        $0C,$A2,$68,$71,$03,$73,$E8,$F7
FF0E A2
FF0F 6R
FF10 71
FF11 03
FF12 73
FF13 E8
FF14 F7
FF15 6D        DB        $6D,$05,$B5,$20,$EF,$23,$47,$0C
FF16 06
FF17 B5
FF18 20
FF19 EF
FF1A 23
FF1B 47
FF1C 0C
FF1D 51        DB        $51,$55,$C8,$FE,$F4,$58,$C4,$3F
FF1E 55
FF1F C8
FF20 FE
FF21 F4
FF22 58
FF23 C4
FF24 3F
FF25 20        DB        $20,$A7,$67,$38,$B0,$76,$E2,$C4
FF26 A7
FF27 67
FF28 38
FF29 B0
FF2A 76
FF2B E2
FF2C C4
FF2D D8        DB        $D8,$05,$63,$F8,$3C,$58,$38,$2D
```

Assembling DECRYPT.S
------- 65xx Assembler V5.6 --

```
FF2E  05
FF2F  63
FF30  F8
FF31  3C
FF32  58
FF33  3B
FF34  2D
FF35  22          ..    DB        $22,$CC,$88,$B3,$71,$8F,$1D,$80
FF36  CC
FF37  88
FF38  B3
FF39  71
FF3A  8F
FF3B  1D
FF3C  80
FF3D  0A          DB        $0A,$87,$BD,$A1,$59,$23,$E9,$70
FF3E  87
FF3F  BD
FF40  A1
FF41  59
FF42  23
FF43  E9
FF44  70
FF45  E2          DB        $E2,$03,$EC,$46,$68,$80,$42,$39
FF46  03
FF47  EC
FF48  46
FF49  68
FF4A  80
FF4B  42
FF4C  39
```

```
              *   END.S           END OF CODE

FF4D  EA      ENDROM2   NOP

FFEE                    ORG       $FFEE
FFEE  47                DB        'GCC(C)1984'
FFEF  43
FFF0  43
FFF1  28
FFF2  43
FFF3  29
FFF4  31
FFF5  39
FFF6  38
FFF7  34

FFF8                    ORG       $FFF8
FFF8  F0                DB        $F0         ;CHECKSUM, MAKES EOR CHECK
FFF9  F7                DB        $F7         ;CART STARTS AT $F000 - 7
FFFA  00F0              DW        INTOLI      ;INTERNAL GAME DLI HANDLER
FFFC  84F8              DW        MAIN
```

Assembling DECRYPT.S   PAGE   47
-------- 65xx Assembler V5.6 --- Kontron --------

FFFE 33F9   DW   IRQINT

0000   END

Assemb... YPT.S
          Assembler V5.6 -----------------------------------

Sym	Val	Sym	Val	Sym	Val	Sym	Val
ACC		ADACMOD0	F764	ADACMOD1	F76A		F772
ADACMOD3		ADDLOOP	F764	...LOOP2	F770		F767
ADDOUT	F7..	AUDC0	u001.	AUDC1	u0016		0017
AUDF1	u0018	AUDV0	u001?	AUDV1	u001A		0020
BAD6116A	0001	BAD6116B	000?	...CAST	F403		0000
BADMARIA	0004	BADRAM	0003	...ALT?	u0005		00EE
CARTTEST	F406	CHARBASE	u0034				F79F
CMPLOOP	F79F	CMPMOD	F7A?		F7A..		7A7
CCCLDIF	D100	CPUERR	F933	CPU...	F9F.		15B
CPUTEST	F938	CSOLOOP	F464	CSALOOP	F4..		46.
CSCDLOOP	F518	CSCDOUT	F523	CSC.L..	F4C2		
CSCHKOLI	F512	CSCLOOP	F4F8	CSCM...	F4F.		
CSCSLOOP	F482	CSCTLOOP	F4AA	CSPLOOP	F5.5		
CTA	F943	CTB	F948	CTC	F959		
CTD	F95E	CTE	F969	CTF	F97.		
CTJSRRET	FA8E	CTLSWA	u0281	CTLSW.	u02..	CT..	
CTSHIFT	FA5B	CTSTLOOP	F40E	...LOOP	F...	DC..	
DC...0	F66B	DECRYPT	F620	DEX	F75.	DE..	
D..JMP.	FAC0	DFLOOP	FABC	DF.EXT	FAC.	DIV.	
......	F60D	DIVLOOP	F714	DIV..OP.	...	DIV..	
...UT	72.	DIVSUB	F77F	DL.J..	...	DLI.	
DLIAT...	00E4	DLIATARI	uFAD0	DLI.ONE	F.CC	DLI0F	
DLJ...	FB02	DONE	F814	OPPH	002C	OPPL.	
..L..O	F.7E	ORLOOP	FB2C	PROPRAM	u.02.	DUMMY	
......	F7E.	ENDDLI	FB12	ENDROM	u.81.	.ENDROM	
FATAL..	F6..	FUJICOLR	00EF	GRAPHON	F8.4	GRAPH..	
HSE.OFF	F759	INFLOOP	FB14	INPT4	u000.	INPT5	
INPTCTRL	0001	INTDLI	F000	INTIM	u02.4	IRQINT	
KNLCOUNT	00F1	KNLOFSET	00F3	KNLSTATE	00F.	KNLTIME	
LDL.UP	F73F	LOCK2600	F789	MAIN	F9.4	MARIAERR	F8..
MDMRLOOP	F64F	MSTAT	0028	MULCALP	F6.5	MULLOOP	
MULLOOP0	F6..	MULNEXT	F6C0	MULOUT	F.C.	MULTADD	F.C..
MULTIPLY	F6.2	MULTMASK	F5D0	N.	F..	N..N	0027
VMASK	00.7	...CART	F400	NOCHECK	F4F.	.FFSET	u.03.
OFFSETA	00E1	OFFSETR	00E2	OUT	F.C6	P0C1	u0021
P0C2	00.2	P0C3	u0023	P1C1	002.	P1C2	u0026
P1C3	0027	P2C1	u0029	P2C2	u002.	P2C3	u0028
P3C1	u002C	P3C2	u002E	P3C3	u002F	P4C1	u0031
P4C2	u0032	P4C3	u0033	P5C1	u0035	P5C2	u0036
P5C3	u0037	P6C1	u0039	P6C2	u003A	P6C3	u0038
P7C1	u003D	P7C2	u003E	P7C3	u003F	RAMAERR	F8DB
RAMAPAGE	F923	RAMBERR	F8DD	RAMBPAGE	F92B	RAMCHECK	uF89A
RAMCODE	2300	RAMDLIST	2200	RAMDLL	1F84	RAMERR	F8E2
RAMGRAPH	1984	RAMPAT	F91D	RAMTEST	F8E7	RANDBYTE	0004
RCAGAIN	F89C	RCLOOP	F8A1	RDL5BYTE	FBC4	RDLFUJI1	FBCB
RDLFUJI2	F8D1	RDLFUJI3	F8D7	RDLFUJI4	F8DD	RDLFUJI5	FBE3
RDLFUJI6	F..	RDLFUJI7	F.EF	RDLINE01	FC09	RDLINE02	FC0F
RDLINE03	FC15	RDLINE04	FC13	RDLINE05	FC21	RDLINE06	FC27
RDLINE07	FC2D	RDLINE08	FC33	RDLINE09	FC39	RDLINE10	FC3F
RDLINE11	FC45	RDLRACE	F8F5	RDLRACEL	F8FF	REG0	1900
REG1	2000	REG10	1F00	REG12	1F00	REG14	2100
REG.	1A00	REG4	1F00	REG6	1C00	REG8	1000
REG1.	00FE	ROMCODE	F400	ROMCODE2	F880	ROMDLIST	F8BE

Assembling DECRYPT.S
------- 65xx Assembler V5.6 ---

ROMDLL	FC4B	ROMFUJI1	FCC7	ROMFUJI2	FCCB	ROMFUJI3	FCCF
ROMFUJI4	FCD3	ROMFUJI5	FCDB	ROMFUJI6	FCDF	ROMFUJI7	FCE3
ROMGRAPH	FCC6	ROMGRPH1	FE96	ROMGRPH2	FE57	ROMGRPH3	FE18
ROMGRPH4	FDB4	ROMGRPH5	FD3D	ROMGRPH6	FCC6	ROMLINE1	FCF1
ROMLINE2	FD04	ROMLINE3	FD17	ROMLINE4	FD2A	ROMSTRIP	FE43
RTLOOP	F902	RTPAGE	F8F1	RTPAT	F8FD	S	F525
SBACMOD0	F782	SBACMOD1	F788	SBACMOD2	F790	SBACMOD3	F795
SCREENOF	FBA9	SCREENON	FBAD	SCRLOOP	F73A	SEROUT	uF758
SERSLOOP	F74A	SERSMOD1	F74A	SERSMOD2	F74E	SETMARTA	F780
SETREGS	F730	SIZEA	00E3	SIZERO	00E4	SIZER1	00E5
SIZER3	u00E6	SIZER5	u00E7	STACKPTR	00FF	STARTA	00E0
STARTVND	FB17	STRTCRPT	FBBD	SUBLOOP	F782	SUBLOOP2	F78E
SUBMOD	F785	SUBOUT	F79C	SWCHA	u0280	SWCHB	u0282
SYNC	F7CB	T	F52D	TEMP0	u00D0	TEMP1	u00D1
TEMP2	u00D2	TEMP3	u00D3	TEMP4	u00D4	TEMP5	u00D5
TEST0	u00C0	TEST1	u00C1	TESTW0	u00C2	TESTW1	u00C4
TIAOLOOP	F610	TIM64T	u0296	TIM64TI	u029E	TIM8T	u0295
WSYNC	0024	ZEROLP	F7CE				

o errors