

```

FC70:A0 06      193      LDY   #6           ;CODE=SCROLL/RRA0981
FC72:D0 B5      FC29     BNE   GOTOCX1      ;DO 40/80 /RRA0981
FC74:           195 *
FC74:           196 * Jump here to swap out ROMs
FC74:           197 * for interrupt handlers in peripheral cards
FC74:           198 *
FC74:8D 06 C0    199      IRQUSER STA SETSLOTCXROM ;switch in slots
FC77:6C FE 03    200      JMP   ($3FE)      ;and jump to user
FC7A:           201 *
FC7A:           202 * IRQDONE ($C3F4) jumps here after interrupt
FC7A:           203 * because this cannot be done from $Cn00 space
FC7A:           204 *
FC7A:68         205      IRQDONE2 PLA          ;Fix $C800 space
FC7B:8D F8 07    206      STA   MSLOT        ;restore MSLOT
FC7E:C9 C1       207      CMP   #$C1         ;valid Cn?
FC80:90 0D      FC8F     BCC   IRQNOSLT
FC82:8D FF CF    209      STA   $CFFF        ;Deselect all $C800
FC85:A0 00       210      LDY   #0
FC87:A6 01       211      LDX   $1
FC89:85 01       212      STA   $1
FC8B:B1 00       213      LDA   ($0),Y      ;do $Cn00 reference
FC8D:86 01       214      STX   $1          ;fix zp location
FC8F:8D 07 C0    215      IRQNOSLT STA SETINTCXROM
FC92:4C 7C C4    216      JMP   IRQFIX      ;and restore the machine state
FC95:           217 *
FC95:90 02      FC99     218      DOCOUT1 BCC   DOCOUT2 ;don't mask controls
FC97:25 32       219      AND   INVFLG      ;apply inverse mask
FC99:4C F7 FD    220      DOCOUT2 JMP   COUTZ1   ;go back to COUT1
FC9C:           221 *
FC9C:           0000     222      DS    F80RG+$49C-*,0 ;pad to clreol
FC9C:           223 *
FC9C:           224 * Note: bytes CLREOL and CLREOLZ ($38 and $18)
FC9C:           225 * are used by slot test at $FBB7.
FC9C:           226 *
FC9C:38         227      CLREOL SEC          ;say it is EOL
FC9D:90         228      DFB   $90          ;'BCC' opcode
FC9E:18         229      CLREOLZ CLC        ;say it is EOLZ
FC9F:84 2A      230      STY   BAS2L        ;save Y in temp
FCA1:A0 07      231      LDY   #7          ;code=CLREOL
FCA3:B0 78      FD1D     232      BCS   GOTOCX2 ;do it
FCA5:C8         233      INY           ;code 8=CLREOLZ
FCA6:D0 75      FD1D     234      BNE   GOTOCX2
FCA8:           235 *
FCA8:38         236      WAIT SEC          ;enter with count in A
FCA9:48         237      WAIT2 PHA         ;delay is:
FCAA:E9 01      238      WAIT3 SBC   #$01
FCAC:D0 FC      FCAA     239      BNE   WAIT3 ;13+11*A+5*A*A cycles
FCAE:68         240      PLA          ;@ 1.023 usec per cycle
FCAF:E9 01      241      SBC   #$01
FCB1:D0 66      FCA9     242      BNE   WAIT2
FCB3:60         243      RTS
FCB4:           244 *
FCB4:E6 42      245      NXTA4 INC   A4L      ;INCR 2-BYTE A4
FCB6:D0 02      FCBA     246      BNE   NXTA1 ; AND A1

```

```

FCB8:E6 43      247      INC   A4H
FCBA:A5 3C      248      LDA   A1L      ;INCR 2-BYTE A1.
FCBC:C5 3E      249      CMP   A2L      ; AND COMPARE TO A2
FCBE:A5 3D      250      LDA   A1H      ; (CARRY SET IF >=)
FCC0:E5 3F      251      SBC   A2H
FCC2:E6 3C      252      INC   A1L
FCC4:D0 02      FCC8     253      BNE   RTS4B
FCC6:E6 3D      254      INC   A1H
FCC8:60         255      RTS4B RTS
FCC9:           256 *
FCC9:8D 07 C0    257      HEADR STA   SETINTCXROM ;force internal ROM
FCCC:20 67 C5    258      JSR   XHEADER ;write header
FCCF:4C C5 FE    259      JMP   RETCX1 ;force slots and return
FCD2:           260 *
FCD2:           261 * For the disassembler to be able to do I/O to slots,
FCD2:           262 * it cannot make calls to the I/O routines with the
FCD2:           263 * internal ROM switched in. This stuff switches the
FCD2:           264 * ROM out for such instances.
FCD2:           265 *
FCD2:8D 06 C0    266      ERR3 STA   SETSLOTCXROM ;force slot ROM
FCD5:20 4A F9    267      JSR   PRBL2 ;tab to the error
FCD8:A9 DE      268      LDA   #$DE ;to print a caret ""
FCDA:20 ED FD    269      JSR   COUT ;print it
FCDD:20 3A FF    270      JSR   BELL ;and beep
FCE0:4C F0 FC    271      JMP   GETINST1 ;and go get next instruction
FCE3:           272 *
FCE3:8D 06 C0    273      DISLIN STA SETSLOTCXROM ;force slot ROM
FCE6:20 D0 F8    274      JSR   INSTDSP ;disassemble the instruction
FCE9:20 53 F9    275      JSR   PCADJ ;calculate new PC
FCEC:84 3B      276      STY   PCH ;and update PC
FCEE:85 3A      277      STA   PCL
FCF0:           278 *
FCF0:           279 * NOTE: The entry point GETINST1 is hard-coded in
FCF0:           280 * BFUNC of the Video firmware.
FCF0:           281 *
FCF0:A9 A1      282      GETINST1 LDA #$A1 ;get mini-prompt "!"
FCF2:85 33      283      STA   PROMPT
FCF4:20 67 FD    284      JSR   GETLNZ ;go get a line of input
FCF7:8D 07 C0    285      STA   SETINTCXROM ;force internal ROM
FCFA:4C 9C CF    286      JMP   DOINST ;and return to CX space
FCFD:           287 *
FCFD:B9 00 02    288      UPMON LDA IN,Y ;get character
FD00:C8         289      INY ;point to next char
FD01:C9 E1      290      CMP   #$E1 ;is it lowercase?
FD03:90 06      FDOB     291      BCC   UPMON2 ;=>nope
FD05:C9 FB      292      CMP   #$FB ;lowercase?
FD07:B0 02      FDOB     293      BCS   UPMON2 ;=>nope
FD09:29 DF      294      AND   #$DF ;else upshift
FD0B:60         295      UPMON2 RTS
FD0C:           296 *
FD0C:A0 0B      297      RDKEY LDY   #$B ;code=RDKEY
FD0E:D0 03      FD13     298      BNE   RDKEY0 ;allow $FD10 entry
FD10:4C 18 FD    299      FD10 JMP   RDKEY1 ;if enter here, do nothing
FD13:20 B4 FB    300      RDKEY0 JSR   GOTOCX ;display cursor

```