# Disassembly of the 2Mhz Time Cruncher V5.1

While browsing through the Web Archive, I had found a machine code monitor disassembly listing for the Time Cruncher V5. The original Time crunch was by Matcham/Network, but Version 5 came along and was done by Tim Rogers (of Stoat and Tim fame). This is probably a modified version of the 2Mhz cruncher, done way back in 1991. So if you can, go disassemble it in a M/C monitor and if works 100% try it out. Feel free to make improvements with the tool.

```
Time Cruncher V5.1 Disassembly.

; Some areas of code are relocated before being used. For these parts,
; I have shown both addresses in the disassembly.

; BASIC PRG ( 1991 SYS2066 )
0801   .BY $0B,$08,$C7,$07,$9E,$32,$30,$36
0809   .BY $36,$00,$00,$00,$00,$00,$00,$00
0811   .BY $00

; Start of machine code
0812   A2 FF      LDX #$FF
0814   9A         TXS
0815   A9 06      LDA #$06
0817   8D 20 D0   STA $D020   ; blue border
081A   20 33 0B   JSR $0B33   ; Display text below on screen

; 2MHz TIME CRUNCH V5.1
;
; LOAD:
081D   .BY $93,$0E,$08,$05,$32,$6D,$68,$5A
0825   .BY $20,$74,$49,$4D,$45,$20,$63,$52
082D   .BY $55,$4E,$43,$48,$20,$76,$35,$2E
0835   .BY $31,$20,$20,$20,$20,$20,$20,$20
083D   .BY $0D,$20,$20,$20,$20,$20,$20,$20
0845   .BY $20,$20,$20,$20,$20,$20,$20,$20
084D   .BY $20,$20,$20,$20,$20,$20,$20,$20
0855   .BY $20,$20,$20,$20,$20,$20,$20,$20
085D   .BY $0D,$0D,$6C,$4F,$41,$44,$20,$3A
0865   .BY $00
; A lot of extra spaces above!

;
0866   20 B1 0A   JSR $0AB1   ; Input filename
0869   AD 7E 04   LDA $047E   ; get 1st charcter
086C   F0 04      BEQ $0872
086E   C9 24      CMP #$24   ; @ character
0870   D0 05      BNE $0877
0872   EA         NOP   ; unused place for JMP
0873   EA         NOP   ; Is for access to disk commands
0874   EA         NOP   ; but removed form this dissasembly
0875   F0 9B      BEQ $0812   ; Restart
```

```
0877   8A         TXA
0878   48         PHA
0879   20 30 0B   JSR $0B30   ; Display text below on screen

; SAVE@:
087C   .BY $73,$41,$56,$45,$40,$3A,$00

;
0883   20 B1 0A   JSR $0AB1   ; Input filename
0886   E8         INX
0887   E8         INX   ; add two to filename length
0888   86 8E      STX $8E   ; Store filename length
088A   20 30 0B   JSR $0B30   ; Display text below on screen

; RUN $____
088D   .BY $20,$6A,$6D,$70,$20,$24,$A4,$A4
0895   .BY $A4,$A4,$9D,$9D,$9D,$9D,$00

;
089C   20 EA 0A   JSR $0AEA   ; Input two character hex number
089F   8D 22 0F   STA $0F22   ; store
08A2   20 EA 0A   JSR $0AEA   ; Input two character hex number
08A5   8D 21 0F   STA $0F21   ; store
08A8   20 30 0B   JSR $0B30   ; Display text below on screen

; $01=$__
08AB   .BY $20,$24,$30,$31,$3D,$24,$A4,$A4
08B3   .BY $9D,$9D,$00

;
08B6   20 EA 0A   JSR $0AEA   ; Input two character hex number
08B9   8D 0F 0F   STA $0F0F   ; store
08BC   20 30 0B   JSR $0B30   ; Display text below on screen

; STEP $01-$08=$__
08BF   .BY $73,$54,$45,$50,$20,$24,$30,$31
08C7   .BY $2D,$24,$30,$38,$0D,$20,$20,$20
08CF   .BY $20,$20,$24,$A4,$A4,$9D,$9D,$00

;
08D7   20 EA 0A   JSR $0AEA   ; Input two character hex number
08DA   F0 51      BEQ $092D   ; if 0 restart
08DC   C9 09      CMP #$09
08DE   B0 4D      BCS $092D   ; if more than 8 restart
08E0   AA         TAX
08E1   8D B8 0D   STA $0DB8   ; store
08E4   8D DA 0D   STA $0DDA   ; store
08E7   69 07      ADC #$07    ; add 7
08E9   8D 2B 0F   STA $0F2B   ; store
08EC   A9 01      LDA #$01
```

```
08EE   0A         ASL
08EF   CA         DEX
08F0   D0 FC      BNE $08EE
08F2   8D 15 0C   STA $0C15   ; store bit position
08F5   20 30 0B   JSR $0B30   ; Display text below on screen

; SPACE=CRUNCH - RUN/STOP=RESTART
08F8   .BY $0D,$73,$50,$41,$43,$45,$20,$3D
0900   .BY $20,$63,$52,$55,$4E,$43,$48,$20
0908   .BY $20,$2D,$20,$20,$72,$55,$4E,$2F
0910   .BY $73,$54,$4F,$50,$20,$3D,$20,$72
0918   .BY $45,$53,$54,$41,$52,$54,$00

;
091F   A9 10      LDA #$10
0921   2C 01 DC   BIT $DC01
0924   10 07      BPL $092D
0926   D0 F9      BNE $0921
0928   F0 06      BEQ $0930   ; if space then continue
092A   20 42 F6   JSR $F642
092D   4C 12 08   JMP $0812   ; restart
0930   20 E3 0A   JSR $0AE3   ; Get character under cursor
0933   E6 CC      INC $CC   ; cursor off
0935   68         PLA
0936   A2 7E      LDX #$7E
0938   A0 04      LDY #$04
093A   20 BD FF   JSR $FFBD   ; set load filename on screen $047E

; Load file to compress
093D   A9 08      LDA #$08
093F   85 B8      STA $B8
0941   85 BA      STA $BA
0943   A2 01      LDX #$01
0945   0A         ASL   ; change A to #$10
0946   86 FB      STX $FB
0948   85 FC      STA $FC   ; start to load file at $1001
094A   A9 60      LDA #$60
094C   85 B9      STA $B9
094E   20 D5 F3   JSR $F3D5
0951   A5 BA      LDA $BA
0953   20 09 ED   JSR $ED09
0956   A5 B9      LDA $B9
0958   20 C7 ED   JSR $EDC7
095B   20 13 EE   JSR $EE13
095E   20 13 EE   JSR $EE13   ; ignore start address of file
0961   20 13 EE   JSR $EE13
0964   78         SEI
0965   A0 00      LDY #$00
0967   E6 01      INC $01
0969   91 FB      STA ($FB),Y
096B   C6 01      DEC $01
```

```
096D   8D 20 D0   STA $D020   ; change border colour
0970   E6 FB      INC $FB
0972   D0 04      BNE $0978
0974   E6 FC      INC $FC
0976   F0 B2      BEQ $092A
0978   24 90      BIT $90
097A   50 E5      BVC $0961
097C   20 EF ED   JSR $EDEF
097F   20 42 F6   JSR $F642

; start compression of file
0982   78         SEI
0983   A9 00      LDA #$00
0985   85 C6      STA $C6   ; clear keyboard
0987   8D 11 D0   STA $D011   ; blank screen
098A   EE 30 D0   INC $D030   ; 2MHz mode on (if c128)
098D   85 01      STA $01   ; switch out roms
098F   20 63 0B   JSR $0B63   ; move whole file to the end of memory
0992   20 B8 0C   JSR $0CB8   ; do compression
0995   A9 37      LDA #$37
0997   85 01      STA $01   ; switch on roms
0999   A9 FC      LDA #$FC
099B   8D 30 D0   STA $D030   ; 2MHz mode off
099E   A9 1B      LDA #$1B
09A0   8D 11 D0   STA $D011   ; unblank screen
09A3   20 30 0B   JSR $0B30   ; Display text below on screen

; OLD END=$
09A6   .BY $0D,$20,$6F,$4C,$44,$20,$65,$4E
09AE   .BY $44,$3D,$24,$00

;
09B2   A5 FC      LDA $FC
09B4   E9 07      SBC #$07
09B6   AA         TAX
09B7   A5 FB      LDA $FB
09B9   20 14 0B   JSR $0B14   ; Display 4 character hex number
09BC   20 30 0B   JSR $0B30   ; Display text below on screen

; NEW END=$
09BF   .BY $91,$20,$6E,$45,$57,$20,$65,$4E
09C7   .BY $44,$3D,$24,$00

;
09CB   A5 AF      LDA $AF
09CD   E9 05      SBC #$05
09CF   AA         TAX
09D0   A5 AE      LDA $AE
09D2   20 14 0B   JSR $0B14   ; Display 4 character hex number
09D5   20 30 0B   JSR $0B30   ; Display text below on screen
```

```
; 1-SAVE 2-RUN 3-NEW FILE 4-RESET
09D8   .BY $0D,$31,$3E,$73,$41,$56,$45,$20
09E0   .BY $32,$3E,$72,$55,$4E,$20,$33,$3E
09E8   .BY $6E,$45,$57,$20,$66,$49,$4C,$45
09F0   .BY $20,$34,$3E,$72,$65,$73,$65,$74
09F8   .BY $00

; Check for 1234 key option
09F9   A9 00      LDA #$00
09FB   85 CC      STA $CC   ; cursor on
09FD   A9 06      LDA #$06
09FF   8D 20 D0   STA $D020   ; Blue border
0A02   20 E4 FF   JSR $FFE4
0A05   AA         TAX
0A06   20 E3 0A   JSR $0AE3   ; Get character under cursor
0A09   E6 CC      INC $CC   ; cursor off
0A0B   E0 31      CPX #$31   ; option 1
0A0D   F0 36      BEQ $0A45   ; Save file
0A0F   E0 32      CPX #$32   ; option 2
0A11   F0 0B      BEQ $0A1E   ; Run file
0A13   E0 33      CPX #$33   ; option 3
0A15   F0 2B      BEQ $0A42   ; New file
0A17   E0 34      CPX #$34   ; option 4
0A19   D0 DE      BNE $09F9
0A1B   4C E2 FC   JMP $FCE2   ; Reset

; Run newly compressed file
0A1E   78         SEI
0A1F   A0 16      LDY #$16   ; move code into place
0A21   B9 2E 0A   LDA $0A2E,Y
0A24   99 FF 00   STA $00FF,Y
0A27   88         DEY
0A28   D0 F7      BNE $0A21
0A2A   E6 01      INC $01
0A2C   4C 00 01   JMP $0100

; Continuation of running newly compressed file code
; to be run from $0100
; move compressed file from $0E00-$FFFF to $0800-$F9FF and run it
0A2F/0100   B9 00 0E   LDA $0E00,Y
0A32/0103   99 00 08   STA $0800,Y
0A35/0106   C8         INY
0A36/0107   D0 F7      BNE $0A2F/$0100
0A38/0109   EE 05 01   INC $0105
0A3B/010C   EE 02 01   INC $0102
0A3E/010F   D0 EF      BNE $0A2F/$0100
0A40/0111   C6 01      DEC $01
0A42/0113   4C 12 08   JMP $0812

; Save compressed file
0A45   A9 40      LDA #$40   ; zero char
```

```
0A47   8D CC 04   STA $04CC  ; put on screen
0A4A   A5 8E      LDA $8E   ; Filename length
0A4C   A2 CC      LDX #$CC
0A4E   A0 04      LDY #$04
0A50   20 BD FF   JSR $FFBD   ; Set save filename on screen $04CC
0A53   E6 CC      INC $CC
0A55   A9 61      LDA #$61
0A57   85 B9      STA $B9
0A59   20 D5 F3   JSR $F3D5
0A5C   A5 BA      LDA $BA
0A5E   20 0C ED   JSR $ED0C
0A61   A5 B9      LDA $B9
0A63   20 B9 ED   JSR $EDB9
0A66   A9 01      LDA #$01
0A68   85 8B      STA $8B
0A6A   20 DD ED   JSR $EDDD
0A6D   A9 0E      LDA #$0E   ; Save from $0E01
0A6F   85 8C      STA $8C
0A71   A9 08      LDA #$08   ; Save to reload at $0801
0A73   20 DD ED   JSR $EDDD
0A76   78         SEI
0A77   A0 00      LDY #$00
0A79   E6 01      INC $01
0A7B   B1 8B      LDA ($8B),Y
0A7D   C6 01      DEC $01
0A7F   20 DD ED   JSR $EDDD
0A82   8D 20 D0   STA $D020   ; change border
0A85   E6 8B      INC $8B
0A87   D0 02      BNE $0A8B
0A89   E6 8C      INC $8C
0A8B   A5 8B      LDA $8B
0A8D   C5 AE      CMP $AE
0A8F   A5 8C      LDA $8C
0A91   E5 AF      SBC $AF
0A93   90 E1      BCC $0A76
0A95   20 FE ED   JSR $EDFE
0A98   24 B9      BIT $B9
0A9A   30 11      BMI $0AAD
0A9C   A5 BA      LDA $BA
0A9E   20 0C ED   JSR $ED0C
0AA1   A5 B9      LDA $B9
0AA3   29 EF      AND #$EF
0AA5   09 E0      ORA #$E0
0AA7   20 B9 ED   JSR $EDB9
0AAA   20 FE ED   JSR $EDFE
0AAD   58         CLI
0AAE   4C F9 09   JMP $09F9   ; Check for 1234 key option

; Input filename
0AB1   A2 00      LDX #$00
```

```
0AB3   86 C6      STX $C6   ; clear buffer
0AB5   86 CC      STX $CC   ; cursor on
0AB7   86 8D      STX $8D
0AB9   20 E4 FF   JSR $FFE4
0ABC   C9 03      CMP #$03   ; RUN STOP key
0ABE   F0 82      BEQ $0A42   ; If so then restart
0AC0   A6 8D      LDX $8D
0AC2   F0 08      BEQ $0ACC   ; branch if X = 0
0AC4   C9 14      CMP #$14   ; Delete key
0AC6   F0 18      BEQ $0AE0
0AC8   C9 0D      CMP #$0D   ; Return key
0ACA   F0 17      BEQ $0AE3   ; If so then End input
0ACC   29 7F      AND #$7F   ; Remove high byte
0ACE   C9 20      CMP #$20   ; ignore spaces
0AD0   90 E7      BCC $0AB9
0AD2   E0 10      CPX #$10   ; upto 16 characters
0AD4   F0 E3      BEQ $0AB9
0AD6   E8         INX
0AD7   0A         ASL
0AD8   C9 82      CMP #$82
0ADA   6A         ROR
0ADB   20 D2 FF   JSR $FFD2   ; display character
0ADE   90 D7      BCC $0AB7
0AE0   CA         DEX   ; reduce filename length
0AE1   10 F8      BPL $0ADB   ; branch back (always)

; Get character under cursor
0AE3   A4 D3      LDY $D3   ; Cursor column number
0AE5   B1 D1      LDA ($D1),Y   ; Get value of character on screen
0AE7   30 FC      BMI $0AE5   ; Wait until cursor off
0AE9   60         RTS     ; continue

; Input two character hex number
0AEA   A9 00      LDA #$00
0AEC   85 C6      STA $C6
0AEE   20 F5 0A   JSR $0AF5   ; do 1st character
0AF1   0A         ASL
0AF2   0A         ASL
0AF3   0A         ASL
0AF4   0A         ASL   ; shift 1st and do 2nd
0AF5   85 8B      STA $8B
0AF7   20 E4 FF   JSR $FFE4
0AFA   C9 03      CMP #$03   ; RUN STOP key
0AFC   F0 C0      BEQ $0ABE   ; If so then restart
0AFE   AA         TAX
0AFF   E9 30      SBC #$30
0B01   C9 0A      CMP #$0A
0B03   90 06      BCC $0B0B   ; test for number
0B05   E9 07      SBC #$07
0B07   C9 10      CMP #$10
0B09   B0 EC      BCS $0AF7   ; test for letter
```

```
0B0B   A8         TAY
0B0C   8A         TXA
0B0D   20 D2 FF   JSR $FFD2   ; output character
0B10   98         TYA
0B11   05 8B      ORA $8B
0B13   60         RTS

; Display 4 character hex number in AX
; (A very elegant piece of code in my opinion)
0B14   48         PHA
0B15   8A         TXA
0B16   20 1A 0B   JSR $0B1A   ; display 2 character hex number in A
0B19   68         PLA
0B1A   48         PHA
0B1B   4A         LSR
0B1C   4A         LSR
0B1D   4A         LSR
0B1E   4A         LSR
0B1F   20 25 0B   JSR $0B25   ; display 1 character hex number
0B22   68         PLA
0B23   29 0F      AND #$0F
0B25   C9 0A      CMP #$0A
0B27   90 02      BCC $0B2B   ; if number then branch
0B29   69 06      ADC #$06
0B2B   69 30      ADC #$30
0B2D   4C D2 FF   JMP $FFD2   ; output character

; Display text just after JSR on screen
0B30   20 E3 0A   JSR $0AE3   ; Get character under cursor
0B33   68         PLA    ; get low memory loc after JSR
0B34   A8         TAY    ; store low in Y
0B35   68         PLA    ; get high memory loc after JSR
0B36   85 8C      STA $8C   ; store high byte
0B38   A2 01      LDX #$01
0B3A   8E 86 02   STX $0286   ; white characters
0B3D   CA         DEX
0B3E   86 8B      STX $8B   ; store #$00
0B40   A9 0D      LDA #$0D   ; Return char
0B42   20 D2 FF   JSR $FFD2   ; Output Return
0B45   90 09      BCC $0B50   ; Jump to $0B50
0B47   C8         INY    ; incrememnt pointer
0B48   D0 02      BNE $0B4C
0B4A   E6 8C      INC $8C   ; and high byte in necess
0B4C   B1 8B      LDA ($8B),Y   ; read next memory loc in A
0B4E   F0 05      BEQ $0B55   ; Is A = #$00, if yes branch
0B50   20 D2 FF   JSR $FFD2   ; Output char in A
0B53   90 F2      BCC $0B47   ; Jump to $0B47
0B55   A9 03      LDA #$03
0B57   8D 86 02   STA $0286   ; cyan characters
0B5A   8D 87 02   STA $0287   ; and cursor
```

```
0B5D   A5 8C      LDA $8C
0B5F   48         PHA   ; push return address after
0B60   98         TYA   ; text back onto the stack
0B61   48         PHA
0B62   60         RTS

; Move whole file to the end of memory
0B63   A0 00      LDY #$00
0B65   98         TYA
0B66   38         SEC
0B67   E5 FB      SBC $FB
0B69   08         PHP
0B6A   18         CLC
0B6B   69 01      ADC #$01
0B6D   85 FD      STA $FD
0B6F   98         TYA
0B70   69 10      ADC #$10
0B72   28         PLP
0B73   E5 FC      SBC $FC
0B75   85 FE      STA $FE
0B77   A5 FB      LDA $FB
0B79   A6 FC      LDX $FC
0B7B   85 02      STA $02
0B7D   CA         DEX
0B7E   86 03      STX $03
0B80   CE 87 0B   DEC $0B87
0B83   B1 02      LDA ($02),Y
0B85   99 00 00   STA $0000,Y
0B88   C8         INY
0B89   D0 F8      BNE $0B83
0B8B   E0 10      CPX #$10
0B8D   B0 EE      BCS $0B7D
0B8F   8C 87 0B   STY $0B87
0B92   A5 FC      LDA $FC
0B94   E9 07      SBC #$07
0B96   A6 FB      LDX $FB
0B98   8E 28 0E   STX $0E28
0B9B   8D 2A 0E   STA $0E2A
0B9E   D0 02      BNE $0BA2
0BA0   E9 01      SBC #$01
0BA2   CA         DEX
0BA3   8E 3D 0E   STX $0E3D
0BA6   8D 3E 0E   STA $0E3E
0BA9   60         RTS

; Do compression part 2 (main part)
; Relocated to $0002-$00FF before being used
0BAA/0002   85 28      STA $28
0BAC/0004   86 74      STX $74
0BAE/0006   8A         TXA
0BAF/0007   E5 71      SBC $71
```

```
0BB1/0009   AA         TAX
0BB2/000A   A5 75      LDA $75
0BB4/000C   E5 72      SBC $72
0BB6/000E   F0 02      BEQ $0BBA/$0012
0BB8/0010   A2 FF      LDX #$FF
0BBA/0012   A0 00      LDY #$00
0BBC/0014   C8         INY
0BBD/0015   CA         DEX
0BBE/0016   F0 06      BEQ $0BC6/$001E
0BC0/0018   B1 71      LDA ($71),Y
0BC2/001A   D1 74      CMP ($74),Y
0BC4/001C   F0 F6      BEQ $0BBC/$0014
0BC6/001E   88         DEY
0BC7/001F   C0 00      CPY #$00
0BC9/0021   90 02      BCC $0BCD/$0025
0BCB/0023   D0 0A      BNE $0BD7/$002F
0BCD/0025   A6 74      LDX $74
0BCF/0027   A9 00      LDA #$00
0BD1/0029   A0 00      LDY #$00
0BD3/002B   84 74      STY $74
0BD5/002D   F0 49      BEQ $0C20/$0078
0BD7/002F   A6 75      LDX $75
0BD9/0031   E8         INX
0BDA/0032   D0 0F      BNE $0BEB/$0043
0BDC/0034   98         TYA
0BDD/0035   65 74      ADC $74
0BDF/0037   90 0A      BCC $0BEB/$0043
0BE1/0039   85 3D      STA $3D
0BE3/003B   98         TYA
0BE4/003C   E9 FF      SBC #$FF
0BE6/003E   C5 20      CMP $20
0BE8/0040   90 E3      BCC $0BCD/$0025
0BEA/0042   A8         TAY
0BEB/0043   84 20      STY $20
0BED/0045   A6 74      LDX $74
0BEF/0047   A5 75      LDA $75
0BF1/0049   86 89      STX $89
0BF3/004B   85 93      STA $93
0BF5/004D   C0 FE      CPY #$FE
0BF7/004F   90 D6      BCC $0BCF/$0027
0BF9/0051   B0 35      BCS $0C30/$0088
;
0BFB/0053   A6 71      LDX $71
0BFD/0055   A4 72      LDY $72
0BFF/0057   E8         INX
0C00/0058   D0 03      BNE $0C05/$005D
0C02/005A   C8         INY
0C03/005B   F0 63      BEQ $0C68/$00C0
0C05/005D   84 75      STY $75
0C07/005F   A9 00      LDA #$00
```

```
0C09/0061   C6 01      DEC $01
0C0B/0063   EE 20 D0   INC $D020   ; change border
0C0E/0066   85 01      STA $01
0C10/0068   85 20      STA $20
0C12/006A   85 74      STA $74
0C14/006C   A9 08      LDA #$08   ; STEP bit position
0C16/006D   85 DE      STA $DE
0C18/0070   AD 00 FF   LDA $FF00
0C1B/0073   DD 00 FF   CMP $FF00,X
0C1E/0076   F0 8A      BEQ $0BAA/$0002
0C20/0078   E8         INX
0C21/0079   D0 F8      BNE $0C1B/$0073
0C23/007B   E6 75      INC $75
0C25/007D   F0 04      BEQ $0C2B/$0083
0C27/007F   C6 DE      DEC $DE
0C29/0081   D0 F0      BNE $0C1B/$0073
0C2B/0083   A4 20      LDY $20
0C2D/0085   F0 3B      BEQ $0C6A/$00C2
0C2F/0087   38         SEC
0C30/0088   A9 00      LDA #$00
0C32/008A   E5 20      SBC $20
0C34/008C   08         PHP
0C35/008D   18         CLC
0C36/008E   E5 71      SBC $71
0C38/0090   85 DA      STA $DA
0C3A/0092   A9 00      LDA #$00
0C3C/0094   E5 72      SBC $72
0C3E/0096   28         PLP
0C3F/0097   E9 00      SBC #$00
0C41/0099   85 DB      STA $DB
0C43/009B   C0 02      CPY #$02
0C45/009D   B0 08      BCS $0C4F/$00A7
0C47/009F   AA         TAX
0C48/00A0   D0 1D      BNE $0C67/$00BF
0C4A/00A2   20 5E 0D   JSR $0D5E
0C4D/00A5   D0 0A      BNE $0C59/$00B1
0C4F/00A7   D0 05      BNE $0C56/$00AE
0C51/00A9   20 A1 0D   JSR $0DA1
0C54/00AC   F0 03      BEQ $0C59/$00B1
0C56/00AE   20 C3 0D   JSR $0DC3
0C59/00B1   A5 20      LDA $20
0C5B/00B3   38         SEC
0C5C/00B4   65 71      ADC $71
0C5E/00B6   85 71      STA $71
0C60/00B8   90 99      BCC $0BFB/$0053
0C62/00BA   E6 72      INC $72
0C64/00BC   D0 95      BNE $0BFB/$0053
0C66/00BE   60         RTS
;
0C67/00BF   88         DEY
0C68/00C0   B1 71      LDA ($71),Y
```

```
0C6A/00C2   8D FF FF   STA $FFFF
0C6D/00C5   E6 C3      INC $C3
0C6F/00C7   D0 02      BNE $0C73/$00CB
0C71/00C9   E6 C4      INC $C4
0C73/00CB   E6 CE      INC $CE
0C75/00CD   A2 00      LDX #$00
0C77/00CF   E8         INX
0C78/00D0   D0 03      BNE $0C7D/$00D5
0C7A/00D2   20 2A 0D   JSR $0D2A
0C7D/00D5   98         TYA
0C7E/00D6   F0 DB      BEQ $0C5B/$00B3
0C80/00D8   08         PHP

; Filler
0C81/00D9   .BY $00,$00,$00,$00,$00,$00,$00,$00
0C89/00E1   .BY $00,$00,$00,$00,$00,$00,$00,$00
0C91/00E9   .BY $00,$00,$00,$00,$00,$00,$00,$00
0C99/00F1   .BY $00,$00,$00,$00,$00,$00,$00,$00
0CA1/00F9   .BY $00,$00,$00,$00,$00,$00,$00

; Swap $0002-$00FF and $0BAA-$0CA7
0CA8   A2 02      LDX #$02
0CAA   B5 00      LDA $00,X
0CAC   BC A8 0B   LDY $0BA8,X
0CAF   94 00      STY $00,X
0CB1   9D A8 0B   STA $0BA8,X
0CB4   E8         INX
0CB5   D0 F3      BNE $0CAA
0CB7   60         RTS

; Do compression part 1
0CB8   A5 FD      LDA $FD
0CBA   A6 FE      LDX $FE
0CBC   8D 19 0C   STA $0C19
0CBF   8E 1A 0C   STX $0C1A
0CC2   A9 2C      LDA #$2C
0CC4   A2 0F      LDX #$0F   ; position to store compressed code
0CC6   8D 6B 0C   STA $0C6B
0CC9   8E 6C 0C   STX $0C6C
0CCC   20 A8 0C   JSR $0CA8   ; Swap $0002-$00FF and $0BAA-$0CA7
0CCF   20 53 00   JSR $0053
0CD2   20 2C 0D   JSR $0D2C
0CD5   A5 D9      LDA $D9
0CD7   8D 39 0E   STA $0E39   ; store value in decompression routine
0CDA   A9 08      LDA #$08
0CDC   38         SEC
0CDD   E5 D8      SBC $D8
0CDF   D0 0A      BNE $0CEB
0CE1   A5 C3      LDA $C3
0CE3   D0 02      BNE $0CE7
```

```
0CE5   C6 C4      DEC $C4
0CE7   C6 C3      DEC $C3
0CE9   A9 08      LDA #$08
0CEB   8D 3A 0E   STA $0E3A   ; store value in decompression routine
0CEE   A5 C3      LDA $C3
0CF0   48         PHA
0CF1   E9 45      SBC #$45
0CF3   8D 3B 0E   STA $0E3B   ; store value in decompression routine
0CF6   A5 C4      LDA $C4
0CF8   48         PHA
0CF9   E9 07      SBC #$07
0CFB   8D 3C 0E   STA $0E3C   ; store value in decompression routine
0CFE   A5 C4      LDA $C4
0D00   E9 0D      SBC #$0D
0D02   8D 34 0E   STA $0E34
0D05   20 A8 0C   JSR $0CA8    ; Swap $0002-$00FF and $0BAA-$0CA7
0D08   68         PLA
0D09   85 AF      STA $AF
0D0B   68         PLA
0D0C   85 AE      STA $AE
0D0E   60         RTS

; More decompression code
0D0F   20 2C 0D   JSR $0D2C
0D12   A5 DD      LDA $DD
0D14   F0 F8      BEQ $0D0E
0D16   A2 04      LDX #$04
0D18   20 82 0D   JSR $0D82
0D1B   86 DD      STX $DD
0D1D   A9 FD      LDA #$FD
0D1F   A2 08      LDX #$08
0D21   20 82 0D   JSR $0D82
0D24   A9 07      LDA #$07
0D26   A2 03      LDX #$03
0D28   D0 58      BNE $0D82
0D2A   E6 DD      INC $DD
0D2C   A5 CE      LDA $CE
0D2E   C9 06      CMP #$06
0D30   B0 08      BCS $0D3A
0D32   A2 00      LDX #$00
0D34   86 CE      STX $CE
0D36   A2 03      LDX #$03
0D38   D0 48      BNE $0D82
0D3A   E9 06      SBC #$06
0D3C   C9 10      CMP #$10
0D3E   B0 0F      BCS $0D4F
0D40   A2 04      LDX #$04
0D42   20 82 0D   JSR $0D82
0D45   86 CE      STX $CE
0D47   18         CLC
0D48   20 69 0D   JSR $0D69
```

```
0D4B   A9 03      LDA #$03
0D4D   D0 31      BNE $0D80
0D4F   A2 08      LDX #$08
0D51   20 82 0D   JSR $0D82
0D54   86 CE      STX $CE
0D56   38         SEC
0D57   20 69 0D   JSR $0D69
0D5A   A9 03      LDA #$03
0D5C   D0 22      BNE $0D80
0D5E   20 0F 0D   JSR $0D0F
0D61   A5 DA      LDA $DA
0D63   A2 08      LDX #$08
0D65   20 82 0D   JSR $0D82
0D68   38         SEC
0D69   66 D9      ROR $D9
0D6B   C6 D8      DEC $D8
0D6D   D0 10      BNE $0D7F
0D6F   A5 D9      LDA $D9
0D71   A0 00      LDY #$00
0D73   91 C3      STA ($C3),Y
0D75   A9 08      LDA #$08
0D77   85 D8      STA $D8
0D79   E6 C3      INC $C3
0D7B   D0 02      BNE $0D7F
0D7D   E6 C4      INC $C4
0D7F   60         RTS

; More decompression code
0D80   A2 02      LDX #$02
0D82   6A         ROR
0D83   66 D9      ROR $D9
0D85   C6 D8      DEC $D8
0D87   D0 14      BNE $0D9D
0D89   85 DC      STA $DC
0D8B   A5 D9      LDA $D9
0D8D   A0 00      LDY #$00
0D8F   91 C3      STA ($C3),Y
0D91   A9 08      LDA #$08
0D93   85 D8      STA $D8
0D95   A5 DC      LDA $DC
0D97   E6 C3      INC $C3
0D99   D0 02      BNE $0D9D
0D9B   E6 C4      INC $C4
0D9D   CA         DEX
0D9E   D0 E2      BNE $0D82
0DA0   60         RTS

; More decompression code
0DA1   20 0F 0D   JSR $0D0F
0DA4   A5 DA      LDA $DA
```

```
0DA6   A2 08      LDX #$08
0DA8   A4 DB      LDY $DB
0DAA   D0 06      BNE $0DB2
0DAC   20 82 0D   JSR $0D82
0DAF   18         CLC
0DB0   90 0B      BCC $0DBD
0DB2   20 82 0D   JSR $0D82
0DB5   A5 DB      LDA $DB
0DB7   A2 03      LDX #$03   ; STEP value
0DB9   20 82 0D   JSR $0D82
0DBC   38         SEC
0DBD   20 69 0D   JSR $0D69
0DC0   8A         TXA
0DC1   F0 BD      BEQ $0D80
0DC3   20 0F 0D   JSR $0D0F
0DC6   A5 DA      LDA $DA
0DC8   A2 08      LDX #$08
0DCA   A4 DB      LDY $DB
0DCC   D0 06      BNE $0DD4
0DCE   20 82 0D   JSR $0D82
0DD1   18         CLC
0DD2   90 0B      BCC $0DDF
0DD4   20 82 0D   JSR $0D82
0DD7   A5 DB      LDA $DB
0DD9   A2 03      LDX #$03   ; STEP value
0DDB   20 82 0D   JSR $0D82
0DDE   38         SEC
0DDF   20 69 0D   JSR $0D69
0DE2   A5 20      LDA $20
0DE4   38         SEC
0DE5   E9 03      SBC #$03
0DE7   C9 10      CMP #$10
0DE9   B0 08      BCS $0DF3
0DEB   A2 04      LDX #$04
0DED   20 82 0D   JSR $0D82
0DF0   18         CLC
0DF1   90 06      BCC $0DF9
0DF3   A2 08      LDX #$08
0DF5   20 82 0D   JSR $0D82
0DF8   38         SEC
0DF9   20 69 0D   JSR $0D69
0DFC   A9 01      LDA #$01
0DFE   D0 80      BNE $0D80

;
; Extraction code $0E00 - $0F2C
; Normally located at $0800 - $092C
;

; BASIC PRG ( 1991 SYS2061 )
0E00/0800   .BY $00,$0A,$08,$C7,$07,$9E,$32,$30
```

```
0E08/0808   .BY $36,$31,$00,$00,$00

0E0D/080D   A9 00      LDA #$00
0E0F/080F   20 44 E5   JSR $E544   ; clear screen
0E12/0812   78         SEI
0E13/0813   A2 FF      LDX #$FF
0E15/0815   9A         TXS   ; clear stack
0E16/0816   BD 37 08   LDA $0837,X
0E19/0819   9D F8 00   STA $00F8,X   ; relocate decompression code
0E1C/081C   CA         DEX
0E1D/081D   D0 F7      BNE $0E16/$0816
0E1F/081F   8E 11 D0   STX $D011   ; blank screen
0E22/0822   EE 30 D0   INC $D030   ; 2MHz mode on (if c128)
0E25/0825   86 01      STX $01
0E27/0827   A9 00      LDA #$00   ; replaced by pointer to
0E29/0829   A0 00      LDY #$00   ; end of program
0E2B/082B   85 2D      STA $2D
0E2D/082D   84 2E      STY $2E
0E2F/082F   85 AE      STA $AE
0E31/0831   84 AF      STY $AF
0E33/0833   A0 00      LDY #$00   ; Value is changed
0E35/0835   4C 00 01   JMP $0100

; Zero page bytes (are changed from zeros)
0E38/00F9   .BY $00,$00,$00,$00,$00,$00,$00

; Decompression code at $0100
0E3F/0100   BD 2C 09   LDA $092C,X
0E42/0103   9D E8 07   STA $07E8,X
0E45/0106   E8         INX
0E46/0107   D0 F7      BNE $0E3F/$0100
0E48/0109   EE 02 01   INC $0102
0E4B/010C   EE 05 01   INC $0105
0E4E/010F   88         DEY
0E4F/0110   D0 EE      BNE $0E3F/$0100
0E51/0112   A2 02      LDX #$02
0E53/0114   20 9D 01   JSR $019D
0E56/0117   F0 2C      BEQ $0E84/$0145
0E58/0119   C9 06      CMP #$06
0E5A/011B   90 12      BCC $0E6E/$012F
0E5C/011D   29 01      AND #$01
0E5E/011F   A8         TAY
0E5F/0120   20 9A 01   JSR $019A
0E62/0123   69 06      ADC #$06
0E64/0125   90 08      BCC $0E6E/$012F
0E66/0127   AA         TAX
0E67/0128   20 9D 01   JSR $019D
0E6A/012B   85 F9      STA $F9
0E6C/012D   10 E3      BPL $0E51/$0112
0E6E/012F   85 8B      STA $8B
```

```
0E70/0131   A5 FC      LDA $FC
0E72/0133   38         SEC
0E73/0134   E5 8B      SBC $8B
0E75/0136   85 FC      STA $FC
0E77/0138   85 8C      STA $8C
0E79/013A   A5 FD      LDA $FD
0E7B/013C   E9 00      SBC #$00
0E7D/013E   85 FD      STA $FD
0E7F/0140   85 8D      STA $8D
0E81/0142   20 86 01   JSR $0186
0E84/0145   A6 F9      LDX $F9
0E86/0147   F0 04      BEQ $0E8C/$014D
0E88/0149   C6 F9      DEC $F9
0E8A/014B   10 C5      BPL $0E51/$0112
0E8C/014D   20 9D 01   JSR $019D
0E8F/0150   F0 09      BEQ $0E9A/$015B
0E91/0152   20 9A 01   JSR $019A
0E94/0155   A2 02      LDX #$02
0E96/0157   86 8B      STX $8B
0E98/0159   90 1B      BCC $0EB5/$0176
0E9A/015B   E8         INX
0E9B/015C   20 9D 01   JSR $019D
0E9E/015F   F0 09      BEQ $0EA9/$016A
0EA0/0161   E8         INX
0EA1/0162   20 9D 01   JSR $019D
0EA4/0165   20 9A 01   JSR $019A
0EA7/0168   69 01      ADC #$01
0EA9/016A   69 03      ADC #$03
0EAB/016C   85 8B      STA $8B
0EAD/016E   E8         INX
0EAE/016F   20 9D 01   JSR $019D
0EB1/0172   C8         INY
0EB2/0173   20 9A 01   JSR $019A
0EB5/0176   65 FE      ADC $FE
0EB7/0178   85 8C      STA $8C
0EB9/017A   A5 8D      LDA $8D
0EBB/017C   65 FF      ADC $FF
0EBD/017E   85 8D      STA $8D
0EBF/0180   38         SEC
0EC0/0181   20 86 01   JSR $0186
0EC3/0184   F0 8C      BEQ $0E51/$0112   ; branch always
;
0EC5/0186   A4 8B      LDY $8B
0EC7/0188   A5 FE      LDA $FE
0EC9/018A   E5 8B      SBC $8B
0ECB/018C   85 FE      STA $FE
0ECD/018E   B0 02      BCS $0ED1/$0192
0ECF/0190   C6 FF      DEC $FF
0ED1/0192   B1 8C      LDA ($8C),Y
0ED3/0194   91 FE      STA ($FE),Y
0ED5/0196   88         DEY
```

```
0ED6/0197   D0 F9      BNE $0ED1/$0192
0ED8/0199   60         RTS

;
0ED9/019A   BE EA 01   LDX $01EA,Y
0EDC/019D   A9 00      LDA #$00
0EDE/019F   85 8D      STA $8D
0EE0/01A1   A4 FB      LDY $FB
0EE2/01A3   F0 0C      BEQ $0EF0/$01B1
0EE4/01A5   06 FA      ASL $FA
0EE6/01A7   2A         ROL
0EE7/01A8   26 8D      ROL $8D
0EE9/01AA   C6 FB      DEC $FB
0EEB/01AC   CA         DEX
0EEC/01AD   10 F2      BPL $0EE0/$01A1
0EEE/01AF   A8         TAY
0EEF/01B0   60         RTS

;
0EF0/01B1   85 8E      STA $8E
0EF2/01B3   B1 FC      LDA ($FC),Y
0EF4/01B5   85 FA      STA $FA
0EF6/01B7   A9 08      LDA #$08
0EF8/01B9   85 FB      STA $FB
0EFA/01BB   A5 8E      LDA $8E
0EFC/01BD   A4 FC      LDY $FC
0EFE/01BF   D0 02      BNE $0F02/$01C3
0F00/01C1   C6 FD      DEC $FD
0F02/01C3   C6 FC      DEC $FC
0F04/01C5   C0 E7      CPY #$E7
0F06/01C7   D0 DC      BNE $0EE4/$01A5
0F08/01C9   A4 FD      LDY $FD
0F0A/01CB   C0 07      CPY #$07
0F0C/01CD   D0 D6      BNE $0EE4/$01A5
0F0E/01CF   A9 33      LDA #$33   ; set value of $01 register
0F10/01D1   85 01      STA $01
0F12/01D3   CE 30 D0   DEC $D030   ; 2MHz mode off
0F15/01D6   A9 1B      LDA #$1B
0F17/01D8   8D 11 D0   STA $D011   ; unblank screen
0F1A/01DB   58         CLI
0F1B/01DC   A9 00      LDA #$00
0F1D/01DE   8D 00 08   STA $0800
0F20/01E1   20 33 33   JSR $3333   ; JSR to start address of file
0F23/01E4   20 8E A6   JSR $A68E
0F26/01E7   4C AE A7   JMP $A7AE
0F29/01EA  .BY $03
0F2A/01EB  .BY $07
0F2B/01EC  .BY $0A   ; STEP plus 7
0F2C/01ED  .BY $00
```

```
; The end
```