```
       1/       0 :                             cpu Z8601
       2/       0 :                             page 0
       3/       0 :                             include stddefZ8.inc
(1)    1/       0 :                                     save
(1)   55/       0 :                                     restore
(1)   56/       0 :
(1)   57/       0 :
       4/       0 :
       5/       0 :                     ;*****************************************************************
       6/       0 :                     ;
       7/       0 :                     ;              Apple Widget Voicecoil Servo - Z8 firmware
       8/       0 :                     ;                           Version 341-0260-A
       9/       0 :                     ;            original code (c) by Apple Computer Inc., 1983
      10/       0 :                     ;
      11/       0 :                     ;
      12/       0 :                     ;        disassembled, commented and converted into as source
      13/       0 :                     ;                        by Patrick Schaefer, 2012
      14/       0 :                     ;
      15/       0 :                     ;    use as build 2011-08-02 or later. Last changes: 2013-07-31 PS
      16/       0 :                     ;
      17/       0 :                     ;*****************************************************************
      18/       0 :
      19/       0 : =800H               ROMsize         SET 2048
      20/       0 :
      21/       0 :
      22/       0 :                     ; ***************************************************************
      23/       0 :                     ;  Hardware
      24/       0 :                     ; ***************************************************************
      25/       0 :                     ;
      26/       0 :                     ; *** Port 0
      27/       0 : =1H                 ParkHeads       EQU 001h        ; P0.0 1 to disable power amp and park heads
      28/       0 : =2H                 Mux1D0          EQU 002h        ; P0.1 HA-2405 (U3E) D0
      29/       0 : =4H                 Mux1D1          EQU 004h        ; P0.2 HA-2405 (U3E) D1
      30/       0 : =8H                 Mux1Ena         EQU 008h        ; P0.3 HA-2405 (U3E) Enable
      31/       0 : =10H                OffsetDacStrb   EQU 010h        ; P0.4 L291 strobe 0 = active, 1 = 0V
      32/       0 : =20H                RecalMode       EQU 020h        ; P0.5 1 = select recal mode (velocity control)
      33/       0 : =40H                SettlingMode    EQU 040h        ; P0.6 1 = select settling mode
      34/       0 : =80H                AccessMode      EQU 080h        ; P0.7 1 = select access mode
      35/       0 :                     ; *** Port 1
      36/       0 :                     ;  bit 0..4 offset magnitude (1Fh = 0; 1Eh = 1/16 Iref; ...; 00h = 31/16 Iref)
      37/       0 : =1FH                OffsetDacZero   EQU 01Fh
      38/       0 : =20H                OffsetDacSign   EQU 020h        ; bit 5 offset sign (0=minus, 1=plus)
      39/       0 : =40H                Mux2D0          EQU 040h        ; bit 6 HA-2405 (U5D) D0
      40/       0 : =80H                Mux2D1          EQU 080h        ; bit 7 HA-2405 (U5D) D1
      41/       0 :                     ; *** Port 2
      42/       0 : =1H                 U2Gfast         EQU 001h        ; P2.0 1 = OpAmp U2G fast, 0 = slow
      43/       0 : =2H                 Port21          EQU 002h        ; P2.1 in  On Track Window ??
      44/       0 : =4H                 PosErr_Cmp      EQU 004h        ; P2.2 in  1 when Position Error > DAC value
      45/       0 : =8H                 Port23          EQU 008h        ; P2.3 0 = zero integrator U1B,
      46/       0 :                                                     ;      1 integrator zeroed by Pulse A
      47/       0 : =10H                Port24          EQU 010h        ; P2.4 in  Final Track Window ??
      48/       0 : =20H                U2Hfast         EQU 020h        ; P2.5 1 = OpAmp U2H fast, 0 = slow
      49/       0 : =40H                Mux3D0          EQU 040h        ; P2.6 HA-2405 (U3C) D0
      50/       0 : =80H                OddEven         EQU 080h        ; P2.7 1 = Odd, 0 = Even
      51/       0 :                     ; *** Port 3
      52/       0 : =1H                 Servo_SI        EQU 001h        ; P3.0 SIO RxD in
      53/       0 : =2H                 Port31          EQU 002h
      54/       0 : =4H                 Port32          EQU 004h
      55/       0 : =8H                 Port33          EQU 008h
      56/       0 :                     ; Mode      | P3.1                      P3.2                    P3.3
      57/       0 :                     ;           | (T1 clock or INT2)        (INT0)                  (INT1)
      58/       0 :                     ; ----------+-------------------------------------------------------------+
      59/       0 :                     ; idle      | ??                        ??                      ??
      60/       0 :                     ;           |
      61/       0 :                     ; Recal     | Pulse A NAND Pulse B      ??                      ??
      62/       0 :                     ;  (P0.5=1) |
      63/       0 :                     ; Settling  | U1E output                ??                      Position Error Sample
      64/       0 :                     ;  (P0.6=1) |
      65/       0 :                     ; Access    | Pulse A XOR Pulse B       ??                      ??
      66/       0 :                     ;  (P0.7=1) |
      67/       0 :                     ;
      68/       0 : =10H                SioRdy          EQU 010h        ; P3.4 SIO Ready
      69/       0 : =20H                SrvoRdy         EQU 020h        ; P3.5 Servo Ready
      70/       0 : =40H                SrvoErr         EQU 040h        ; P3.6 Servo Error
      71/       0 : =80H                Servo_SO        EQU 080h        ; P3.7 SIO TxD out
      72/       0 :
      73/       0 :
      74/       0 :                     ; ***************************************************************
      75/       0 :                     ;  internal RAM
      76/       0 :                     ; ***************************************************************
      77/       0 :                     ;
      78/       0 : =4H                 CmdByte0        EQU 04h         ; bytes received from host
      79/       0 : =5H                 CmdByte1        EQU 05h
      80/       0 : =6H                 CmdByte2        EQU 06h
      81/       0 : =7H                 CmdByte3        EQU 07h
      82/       0 : =8H                 CmdByte4        EQU 08h
      83/       0 : =9H                 BytePtr         EQU 09h         ; byte pointer for serial comms
      84/       0 :                     ;               EQU 0Ah
      85/       0 :                     ;               EQU 0Bh
      86/       0 :                     ;               EQU 0Ch
      87/       0 :                     ;               EQU 0Dh
      88/       0 :                     ;               EQU 0Eh
      89/       0 :                     ;               EQU 0Fh
      90/       0 :                     ;               EQU 10h
      91/       0 : =11H                SM_State        EQU 11h         ; state machine
      92/       0 : =12H                T0_MSB          EQU 12h         ; MSB for access timeout (T0)
      93/       0 : =13H                Temp1           EQU 13h
      94/       0 : =14H                Temp2           EQU 14h
      95/       0 : =15H                Temp3           EQU 15h
      96/       0 : =16H                P1_Mask         EQU 16h         ; bit 6 (040h) set during init,
      97/       0 :                                                     ; bit 7 (080h) set when 57k6
      98/       0 : =17H                StatusByte0     EQU 17h         ; bytes sent to host
      99/       0 : =18H                StatusByte1     EQU 18h
     100/       0 : =19H                StatusByte2     EQU 19h
     101/       0 : =1AH                StatusByte3     EQU 1Ah
     102/       0 : =1BH                SM_FaultState   EQU 1Bh         ; state where failure occured
     103/       0 : =1CH                DelayL          EQU 1Ch         ; delay time value
     104/       0 : =1DH                DelayH          EQU 1Dh
     105/       0 : =1EH                CmdByte0H       EQU 1Eh         ; upper 4 bit of CmdByte0 (00..0Fh)
     106/       0 : =1FH                FaultCtr        EQU 1Fh         ; fault counter
     107/       0 : =20H                SioBaud         EQU 20h         ; 3 for 19k2, 1 for 57k6
     108/       0 : =21H                LastRCmd0       EQU 21h         ; last received command
     109/       0 : =22H                LastRCmd1       EQU 22h
     110/       0 : =23H                LastRCmd2       EQU 23h
     111/       0 : =24H                LastRCmd3       EQU 24h
```

```
112/       0 : =25H                LastPCmd0       EQU 25h         ; last processed command
113/       0 : =26H                LastPCmd1       EQU 26h
114/       0 : =27H                LastPCmd2       EQU 27h
115/       0 : =28H                LastPCmd3       EQU 28h
116/       0 : =29H                LastPCmdAbort   EQU 29h         ; 1 when last command has been aborted
117/       0 : =2AH                OffsetFlag      EQU 2Ah         ; bit 7 set when offset control on (DAC <> 0x1Fh)
118/       0 : =2BH                SAR_ANDmask     EQU 2Bh         ; for Auto Offset approximation
119/       0 : =2CH                SAR_ORmask      EQU 2Ch
120/       0 :                     ; 2D:2Fh unused
121/       0 :                     ; 30:3Fh used during Access
122/       0 :
123/       0 :
124/       0 :
125/       0 : =4H                 AccForward      EQU 004h        ; CmdByte0 bit 2: 1 = forward (to spindle), 0 = back
126/       0 : =80H                OfsForward      EQU 080h        ; CmdByte2 bit 7: 1 = forward (to spindle), 0 = back
127/       0 : =40H                OfsSetAuto      EQU 040h        ; CmdByte2 bit 6: 1 = auto offset enable
128/       0 : =20H                OfsReadDAC      EQU 020h        ; CmdByte2 bit 5: 1 = take offset value from DAC
129/       0 : =80H                Sta57k6         EQU 080h        ; CmdByte3 bit 7: 1 = set comm rate to 57k6
130/       0 :
131/       0 :
132/       0 :                     ; *****************************************************************
133/       0 :
134/       0 :
135/       0 :                             org 0000h
136/       0 : 07 5F               Int_Vec0        DW HW_Interrupt         ; P3.2
137/       2 : 07 5F               Int_Vec1        DW HW_Interrupt         ; P3.3
138/       4 : 07 5F               Int_Vec2        DW HW_Interrupt         ; P3.1
139/       6 : 07 5F               Int_Vec3        DW HW_Interrupt         ; P3.0 and SIO in
140/       8 : 07 6B               Int_Vec4        DW T0_Interrupt         ; T0 and SIO out
141/       A : 07 8F               Int_Vec5        DW T1_Interrupt         ; T1
142/       C :
143/       C :                     ; *** State 0 System Initialization
144/       C : E6 F7 01            Start           ld P3M,#001h            ; totem-pole I/O, parity off
145/       F : B0 03                               clr P3
146/      11 : E6 F8 04                            ld P01M,#004h           ; P0, P1 output, internal stack
147/      14 : E6 FF 80                            ld SPL,#080h
148/      17 : 8F                                  di
149/      18 : B0 FB                               clr IMR
150/      1A : B0 F1                               clr TMR
151/      1C : 31 00                               srp #000h
152/      1E :                            assume RP:000h
153/      1E : B0 E4                               clr R4                  ; clear internal RAM
154/      20 : 5C 7A                               ld R5,#122
155/      22 : D7 45 05           ZeroRAM_Lp       ld 05h(R5),R4
156/      25 : 5A FB                               djnz R5,ZeroRAM_Lp
157/      27 : E6 01 1F                            ld >P1,#OffsetDacZero   ; set offset DAC to zero
158/      2A : E6 F6 16                            ld P2M,#016h            ; set P2.4, P2.2, P2.1 as input
159/      2D : B0 02                               clr P2
160/      2F : E6 16 40                            ld P1_Mask,#040h
161/      32 :                    ; *** State 1 SIO Communication Setup
162/      32 : E6 11 01                            ld SM_State,#1
163/      35 : 9F                                  ei
164/      36 : E6 20 03                            ld SioBaud,#3           ; default baud rate = 19k2
165/      39 :
166/      39 :                    ; command C: Home (park heads)
167/      39 :                    ;
168/      39 : E6 00 11           Cmd_Home         ld >P0,#OffsetDacStrb+ParkHeads ; park and wait
169/      3C : 56 03 DF                            and >P3,#0FFh-SrvoRdy   ; clear Servo Ready
170/      3F : D6 05 B2                            call RxCmdBytes         ; get host command
171/      42 : D6 03 E0                            call Cmd_Decode         ; get cmd vector into RR4
172/      45 : A6 1E 00                            cp CmdByte0H,#0         ;  ReadStatus?
173/      48 : 6B 02                               jr Z, Cmd_Home1
174/      4A : 30 E4                               jp @RR4                 ; otherwise jump to vector
175/      4C :                    ; send status command
176/      4C : B0 FE              Cmd_Home1        clr SPH
177/      4E : D4 E4                               call @RR4               ; call Cmd_ReadStatus
178/      50 : 76 FE 80                            tm SPH,#Sta57k6         ; TxStatusBytes stored CmdByte3 here
179/      53 : 6B 06                               jr Z, Cmd_Home2
180/      55 : E6 20 01                            ld SioBaud,#1           ; go to 57k6
181/      58 : 46 16 80                            or P1_Mask,#080h
182/      5B : 8B DC              Cmd_Home2        jr Cmd_Home
183/      5D :
184/      5D :                    ; dead code (never accessed)
185/      5D : A6 11 0A                            cp SM_State,#10         ; error state?
186/      60 : 6D 03 7C                            jp Z, SM_State_10
187/      63 : B0 F1                               clr TMR
188/      65 :
189/      65 :
190/      65 :                    ; *****************************************************************
191/      65 :                    ; command 4/7: Recal
192/      65 :                    ;
193/      65 : E6 00 11           Cmd_Recal        ld >P0,#OffsetDacStrb+ParkHeads ; park and wait
194/      68 : 56 03 80                            and >P3,#0F0h-SrvoErr-SrvoRdy-SioRdy    ; clear Err and Rdy flags
195/      6B : E6 15 06                            ld Temp3,#6             ; ??
196/      6E : E6 1C FF                            ld DelayL,#255
197/      71 : E6 1D FF                            ld DelayH,#255
198/      74 : D6 06 75                            call Delay
199/      77 : D6 06 75                            call Delay
200/      7A : B0 1F                               clr FaultCtr            ; first attempt
201/      7C :                    ; *** State 2 Recal State
202/      7C : E6 11 02           SM_State_2       ld SM_State,#2
203/      7F : 56 03 CF                            and >P3,#0FFh-SioRdy-SrvoRdy    ; clear SioRdy, SrvoRdy
204/      82 : 8F                                  di
205/      83 : B0 FA                               clr IRQ
206/      85 : E6 F4 FF                            ld T0,#0FFh             ; load access timer T0
207/      88 : E6 F5 FF                            ld PRE0,#0FFh           ;  Timer 0 int/63 continuous
208/      8B : E6 12 0A                            ld T0_MSB,#10
209/      8E : A6 1F 00                            cp FaultCtr,#0          ; is this a retry?
210/      91 : EB 13                               jr NZ, SM_St2_1         ;  yes --> (we already have T1 set up)
211/      93 : 76 04 30                            tm >CmdByte0,#030h      ; normal recal?
212/      96 : EB 08                               jr NZ, SM_St2_2         ;  yes -->
213/      98 : E6 F2 12                            ld T1,#72/4             ; load pulse count for Format Recal
214/      9B : E6 F3 04                            ld PRE1,#004h           ;  Timer 1 ext/1 single (count P3.1)
215/      9E : 8B 06                               jr SM_St2_1
216/      A0 : E6 F2 08           SM_St2_2         ld T1,#32/4             ; load pulse count for Data Recal
217/      A3 : E6 F3 04                            ld PRE1,#004h           ;  Timer 1 ext/1 single (count P3.1)
218/      A6 : B0 04              SM_St2_1         clr CmdByte0            ; set ports for recal motion
219/      A8 : E6 01 DA                            ld >P1,#Mux2D1+Mux2D0+01Ah      ; DAC value 5 (01Ah)
220/      AB : E6 00 38                            ld >P0,#RecalMode+OffsetDacStrb+Mux1Ena
221/      AE : E6 0E C9                            ld >0Eh,#0C9h           ; 01101010b
222/      B1 :                    ; *** State 3 Start Recal Motion
223/      B1 :                    ;  Wait for the number of PulseA pulses given in T1. Each pulse equals to four
224/      B1 :                    ;  tracks on disk.
225/      B1 : E6 11 03                            ld >SM_State,#3
226/      B4 : E6 F1 0F                            ld TMR,#00Fh            ; load & enable T0 & T1
227/      B7 : E6 FB 10                            ld IMR,#010h            ; enable T0 interrupt (access timeout)
228/      BA : 9F                                  ei
229/      BB : B0 FA                               clr IRQ                 ;  and clear pending interrupts
```

```
230/      BD : A6 11 0A           SM_St3_Lp       cp SM_State,#10         ; error state?
231/      C0 : 6D 03 7C                           jp Z, SM_State_10
232/      C3 : 66 FA 04                           tcm IRQ,#004h           ; pulse at P3.1?
233/      C6 : EB F5                              jr NZ, SM_St3_Lp        ;  no --> wait
234/      C8 : 56 FA FB                           and IRQ,#0FBh           ; acknowledge pulse
235/      CB : 66 FA 20                           tcm IRQ,#020h           ; T1 expired?
236/      CE : EB ED                              jr NZ, SM_St3_Lp        ;  no --> wait
237/      D0 : 56 FA DA                           and IRQ,#0DAh           ; acknowledge ints and set ports for
238/      D3 : E6 00 0C                           ld >P0,#Mux1Ena+Mux1D1  ;    final approach
239/      D6 : 56 02 07                           and >P2,#0FFh-OddEven-Mux3D0-U2Hfast-Port24-Port23
240/      D9 :                    ; *** State 4 Recal Final Approach
241/      D9 : E6 11 04                           ld >SM_State,#4
242/      DC : E6 01 DF                           ld >P1,#0FFh-OffsetDacSign
243/      DF : 8F                                 di
244/      E0 : E6 FB 10                           ld IMR,#010h            ; enable T0 interrupt (access timeout)
245/      E3 : 9F                                 ei
246/      E4 : A6 11 0A           SM_St4_Lp       cp SM_State,#10         ; error state?
247/      E7 : 6D 03 7C                           jp Z, SM_State_10
248/      EA : 66 FA 01                           tcm IRQ,#001h           ; pulse at P3.2 (on track)?
249/      ED : EB F5                              jr NZ, SM_St4_Lp        ;  no --> wait
250/      EF : E6 1C 01                           ld DelayL,#1
251/      F2 : E6 1D 01                           ld DelayH,#1
252/      F5 : D6 06 75                           call Delay
253/      F8 : 56 FA DE                           and IRQ,#0DEh           ; acknowledge pulse
254/      FB : 46 00 16                           or >P0,#OffsetDacStrb+Mux1D1+Mux1D0
255/      FE : E6 1C 0F                           ld DelayL,#15
256/     101 : E6 1D 01                           ld DelayH,#1
257/     104 : D6 06 75                           call Delay
258/     107 :                    ; *** State 5 Settling Control
259/     107 : E6 11 05           SM_State_5      ld SM_State,#5
260/     10A : 56 02 BF                           and >P2,#0FFh-Mux3D0    ; set ports for position settling
261/     10D : E6 00 5A                           ld >P0,#SettlingMode+OffsetDacStrb+Mux1Ena+Mux1D0
262/     110 : 8F                                 di
263/     111 : B0 FB                              clr IMR
264/     113 : B0 F1                              clr TMR
265/     115 : 56 FA 0F                           and IRQ,#00Fh           ; allow only P3.0 - P3.3 interrupts
266/     118 : E6 F4 FF                           ld T0,#0FFh             ; load arm setting timer
267/     11B : E6 F5 1C                           ld PRE0,#01Ch           ;  Timer 0 int/7 single
268/     11E : E6 F2 03                           ld T1,#003h             ; load track crossing counter
269/     121 : E6 F3 04                           ld PRE1,#004h           ;  Timer 1 ext/1 single
270/     124 : E6 F1 0F                           ld TMR,#00Fh            ; load & enable T0 & T1
271/     127 : 9F                                 ei
272/     128 : 56 04 3F                           and >CmdByte0,#03Fh
273/     12B : E6 FE 02                           ld SPH,#2               ; retry counter
274/     12E : A6 11 0A           SM_St5_Lp       cp SM_State,#10         ; error state?
275/     131 : 6D 03 7C                           jp Z, SM_State_10
276/     134 : 76 FA 10                           tm IRQ,#010h            ; T0 expired?
277/     137 : 6B F5                              jr Z, SM_St5_Lp         ;  no, wait
278/     139 : 56 FA EF                           and IRQ,#0EFh           ; acknowledge T0 interrupt
279/     13C : 56 F1 0C                           and TMR,#00Ch           ; and stop T0
280/     13F : 76 03 02                           tm >P3,#Port31          ; P3.1 zero (not on track)?
281/     142 : 6B 0A                              jr Z, SM_State_6        ;  no --> we have landed
282/     144 :                    ;
283/     144 : 00 FE                              dec SPH                 ; otherwise try again
284/     146 : 6D 03 6A                           jp Z, State5_Error      ;  until counter expired
285/     149 : 46 F1 03                           or TMR,#003h            ; load & start T0
286/     14C : 8B E0                              jr SM_St5_Lp
287/     14E :
288/     14E :                    ; *** State 6
289/     14E : E6 11 06           SM_State_6      ld SM_State,#6
290/     151 : E6 1C FF                           ld DelayL,#255
291/     154 : E6 1D 04                           ld DelayH,#4
292/     157 : D6 06 75                           call Delay
293/     15A : 46 02 40                           or >P2,#Mux3D0
294/     15D : A6 1F 00                           cp >FaultCtr,#0         ; is this a retry?
295/     160 : EB 03                              jr NZ, SM_St6_1         ;  yes -->
296/     162 : 56 03 BF                           and >P3,#0FFh-SrvoErr   ; clear SrvoErr
297/     165 : E6 1C FF           SM_St6_1        ld DelayL,#255
298/     168 : E6 1D 0A                           ld DelayH,#10
299/     16B : D6 06 75                           call Delay
300/     16E : 8F                                 di
301/     16F : 56 FA DA                           and IRQ,#0DAh           ; acknowledge P3.1 & P3.2 int
302/     172 : E6 FB 20                           ld IMR,#020h            ; enable T1 interrupt
303/     175 : E6 F1 0C                           ld TMR,#00Ch            ; load & start T1
304/     178 : 9F                                 ei
305/     179 : A6 2A 80                           cp OffsetFlag,#080h     ; offset control active?
306/     17C : EB 10                              jr NZ, SM_St6_2         ;  no -->
307/     17E : B0 2A                              clr OffsetFlag          ; otherwise turn it off
308/     180 : 50 E5                              pop R5
309/     182 : 50 E4                              pop R4
310/     184 : 50 07                              pop CmdByte3
311/     186 : 50 06                              pop CmdByte2
312/     188 : 50 05                              pop CmdByte1
313/     18A : 50 04                              pop CmdByte0
314/     18C : 8B 2A                              jr SM_St6_3
315/     18E :
316/     18E : 76 04 10           SM_St6_2        tm >CmdByte0,#010h      ; access with offset?
317/     191 : 6B 06                              jr Z, SM_State_6a       ;  no -->
318/     193 :
319/     193 :
320/     193 :                    ; command 1: SetOffset
321/     193 :                    ;
322/     193 : D6 04 EA           Cmd_SetOffset   call CallOffset
323/     196 : 8D 01 07                           jp SM_State_5
324/     199 :
325/     199 :
326/     199 :                    ; *** State 6a
327/     199 : E6 11 06           SM_State_6a     ld SM_State,#6
328/     19C : 46 03 20                           or >P3,#SrvoRdy         ; set ServoRdy
329/     19F : D6 05 B2                           call RxCmdBytes         ;  and wait for host command
330/     1A2 : A6 11 0A                           cp SM_State,#10         ; error state?
331/     1A5 : 6D 03 7C                           jp Z, SM_State_10
332/     1A8 :                    ; *** State 7 Start SIO Communication
333/     1A8 : E6 11 07                           ld SM_State,#7
334/     1AB : D6 03 E0                           call Cmd_Decode         ; get Cmd vector into RR4
335/     1AE : A6 1E 00                           cp CmdByte0H,#0         ; ReadStatus command?
336/     1B1 : 6B 07                              jr Z, SM_St7_1          ;  yes -->
337/     1B3 : 76 04 80                           tm >CmdByte0,#080h      ; Access command
338/     1B6 : EB 06                              jr NZ, SM_St7_2         ;  yes (bit 7 was 1) -->
339/     1B8 : 30 E4              SM_St6_3        jp @RR4                 ; execute cmd and continue with State 2
340/     1BA :                    ; ReadStatus command
341/     1BA : D4 E4              SM_St7_1        call @RR4               ; execute ReadStatus and
342/     1BC : 8B DB                              jr SM_State_6a          ;  continue with State 6
343/     1BE :                    ; Access command
344/     1BE : 66 01 1F           SM_St7_2        tcm >P1,#OffsetDacZero  ; DAC at zero?
345/     1C1 : 6B F5                              jr Z, SM_St6_3          ; yes, so we don't have an offset
346/     1C3 : E6 2A 80                           ld OffsetFlag,#080h     ;
347/     1C6 : 70 04                              push CmdByte0
```

```
348/     1C8 : 70 05                              push CmdByte1
349/     1CA : 70 06                              push CmdByte2
350/     1CC : 70 07                              push CmdByte3
351/     1CE : 70 E4                              push R4
352/     1D0 : 70 E5                              push R5
353/     1D2 : B0 06                              clr CmdByte2
354/     1D4 : D6 04 EA                           call CallOffset
355/     1D7 : 8D 01 07                           jp SM_State_5
356/     1DA :
357/     1DA :
358/     1DA :                    ; *****************************************************************
359/     1DA :                    ; command 8/9: Access / AccessOffset
360/     1DA :                    ;
361/     1DA :                    ; *** State 8 Access State (Seek)
362/     1DA : E6 11 08           Cmd_Access      ld SM_State,#8
363/     1DD : 56 03 DF                           and >P3,#0FFh-SrvoRdy   ; clear SrvoRdy
364/     1E0 : 8F                                 di
365/     1E1 : 56 FA 27                           and IRQ,#027h
366/     1E4 : B0 F1                              clr TMR
367/     1E6 : 76 04 04                           tm >CmdByte0,#AccForward        ; access direction forward?
368/     1E9 : EB 07                              jr NZ, SM_St8_1                 ;  yes (bit 3 was 1) -->
369/     1EB : 56 01 DF                           and >P1,#0FFh-OffsetDacSign     ; negative
370/     1EE : B0 16                              clr P1_Mask
371/     1F0 : 8B 06                              jr SM_St8_12
372/     1F2 : 46 01 20           SM_St8_1        or >P1,#OffsetDacSign           ; positive
373/     1F5 : E6 16 20                           ld P1_Mask,#OffsetDacSign
374/     1F8 : E4 05 F2           SM_St8_12       ld T1,>CmdByte1         ; load difference counter T1
375/     1FB : 46 F1 04                           or TMR,#004h            ;   with magnitude LSB
376/     1FE : E4 04 13                           ld Temp1,>CmdByte0      ; magnitude MSB
377/     201 : 56 13 03                           and Temp1,#003h         ; mask out command bits
378/     204 : 31 30                              srp #030h
379/     206 :                                    assume RP:030h
380/     206 : 38 05                              ld R3,CmdByte1          ; set difference LSB
381/     208 : 28 13                              ld R2,Temp1             ; and MSB
382/     20A : 76 E2 03                           tm R2,#003h             ; MSB zero?
383/     20D : EB 06                              jr NZ, SM_St8_2         ;  no -->
384/     20F : A6 E3 00                           cp R3,#0                ; LSB zero?
385/     212 : 6D 01 4E                           jp Z, SM_State_6        ; yes --> go settling (State 6)
386/     215 :                    ; set velocity curve value
387/     215 : 76 E2 03           SM_St8_2        tm R2,#003h             ; MSB zero?
388/     218 : EB 0D                              jr NZ, SM_St8_3         ;  no -->
389/     21A : 76 E3 E0                           tm R3,#0E0h             ; LSB < 63 (1Fh)?
390/     21D : EB 08                              jr NZ, SM_St8_3         ;  no -->
391/     21F : D6 07 29                           call GetCurveValue      ; get DAC value
392/     222 : E4 16 01                           ld P1,P1_Mask           ;  and update P1
393/     225 : 8B 03                              jr SM_St8_4
394/     227 : 56 01 E0           SM_St8_3        and P1,#0FFh-OffsetDacZero      ; set DAC at maximum
395/     22A : E6 F3 04           SM_St8_4        ld PRE1,#004h           ;  Timer 1 ext/1 single
396/     22D : E6 F4 FF                           ld T0,#0FFh
397/     230 : E6 F5 FF                           ld PRE0,#0FFh           ;  Timer 0 int/63 continuous
398/     233 : E6 12 0A                           ld T0_MSB,#10
399/     236 : E6 00 98                           ld P0,#098h
400/     239 : B0 ED                              clr R13                 ; use curve table
401/     23B : D6 06 84                           call L684
402/     23E : 00 F2                              dec T1
403/     240 : 80 E2                              decw RR2                ; difference -1
404/     242 : 46 02 01                           or P2,#U2Gfast          ; set P2.0 (U2G fast)
405/     245 : 56 FA FD                           and IRQ,#0FDh           ; acknowledge P3.1 interrupt
406/     248 : E6 FB 12                           ld IMR,#012h            ; enable T0 and P3.3 int
407/     24B : 46 F1 07                           or TMR,#007h            ; load T1, load & start T0
408/     24E : 76 E2 03                           tm R2,#003h             ; MSB zero?
409/     251 : EB 2E                              jr NZ, SM_St8_5         ;  no -->
410/     253 : 76 E3 E0                           tm R3,#0E0h             ; LSB < 63 (1Fh)?
411/     256 : EB 29                              jr NZ, SM_St8_5         ;  no -->
412/     258 : 76 E3 FF                           tm R3,#0FFh             ; LSB zero
413/     25B : 6B 0B                              jr Z, SM_St8_6          ;  yes -->
414/     25D : 46 02 08                           or P2,#Port23           ; set P2.3 (auto integrator control)
415/     260 : D6 07 29                           call GetCurveValue
416/     263 : E4 16 01                           ld P1,P1_Mask
417/     266 : 8B 13                              jr SM_St8_7
418/     268 :                    ;
419/     268 : 46 FA 20           SM_St8_6        or IRQ,#020h            ; force T1 interrupt
420/     26B : 46 02 08                           or P2,#Port23           ; set P2.3 (auto integrator control)
421/     26E : E6 1C AF                           ld DelayL,#175
422/     271 : E6 1D 01                           ld DelayH,#1
423/     274 : D6 06 75                           call Delay
424/     277 : DC 01                              ld R13,#1               ; use fixed DAC value
425/     279 : 8B 1E                              jr SM_St8_8
426/     27B :                    ;
427/     27B : D6 07 29           SM_St8_7        call GetCurveValue
428/     27E : E4 16 01                           ld P1,P1_Mask
429/     281 : 80 E2              SM_St8_5        decw RR2
430/     283 : 76 E2 03                           tm R2,#003h             ; MSB zero?
431/     286 : EB 08                              jr NZ, SM_St8_9         ;  no -->
432/     288 : 76 E3 E0                           tm R3,#0E0h             ; LSB < 63 (1Fh)?
433/     28B : EB 03                              jr NZ, SM_St8_9         ;  no -->
434/     28D : D6 07 29                           call GetCurveValue
435/     290 : E6 1C 50           SM_St8_9        ld DelayL,#80
436/     293 : E6 1D 03                           ld DelayH,#3
437/     296 : D6 06 75                           call Delay
438/     299 : 46 F1 0C           SM_St8_8        or TMR,#00Ch            ; load & start T1
439/     29C : 56 FA FB                           and IRQ,#0FBh           ; clear pending P3.1 interrupt
440/     29F : 9F                                 ei
441/     2A0 : 66 FA 20           SM_St8_Lp2      tcm IRQ,#020h           ; T1 int occurred?
442/     2A3 : 6B 31                              jr Z, SM_St8_10         ;  yes -->
443/     2A5 : A6 11 0A           SM_St8_Lp1      cp SM_State,#10         ; error state?
444/     2A8 : 6D 03 7C                           jp Z, SM_State_10
445/     2AB : 76 FA 04                           tm IRQ,#004h
446/     2AE : 6B F5                              jr Z, SM_St8_Lp1        ; wait for P3.1 interrupt
447/     2B0 : 80 E2                              decw RR2
448/     2B2 : 56 FA FB                           and IRQ,#0FBh           ; clear P3.1 interrupt
449/     2B5 : 76 FA 20                           tm IRQ,#020h            ; T1 int occurred?
450/     2B8 : EB 1C                              jr NZ, SM_St8_10        ;  yes -->
451/     2BA : 76 E2 03                           tm R2,#003h             ; MSB zero?
452/     2BD : EB 2F                              jr NZ, SM_St8_11        ;  no -->
453/     2BF : A6 E2 FF                           cp R2,#0FFh             ; LSB = 255?
454/     2C2 : 6B 31                              jr Z, SM_State_9        ;  yes -->
455/     2C4 : A6 F2 1F                           cp T1,#01Fh
456/     2C7 : BB 25                              jr ugt, SM_St8_11
457/     2C9 : E4 16 01                           ld P1,P1_Mask
458/     2CC : 46 02 08                           or P2,#Port23           ; set P2.3 (auto integrator control)
459/     2CF : D6 07 29                           call GetCurveValue
460/     2D2 : 31 30                              srp #030h
461/     2D4 :                                    assume RP:030h
462/     2D4 : 8B CA                              jr SM_St8_Lp2
463/     2D6 :                    ;
464/     2D6 : A6 E2 01           SM_St8_10       cp R2,#1
465/     2D9 : BB 1A                              jr ugt, SM_State_9
```

```
466/     2DB : A6 E2 02                         cp R2,#2
467/     2DE : BB 15                            jr ugt, SM_State_9
468/     2E0 : A6 ED 01                         cp R13,#1               ; fixed DAC value?
469/     2E3 : 6B 10                            jr Z, SM_State_9        ;  yes -->
470/     2E5 : 56 FA DF                         and IRQ,#0DFh
471/     2E8 : E6 F2 00                         ld T1,#0
472/     2EB : 46 F1 0C                         or TMR,#00Ch            ; load & start T1
473/     2EE : 56 01 E0             SM_St8_11   and P1,#0FFh-OffsetDacZero      ; DAC at maximum
474/     2F1 : 31 30                            srp #030h
475/     2F3 :                                  assume RP:030h
476/     2F3 : 8B AB                            jr SM_St8_Lp2
477/     2F5 :
478/     2F5 :                      ; *** State 9 Seek Final Approach (one track to go)
479/     2F5 : E6 11 09             SM_State_9  ld SM_State,#9
480/     2F8 : 56 02 FF                         and P2,#0FFh            ; ??
481/     2FB : 46 01 1F                         or P1,#OffsetDacZero
482/     2FE : 56 01 FB                         and P1,#0FFh-4          ; set offset DAC to 4
483/     301 : A6 ED 01                         cp R13,#1               ; fixed DAC value?
484/     304 : EB 03                            jr NZ, SM_St9_1         ;  no -->
485/     306 : 56 01 F9                         and P1,#0FFh-6          ; set offset DAC to 6
486/     309 : 56 01 3F             SM_St9_1    and P1,#0FFh-Mux2D0-Mux2D1      ; select Mux2 channel 0
487/     30C : 44 EF 01                         or P1,R15
488/     30F : E6 00 18                         ld P0,#OffsetDacStrb+Mux1Ena    ; select Mux1 channel 0
489/     312 : E6 1C 01                         ld DelayL,#1
490/     315 : E6 1D 01                         ld DelayH,#1
491/     318 : D6 06 75                         call Delay
492/     31B : 56 F1 03                         and TMR,#003h           ; load and start T0
493/     31E : 6B 00                            jr Z, SM_St9_2
494/     320 : A6 11 0A             SM_St9_2    cp SM_State,#10         ; error state?
495/     323 : 6B 57                            jr Z, SM_State_10
496/     325 : 76 02 10                         tm P2,#Port24           ; Final Track Window??
497/     328 : 6B F6                            jr Z, SM_St9_2
498/     32A : 56 02 F7                         and P2,#0FFh-Port23     ; clear P2.3 (zero integrator)
499/     32D : 46 01 1F                         or P1,#OffsetDacZero    ; set offset DAC to zero
500/     330 : 56 FA FE                         and IRQ,#0FEh           ; acknowledge P3.2 interrupt
501/     333 : E6 00 08                         ld P0,#Mux1Ena          ; select Mux1 channel 0
502/     336 : 8F                               di
503/     337 : E6 FB 10                         ld IMR,#010h            ; enable T0 interrupt
504/     33A : 9F                               ei
505/     33B : A6 11 0A             SM_St9_Lp   cp SM_State,#10         ; error state?
506/     33E : 6B 3C                            jr Z, SM_State_10
507/     340 : 66 FA 01                         tcm IRQ,#001h           ; P3.2 pulse occurred?
508/     343 : EB F6                            jr NZ, SM_St9_Lp        ;  no --> wait
509/     345 : E6 1C FF                         ld DelayL,#255
510/     348 : E6 1D 01                         ld DelayH,#1
511/     34B : A6 ED 01                         cp R13,#1               ; fixed DAC value?
512/     34E : EB 03                            jr NZ, SM_St9_3         ;  no -->
513/     350 : E6 1D 02                         ld DelayH,#2
514/     353 : D6 06 75             SM_St9_3    call Delay
515/     356 : 56 FA DE                         and IRQ,#0DEh           ; acknowledge P3.2 pulse
516/     359 : 46 00 16                         or P0,#OffsetDacStrb+Mux1D1+Mux1D0
517/     35C : E6 1C 01                         ld DelayL,#1
518/     35F : E6 1D 01                         ld DelayH,#1
519/     362 : D6 06 75                         call Delay
520/     365 : B0 1F                            clr FaultCtr
521/     367 : 8D 01 07                         jp SM_State_5           ; begin access settling
522/     36A :
523/     36A :
524/     36A :                      ; *********************************************************************
525/     36A :                      ;  Error Handling
526/     36A :                      ; *********************************************************************
527/     36A :                      ;
528/     36A :                      ; come here if State 5 failed twice
529/     36A : E4 00 17             State5_Error    ld StatusByte0,P0
530/     36D : E4 01 18                             ld StatusByte1,P1
531/     370 : E4 02 19                             ld StatusByte2,P2
532/     373 : E4 FA 1A                             ld StatusByte3,IRQ
533/     376 : 46 1A 90                             or StatusByte3,#090h
534/     379 : E6 11 0A                             ld SM_State,#10         ; set error state
535/     37C :                      ;
536/     37C :                      ; *** State 10 Error Entry
537/     37C :                      ;  State 10 is entered after T0_MSB access timeouts (T0 ints) occurred
538/     37C :                      ;  or counter T1 expired. Status0:3 holds P0, P1, P2, IRQ; SM_FaultState
539/     37C :                      ;  holds the last SM_State.
540/     37C : 8F                   SM_State_10     di
541/     37D : A6 1B 06                             cp SM_FaultState,#6     ; fault in state 6?
542/     380 : EB 1B                                jr NZ, SM_St10_1        ;  no -->
543/     382 : 56 FA DF                             and IRQ,#0DFh           ; acknowledge T1 interrupt
544/     385 : 46 F1 0C                             or TMR,#00Ch            ; load & start T1
545/     388 : B0 FB                                clr IMR                 ; disable all interrupts
546/     38A : E6 1C FF                             ld DelayL,#255
547/     38D : E6 1D 05                             ld DelayH,#5
548/     390 : D6 06 75                             call Delay              ; wait 5ms
549/     393 : E6 1F 01                             ld FaultCtr,#1          ; first retry
550/     396 : 9F                                   ei
551/     397 : 76 FA 20                             tm IRQ,#020h            ; on track
552/     39A : 6D 01 4E                             jp Z, SM_State_6        ; retry
553/     39D : 56 03 CF             SM_St10_1       and P3,#0FFh-SioRdy-SrvoRdy
554/     3A0 : E6 F2 04                             ld T1,#004h
555/     3A3 : E6 F3 04                             ld PRE1,#004h           ;  Timer 1 ext/1 single
556/     3A6 : 46 03 40                             or P3,#SrvoErr          ; set ServoErr
557/     3A9 : 20 1F                                inc FaultCtr            ; next retry
558/     3AB : A6 1F 03                             cp FaultCtr,#3          ; 3rd attempt?
559/     3AE : 7D 00 7C                             jp C, SM_State_2        ;  no, do error recal
560/     3B1 : E6 00 11                             ld P0,#OffsetDacStrb+ParkHeads
561/     3B4 : E6 01 DF                             ld P1,#Mux2D1+Mux2D0+OffsetDacZero
562/     3B7 : E6 11 0C                             ld SM_State,#12
563/     3BA : B0 FB                                clr IMR
564/     3BC : B0 FA                                clr IRQ
565/     3BE : 9F                                   ei
566/     3BF : 8D 00 39                             jp Cmd_Home             ; after 3 attempts go parking
567/     3C2 :
568/     3C2 :                      ; illegal command
569/     3C2 :                      ;
570/     3C2 : E4 11 1B             Cmd_Abort       ld SM_FaultState,SM_State       ; remember faulty state
571/     3C5 : E6 11 0B                             ld SM_State,#11
572/     3C8 : 46 03 40                             or P3,#SrvoErr
573/     3CB : E6 29 01                             ld LastPCmdAbort,#1     ; mark last command as aborted
574/     3CE : E6 1C CD                             ld DelayL,#205
575/     3D1 : E6 1D 01                             ld DelayH,#1
576/     3D4 : D6 06 75                             call Delay
577/     3D7 : 76 03 20                             tm P3,#SrvoRdy          ; ServoReady set?
578/     3DA : ED 01 99                             jp NZ, SM_State_6a      ;  yes --> go settling
579/     3DD : 8D 00 39                             jp Cmd_Home             ; otherwise go parking
580/     3E0 :
581/     3E0 :
582/     3E0 :                      ; *********************************************************************
583/     3E0 :                      ;  Command processing
```

```
584/     3E0 :                    ; *****************************************************************
585/     3E0 :                    ;
586/     3E0 :                    ; decode command given in CmdByte0 (04h), return vector in RR4
587/     3E0 :                    ; In State 1 (after power-up) only ReadStatus and Recal are allowed
588/     3E0 :                    ;
589/     3E0 : E4 04 1E           Cmd_Decode      ld CmdByte0H,CmdByte0   ; get Servo Command Byte
590/     3E3 : F0 1E                              swap CmdByte0H
591/     3E5 : 56 1E 0F                           and CmdByte0H,#00Fh     ; isolate command nibble
592/     3E8 : 31 50                              srp #80                 ; !!! invalid - a 2k device has
593/     3EA :                           assume RP:080h                   ;     only 124 registers (04:7Fh) !!!
594/     3EA : A6 1E 00                           cp CmdByte0H,#0         ; Read Status (0)?
595/     3ED : 6B 0F                              jr Z, LoadCmdAdrs       ;  yes -->
596/     3EF : 76 1E 04                           tm CmdByte0H,#4         ; Recal or Home (4, 7)?
597/     3F2 : EB 0A                              jr NZ, LoadCmdAdrs      ;  yes -->
598/     3F4 : 76 03 40                           tm P3,#SrvoErr          ; Servo Error?
599/     3F7 : EB 14                              jr NZ, CmdReject        ;  yes --> abort
600/     3F9 : A6 11 07                           cp SM_State,#7          ; State 7 (Start SIO Communication)?
601/     3FC : EB 0F                              jr NZ, CmdReject        ;  no --> abort
602/     3FE : 2C 04              LoadCmdAdrs     ld R2,#CmdVectors /256
603/     400 : 3C 13                              ld R3,#CmdVectors #256
604/     402 : 90 1E                              rl CmdByte0H
605/     404 : 04 1E E3                           add R3,CmdByte0H        ; calculate table index
606/     407 : C2 42              CmdReject1      ldc R4,@RR2             ; and get vector adress
607/     409 : 3E                                 inc R3
608/     40A : C2 52                              ldc R5,@RR2             ; return command vector in RR4
609/     40C : AF                                 ret
610/     40D :                    ; reject command
611/     40D : 2C 04              CmdReject       ld R2,#AbortVector /256
612/     40F : 3C 31                              ld R3,#AbortVector #256
613/     411 : 8B F4                              jr CmdReject1
614/     413 :
615/     413 : 04 33              CmdVectors      DW Cmd_ReadStatus       ; 0 ReadStatus
616/     415 : 01 93                              DW Cmd_SetOffset        ; 1 SetOffset
617/     417 : 03 C2                              DW Cmd_Abort            ; 2 (Diagnostic)
618/     419 : 03 C2                              DW Cmd_Abort            ; 3
619/     41B : 00 65                              DW Cmd_Recal            ; 4 DataRecal
620/     41D : 03 C2                              DW Cmd_Abort            ; 5
621/     41F : 03 C2                              DW Cmd_Abort            ; 6
622/     421 : 00 65                              DW Cmd_Recal            ; 7 FormatRecal
623/     423 : 01 DA                              DW Cmd_Access           ; 8 Access
624/     425 : 01 DA                              DW Cmd_Access           ; 9 AccessOffset
625/     427 : 03 C2                              DW Cmd_Abort            ; A
626/     429 : 03 C2                              DW Cmd_Abort            ; B
627/     42B : 00 39                              DW Cmd_Home             ; C Home
628/     42D : 03 C2                              DW Cmd_Abort            ; D
629/     42F : 03 C2                              DW Cmd_Abort            ; E
630/     431 : 03 C2              AbortVector     DW Cmd_Abort            ; F
631/     433 :
632/     433 :
633/     433 :                    ; command 0: ReadStatus
634/     433 :                    ;
635/     433 : E6 1C CD           Cmd_ReadStatus  ld DelayL,#205
636/     436 : E6 1D 02                           ld DelayH,#2
637/     439 : D6 06 75                           call Delay
638/     43C : E4 07 15                           ld Temp3,CmdByte3
639/     43F : 56 15 0F                           and Temp3,#00Fh         ; isolate status register
640/     442 : 8C 04                              ld R8,#StatusVectors /256
641/     444 : 9C 56                              ld R9,#StatusVectors #256
642/     446 : 90 15                              rl Temp3                        ; generate index
643/     448 : 04 15 E9                           add R9,Temp3
644/     44B : C2 A8                              ldc R10,@RR8
645/     44D : 9E                                 inc R9
646/     44E : C2 B8                              ldc R11,@RR8            ; get address
647/     450 : 30 EA                              jp @RR10                ;  call status routine
648/     452 : D6 06 14           SendStatus      call TxStatusBytes      ;  and send data to host
649/     455 : AF                                 ret
650/     456 :
651/     456 :                    ; Status 1 is the standard status which holds the current offset in Byte 1 bit 5:0
652/     456 : 04 76              StatusVectors   DW Status_0             ; 0   (resend last or get fault conditions)
653/     458 : 04 78                              DW Status_1             ; 1 P0,  P1,  P2,  IRQ
654/     45A : 04 86                              DW Status_2             ; 2 SIO, P3,  TMR, FLAGS
655/     45C : 04 94                              DW Status_3             ; 3 T0,  T1,  IMR, RP
656/     45E : 04 A2                              DW Status_4             ; 4 SPH, SPL, CmdByte0H, T0_MSB
657/     460 : 04 B0                              DW Status_5             ; 5 0Eh, 0Ch, SM_FaultState, P1_Mask
658/     462 : 04 BE                              DW Status_6             ; 6 last received command (21:24h)
659/     464 : 04 CC                              DW Status_7             ; 7 IRQ, P0,  P3,  P1_Mask
660/     466 : 04 DB                              DW Status_8             ; 8 last processed cmd (25:27), LastPCmdAbort
661/     468 : 04 DB                              DW Status_8             ; 9
662/     46A : 04 DB                              DW Status_8             ; A
663/     46C : 04 DB                              DW Status_8             ; B
664/     46E : 04 DB                              DW Status_8             ; C
665/     470 : 04 DB                              DW Status_8             ; D
666/     472 : 04 DB                              DW Status_8             ; E
667/     474 : 04 DB                              DW Status_8             ; F
668/     476 :
669/     476 :                    ; Status 0
670/     476 : 8B DA              Status_0        jr SendStatus
671/     478 :                    ; Status 1
672/     478 : E4 00 17           Status_1        ld StatusByte0,P0
673/     47B : E4 01 18                           ld StatusByte1,P1
674/     47E : E4 02 19                           ld StatusByte2,P2
675/     481 : E4 FA 1A                           ld StatusByte3,IRQ
676/     484 : 8B CC                              jr SendStatus
677/     486 :                    ; Status 2
678/     486 : E4 F0 17           Status_2        ld StatusByte0,SIO
679/     489 : E4 03 18                           ld StatusByte1,P3
680/     48C : E4 F1 19                           ld StatusByte2,TMR
681/     48F : E4 FC 1A                           ld StatusByte3,FLAGS
682/     492 : 8B BE                              jr SendStatus
683/     494 :                    ; Status 3
684/     494 : E4 F4 17           Status_3        ld StatusByte0,T0
685/     497 : E4 F2 18                           ld StatusByte1,T1
686/     49A : E4 FB 19                           ld StatusByte2,IMR
687/     49D : E4 FD 1A                           ld StatusByte3,RP
688/     4A0 : 8B B0                              jr SendStatus
689/     4A2 :                    ; Status 4
690/     4A2 : E4 FE 17           Status_4        ld StatusByte0,SPH
691/     4A5 : E4 FF 18                           ld StatusByte1,SPL
692/     4A8 : E4 1E 19                           ld StatusByte2,CmdByte0H
693/     4AB : E4 12 1A                           ld StatusByte3,T0_MSB
694/     4AE : 8B A2                              jr SendStatus
695/     4B0 :                    ; Status 5
696/     4B0 : E4 0E 17           Status_5        ld StatusByte0,0Eh
697/     4B3 : E4 0C 18                           ld StatusByte1,0Ch
698/     4B6 : E4 1B 19                           ld StatusByte2,SM_FaultState
699/     4B9 : E4 16 1A                           ld StatusByte3,P1_Mask
700/     4BC : 8B 94                              jr SendStatus
701/     4BE :                    ; Status 6
```

```
4BE : E4 21 17           Status_6        ld StatusByte0,LastRCmd0
4C1 : E4 22 18                           ld StatusByte1,LastRCmd1
4C4 : E4 23 19                           ld StatusByte2,LastRCmd2
4C7 : E4 24 1A                           ld StatusByte3,LastRCmd3
4CA : 8B 86                              jr SendStatus
4CC :                    ; Status 7
4CC : E4 FA 17           Status_7        ld StatusByte0,IRQ
4CF : E4 00 18                           ld StatusByte1,P0
4D2 : E4 03 19                           ld StatusByte2,P3
4D5 : E4 16 1A                           ld StatusByte3,P1_Mask
4D8 : 8D 04 52                           jp SendStatus
4DB :                    ; Status 8-F
4DB : E4 25 17           Status_8        ld StatusByte0,LastPCmd0
4DE : E4 26 18                           ld StatusByte1,LastPCmd1
4E1 : E4 27 19                           ld StatusByte2,LastPCmd2
4E4 : E4 29 1A                           ld StatusByte3,LastPCmdAbort
4E7 : 8D 04 52                           jp SendStatus
4EA :
4EA :
4EA :                    ; *****************************************************************
4EA :                    ;  Offset DAC adjustment routines
4EA :                    ; *****************************************************************
4EA :                    ;
4EA :                    ; set track offset
4EA : E6 F7 01           CallOffset      ld P3M,#001h
4ED : 8F                                 di
4EE : B0 F1                              clr TMR
4F0 : 76 06 40                           tm CmdByte2,#OfsSetAuto         ; use auto offset?
4F3 : EB 2B                              jr NZ, CallOffset1              ;  yes -->
4F5 : 76 06 1F                           tm CmdByte2,#01Fh               ; offset value given = 0?
4F8 : 6B 1E                              jr Z, CallOffset2               ;  yes -->
4FA : E4 06 1C                           ld DelayL,CmdByte2
4FD : 60 1C                              com DelayL
4FF : 56 1C 1F                           and DelayL,#01Fh                ; convert into DAC format
502 : E4 01 1D                           ld DelayH,P1                    ; get current P1 state
505 : 56 1D C0                           and DelayH,#0FFh-OffsetDacSign-OffsetDacZero    ;  zero DAC
508 : 76 06 80                           tm CmdByte2,#OfsForward         ; dir backwards (away from spindle)?
50B : EB 03                              jr NZ, CallOffset3              ;  no (bit 7 was 1) -->
50D : 46 1D 20                           or DelayH,#OffsetDacSign        ; set sign bit
510 : 44 1C 1D           CallOffset3     or DelayH,DelayL                ;  merge in magnitude
513 : E4 1D 01                           ld P1,DelayH                    ; write back P1
516 : 8B 72                              jr CallOffset4
518 :
518 :                    ; offset value = 0;
518 : 56 02 DF           CallOffset2     and P2,#0FFh-U2Hfast    ; clear P2.5 (U2H slow)
51B : 46 01 1F                           or P1,#OffsetDacZero    ; set offset DAC to zero
51E : 8B 6A                              jr CallOffset4
520 :
520 :                    ; set auto offset
520 :                    ; wait for Position Error Sample pulse, then look at Position Error signal from
520 :                    ; R/W board and adjust DAC in a successive aproximation fashion for minimum error
520 : 56 02 DF           CallOffset1     and P2,#0FFh-U2Hfast    ; clear P2.5 (U2H slow)
523 : 56 FA FD                           and IRQ,#0FDh           ; clear pending P3.1 interrupt
526 : A6 11 0A           CallOffset1_Lp  cp SM_State,#10         ; error state?
529 : 6B 68                              jr Z, CallOffset_Err
52B : 76 FA 02                           tm IRQ,#002h            ; Pos Err Sample (P3.3) pulse occurred?
52E : 6B F6                              jr Z, CallOffset1_Lp    ;  no --> wait
530 : E6 2B EF                           ld SAR_ANDmask,#0FFh-10h        ; AND mask for bit 4
533 : E6 2C 10                           ld SAR_ORmask,#010h             ; OR mask for bit 4
536 : 56 01 C0                           and P1,#0FFh-OffsetDacSign-OffsetDacZero        ;  zero DAC
539 : 46 01 1F                           or P1,#OffsetDacZero    ;  and set offset DAC to zero
53C : E6 1C 01                           ld DelayL,#1
53F : E6 1D 01                           ld DelayH,#1
542 : D6 06 75                           call Delay
545 : 76 02 04                           tm P2,#PosErr_Cmp       ; too low (P2.2 = 0)?
548 : 6B 20                              jr Z, CO_Auto_up        ;  yes -->
54A :                    ;
54A :                    ; DAC value is too high, decrease
54A : 56 01 DF                           and P1,#0FFh-OffsetDacSign      ; minus
54D : 54 2B 01           CO_Auto_down    and P1,SAR_ANDmask      ; clear bit
550 : E0 2B                              rr SAR_ANDmask          ; and shift down AND mask
552 : D6 06 75                           call Delay
555 : 76 02 04                           tm P2,#PosErr_Cmp       ; too high (P2.2 = 1)?
558 : EB 0C                              jr NZ, CO_Auto_down1    ;  no -->
55A : 44 2C 01                           or P1,SAR_ORmask        ; oherwise re-set bit
55D : E0 2C                              rr SAR_ORmask           ;  and shift down OR mask
55F : A6 2C 80           CO_Auto_down2   cp SAR_ORmask,#080h     ; done?
562 : 6B 26                              jr Z, CallOffset4       ;  yes, write down result
564 : 8B E7                              jr CO_Auto_down
566 : E0 2C              CO_Auto_down1   rr SAR_ORmask           ; shift down OR mask
568 : 8B F5                              jr CO_Auto_down2
56A :                    ;
56A :                    ; DAC value is too low, increase
56A : 46 01 20           CO_Auto_up      or P1,#OffsetDacSign    ; plus
56D : 54 2B 01           CO_Auto_up3     and P1,SAR_ANDmask      ; clear bit
570 : E0 2B                              rr SAR_ANDmask          ; and shift down AND mask
572 : D6 06 75                           call Delay
575 : 76 02 04                           tm P2,#PosErr_Cmp       ; too low (P2.2 = 0)?
578 : 6B 0C                              jr Z, CO_Auto_up1       ;  yes -->
57A : 44 2C 01                           or P1,SAR_ORmask        ; oherwise re-set bit
57D : E0 2C                              rr SAR_ORmask           ;  and shift down OR mask
57F : A6 2C 80           CO_Auto_up2     cp SAR_ORmask,#080h     ; done?
582 : 6B 06                              jr Z, CallOffset4       ;  yes, write down result
584 : 8B E7                              jr CO_Auto_up3
586 : E0 2C              CO_Auto_up1     rr SAR_ORmask           ; shift down OR mask
588 : 8B F5                              jr CO_Auto_up2
58A :
58A : 56 03 DF           CallOffset4     and P3,#0FFh-SrvoRdy    ; clear ServoRdy
58D : 56 02 BF                           and P2,#0FFh-Mux3D0
590 : 46 02 20                           or P2,#U2Hfast          ; set P2.5 (U2H fast)
593 : E6 1C 0F           CallOffset_Err  ld DelayL,#15
596 : E6 1D 01                           ld DelayH,#1
599 : D6 06 75                           call Delay
59C : 56 04 EF                           and CmdByte0,#0EFh      ; clear offset request from command
59F : B0 06                              clr CmdByte2
5A1 : 9F                                 ei
5A2 : AF                                 ret
5A3 :
5A3 :
5A3 :                    ; *****************************************************************
5A3 :                    ;  Communications
5A3 :                    ; *****************************************************************
5A3 :                    ;
5A3 :                    ; calculate command checksum, result in Temp1
5A3 :                    ;
5A3 : E4 04 13           CalcChkSum      ld Temp1,CmdByte0
5A6 : 04 05 13                           add Temp1,CmdByte1
5A9 : 04 06 13                           add Temp1,CmdByte2
```

```
820/     5AC : 04 07 13                          add Temp1,CmdByte3
821/     5AF : 60 13                             com Temp1
822/     5B1 : AF                                ret
823/     5B2 :
824/     5B2 :
825/     5B2 :                   ; receive host command and return the result in 21:24h
826/     5B2 :                   ; last command is moved to 25:28h, current to 21:24h.
827/     5B2 :                   ;
828/     5B2 : E6 F7 41          RxCmdBytes      ld P3M,#041h            ; parity off, SIO active, all pins I/O
829/     5B5 : E6 F5 05                          ld PRE0,#005h           ; CPUclk/1, continuous count
830/     5B8 : E4 20 F4                          ld T0,SioBaud           ; 3 for 19k2 or 1 for 57k6
831/     5BB : 8F                SM_ComState     di
832/     5BC : 56 FA 27                          and IRQ,#027h           ; clear serial interrupts
833/     5BF : E6 09 04                          ld BytePtr,#CmdByte0    ; first command byte in 04h
834/     5C2 : 46 F1 0B                          or TMR,#00Bh            ; enable T1, T0 and load T0
835/     5C5 : 9F                                ei
836/     5C6 : 46 03 10                          or P3,#SioRdy
837/     5C9 : E4 21 25                          ld LastPCmd0,LastRCmd0  ; move last received into last processed
838/     5CC : E4 22 26                          ld LastPCmd1,LastRCmd1
839/     5CF : E4 23 27                          ld LastPCmd2,LastRCmd2
840/     5D2 : E4 24 28                          ld LastPCmd3,LastRCmd3
841/     5D5 : A6 11 0A          WaitSioIrq      cp SM_State,#10         ; skip when error state
842/     5D8 : 6B 27                             jr Z, ComptSio
843/     5DA : 76 FA 08          SaveSioData     tm IRQ,#008h            ; wait for serial in interrupt
844/     5DD : 6B F6                             jr Z, WaitSioIrq
845/     5DF : 56 03 BF                          and P3,#0FFh-SrvoErr    ; clear ServoErr
846/     5E2 : F5 F0 09                          ld @BytePtr,SIO         ; get cmd from SIO
847/     5E5 : 20 09                             inc BytePtr             ; increment memory location
848/     5E7 : A6 09 09                          cp BytePtr,#CmdByte4+1
849/     5EA : 6B 05                             jr Z, TestChkSum        ; leave when we got all five
850/     5EC : 56 FA F7                          and IRQ,#0F7h           ; acknowledge interrupt
851/     5EF : 8B E9                             jr SaveSioData          ; and get next one
852/     5F1 :                   ; all data collected
853/     5F1 : 56 FA F7          TestChkSum      and IRQ,#0F7h           ; acknowledge last interrupt
854/     5F4 : D6 05 A3                          call CalcChkSum
855/     5F7 : A4 08 13                          cp Temp1,CmdByte4       ; checksum ok?
856/     5FA : 6B 05                             jr Z, ComptSio          ;  yes, valid command received
857/     5FC : 46 03 40                          or P3,#SrvoErr          ;  otherwise set ServoErr
858/     5FF : 8B BA                             jr SM_ComState          ;  and retry
859/     601 :                   ; we got a valid command
860/     601 : 56 03 EF          ComptSio        and P3,#0FFh-SioRdy     ; clear SioRdy
861/     604 : E4 04 21                          ld LastRCmd0,CmdByte0   ; update last received
862/     607 : E4 05 22                          ld LastRCmd1,CmdByte1
863/     60A : E4 06 23                          ld LastRCmd2,CmdByte2
864/     60D : E4 07 24                          ld LastRCmd3,CmdByte3
865/     610 : E6 F7 01                          ld P3M,#001h            ; SIO inactive
866/     613 : AF                                ret
867/     614 :
868/     614 :
869/     614 :                   ; send status back to host
870/     614 :                   ;
871/     614 : E6 F7 41          TxStatusBytes   ld P3M,#41h             ; parity off, SIO active, all pins I/O
872/     617 : 8F                                di
873/     618 : 56 FA 27                          and IRQ,#027h           ; clear serial interrupts
874/     61B : E6 09 04                          ld BytePtr,#CmdByte0    ; first byte to transmit
875/     61E : 46 F1 03                          or TMR,#003h            ; load and enable T0
876/     621 : 9F                                ei
877/     622 : 56 03 BF                          and P3,#0FFh-SrvoErr    ; clear SrvoErr
878/     625 : 46 03 10                          or P3,#SioRdy           ; set SioRdy
879/     628 : E6 1C 19                          ld DelayL,#019h
880/     62B : E6 1D 01                          ld DelayH,#001h
881/     62E : D6 06 75                          call Delay
882/     631 : E4 07 FE                          ld SPH,CmdByte3         ; remember 57k6 flag
883/     634 : E4 17 04                          ld CmdByte0,StatusByte0
884/     637 : E4 18 05                          ld CmdByte1,StatusByte1
885/     63A : E4 19 06                          ld CmdByte2,StatusByte2
886/     63D : E4 1A 07                          ld CmdByte3,StatusByte3
887/     640 : D6 05 A3                          call CalcChkSum         ; calculate checksum
888/     643 : E4 13 08                          ld CmdByte4,Temp1       ;  and store in CmdByte4
889/     646 : E6 1C 01                          ld DelayL,#001h
890/     649 : E6 1D 01                          ld DelayH,#001h
891/     64C : E5 09 F0          TxStB_Lp2       ld SIO,@BytePtr
892/     64F : 66 FA 10          TxStB_Lp1       tcm IRQ,#010h           ; wait for serial out interrupt
893/     652 : EB FB                             jr NZ, TxStB_Lp1
894/     654 : 56 FA EF                          and IRQ,#0EFh           ; acknowledge interrupt
895/     657 : D6 06 75                          call Delay
896/     65A : 20 09                             inc BytePtr
897/     65C : A6 09 09                          cp BytePtr,#CmdByte4+1  ; last one sent?
898/     65F : EB EB                             jr NZ, TxStB_Lp2        ;  no, do next byte
899/     661 :                   ;
900/     661 : 56 03 EF                          and P3,#0FFh-SioRdy     ; clear SioRdy
901/     664 : 56 FA EF                          and IRQ,#0EFh           ; acknowledge last interrupt
902/     667 : B0 07                             clr CmdByte3
903/     669 : B0 06                             clr CmdByte2
904/     66B : B0 04                             clr CmdByte0
905/     66D : B0 05                             clr CmdByte1
906/     66F : B0 29                             clr LastPCmdAbort
907/     671 : E6 F7 01                          ld P3M,#001h            ; SIO inactive
908/     674 : AF                                ret
909/     675 :
910/     675 :
911/     675 :                   ; wait, time given in DelayL:H
912/     675 :                   ;
913/     675 : E4 1D 13          Delay           ld Temp1,DelayH
914/     678 : E4 1C 14          DelayLoop2      ld Temp2,DelayL
915/     67B : 00 14             DelayLoop1      dec Temp2
916/     67D : EB FC                             jr NZ, DelayLoop1
917/     67F : 00 13                             dec Temp1
918/     681 : EB F5                             jr NZ, DelayLoop2
919/     683 : AF                                ret
920/     684 :
921/     684 :
922/     684 :                   ; ****************************************************************
923/     684 :
924/     684 :
925/     684 :                   ; called from State 8 (Access)
926/     684 : E4 04 0C          L684            ld 0Ch,CmdByte0         ; get magnitude MSB
927/     687 : 56 0C 03                          and 0Ch,#003h           ;  mask out siginficant bits
928/     68A : E4 05 0D                          ld 0Dh,CmdByte1         ; get magnitude LSB
929/     68D : 66 0C 03                          tcm 0Ch,#003h           ; MSB zero?
930/     690 : 6B 0A                             jr Z, L69c              ;  no, go on
931/     692 : 76 0D FC                          tm 0Dh,#0FCh            ; LSB < 3?
932/     695 : EB 05                             jr NZ, L69c             ;  no, go on
933/     697 : E4 0D 0C                          ld 0Ch,0Dh
934/     69A : 8B 03                             jr L69f
935/     69C :                   ;
936/     69C : D6 06 E6          L69c            call L6e6
937/     69F : E4 0C 0F          L69f            ld 0Fh,0Ch
```

```
 938/     6A2 : 76 0C 03                         tm 0Ch,#003h            ; MSB zero?
 939/     6A5 : 6B 17                            jr Z, L6be              ;  yes -->
 940/     6A7 : 76 04 04                         tm CmdByte0,#004h       ; access direction reverse?
 941/     6AA : 6B 0A                            jr Z, L6b6              ;  yes -->
 942/     6AC : 90 0E                 L6ac       rl 0Eh                  ; preset to 0C9h before recal
 943/     6AE : 90 0E                            rl 0Eh
 944/     6B0 : 00 0F                            dec 0Fh
 945/     6B2 : 6B 0A                            jr Z, L6be
 946/     6B4 : 8B F6                            jr L6ac
 947/     6B6 :                       ;
 948/     6B6 : E0 0E                 L6b6       rr 0Eh
 949/     6B8 : E0 0E                            rr 0Eh
 950/     6BA : 00 0F                            dec 0Fh
 951/     6BC : EB F8                            jr NZ, L6b6
 952/     6BE : F8 0E                 L6be       ld R15,0Eh
 953/     6C0 : 56 EF C0                         and R15,#0C0h           ; mask out bit 6+7
 954/     6C3 : F9 10                            ld 10h,R15
 955/     6C5 : A6 10 C0                         cp 10h,#0C0h
 956/     6C8 : 6B 13                            jr Z, Clr_OddEven
 957/     6CA : A6 10 80                         cp 10h,#080h
 958/     6CD : 6B 0E                            jr Z, Clr_OddEven
 959/     6CF : A6 10 00                         cp 10h,#0
 960/     6D2 : 6B 0E                            jr Z, Set_OddEven
 961/     6D4 : A6 10 40                         cp 10h,#040h
 962/     6D7 : 6B 09                            jr Z, Set_OddEven
 963/     6D9 : E6 11 0A                         ld SM_State,#10         ; set error state (should never happen)
 964/     6DC : AF                               ret
 965/     6DD :                       ;
 966/     6DD : 56 02 7F              Clr_OddEven and P2,#0FFh-OddEven
 967/     6E0 : 8B 03                            jr Clr_OE1
 968/     6E2 : 46 02 80              Set_OddEven or P2,#OddEven
 969/     6E5 : AF                    Clr_OE1    ret
 970/     6E6 :
 971/     6E6 : E6 0A 08              L6e6       ld 0Ah,#008h
 972/     6E9 : E6 0B 04                         ld 0Bh,#004h
 973/     6EC : A4 0C 0B                         cp 0Bh,0Ch
 974/     6EF : BB 02                            jr ugt,L6f3
 975/     6F1 : DF                               scf
 976/     6F2 : AF                               ret
 977/     6F3 :
 978/     6F3 : 10 0D                 L6f3       rlc 0Dh
 979/     6F5 : 10 0C                            rlc 0Ch
 980/     6F7 : 7B 05                            jr C,L6fe
 981/     6F9 : A4 0C 0B                         cp 0Bh,0Ch
 982/     6FC : BB 04                            jr ugt,L702
 983/     6FE : 24 0B 0C              L6fe       sub 0Ch,0Bh
 984/     701 : DF                               scf
 985/     702 : 00 0A                 L702       dec 0Ah
 986/     704 : EB ED                            jr NZ, L6f3
 987/     706 : 10 0D                            rlc 0Dh
 988/     708 : AF                               ret
 989/     709 :
 990/     709 :
 991/     709 :                       ; called from State 8 (Access)
 992/     709 :                       ;  get DAC value from table indexed by R3 (seek magnitude)
 993/     709 :                       ;  use fixed value 8 (017h) when R13 = 1
 994/     709 :                       ;  return P1 contents merged with new DAC value in P1_Mask
 995/     709 :                       ;
 996/     709 : E0 E0 E0 E0           CurveTable DB 0E0h, 0E0h, 0E0h, 0E0h
 997/     70D : E0 E1 E1 E2                      DB 0E0h, 0E1h, 0E1h, 0E2h
 998/     711 : E2 E3 E3 E4                      DB 0E2h, 0E3h, 0E3h, 0E4h
 999/     715 : E5 E5 E6 E7                      DB 0E5h, 0E5h, 0E6h, 0E7h
1000/     719 : E7 E8 E9 EA                      DB 0E7h, 0E8h, 0E9h, 0EAh
1001/     71D : EB EC ED EE                      DB 0Ebh, 0ECh, 0EDh, 0EEh
1002/     721 : EF F0 F1 F3                      DB 0EFh, 0F0h, 0F1h, 0F3h
1003/     725 : F4 F6 F8 FF                      DB 0F4h, 0F6h, 0F8h, 0FFh
1004/     729 :
1005/     729 : 31 30                 GetCurveValue   srp #030h
1006/     72B :                               assume RP:030h
1007/     72B : A6 ED 01                         cp R13,#1                       ; R13 = 1?
1008/     72E : 6B 29                            jr Z, GetCV_1                   ;  --> load DAC with 0F7h
1009/     730 :                       ;
1010/     730 : 39 13                            ld Temp1,R3
1011/     732 : 0C 07                            ld R0,#CurveTable /256
1012/     734 : 1C 09                            ld R1,#CurveTable #256
1013/     736 : E6 FE 1F                         ld SPH,#01Fh
1014/     739 : 24 13 FE                         sub SPH,Temp1
1015/     73C : 04 FE E1                         add R1,SPH                      ; calculate table index
1016/     73F : FB 01                            jr NC,GetCV_2
1017/     741 : 0E                               inc R0
1018/     742 : C2 50                 GetCV_2    ldc R5,@RR0                     ; get DAC value from table
1019/     744 : 00 E0                            dec R0                          ; ??
1020/     746 : 59 14                            ld Temp2,R5
1021/     748 : E4 01 13              GetCV_4    ld Temp1,P1                     ; get P1 value
1022/     74B : 56 13 E0                         and Temp1,#0FFh-OffsetDacZero   ; remove DAC bits
1023/     74E : 56 14 1F                         and Temp2,#OffsetDacZero
1024/     751 : 44 13 14                         or Temp2,Temp1                  ;  and merge in Temp2
1025/     754 : E4 14 16                         ld P1_Mask,Temp2                ; store DAC value in P1_Mask
1026/     757 : 8B 05                            jr GetCV_3                      ;  and leave
1027/     759 : E6 14 F7              GetCV_1    ld Temp2,#0FFh-8                ; set DAC to 8
1028/     75C : 8B EA                            jr GetCV_4
1029/     75E : AF                    GetCV_3    ret
1030/     75F :
1031/     75F :
1032/     75F :                       ; *********************************************************************
1033/     75F :                       ;  interrupt 0-3
1034/     75F :                       ; *********************************************************************
1035/     75F :                       ;
1036/     75F : 8F                    HW_Interrupt    di
1037/     760 : E6 FB 10                         ld IMR,#010h
1038/     763 : 56 02 FE                         and P2,#0FEh            ; clear P2.0
1039/     766 : 56 FA FD                         and IRQ,#0FDh
1040/     769 : 9F                               ei
1041/     76A : BF                               iret
1042/     76B :
1043/     76B :
1044/     76B :                       ; *********************************************************************
1045/     76B :                       ;  interrupt 4: access timeout
1046/     76B :                       ; *********************************************************************
1047/     76B :                       ;
1048/     76B : 66 11 03              T0_Interrupt    tcm SM_State,#3         ; Start Recal Motion
1049/     76E : 6B 13                            jr Z, T0_Int_3489
1050/     770 : 66 11 04                         tcm SM_State,#4         ; Recal Final Approach
1051/     773 : 6B 0E                            jr Z, T0_Int_3489
1052/     775 : 66 11 08                         tcm SM_State,#8         ; Access State
1053/     778 : 6B 09                            jr Z, T0_Int_3489
1054/     77A : A6 11 09                         cp SM_State,#9          ; Seek Final Approach
1055/     77D : 6B 04                            jr Z, T0_Int_3489
```

```
1056/     77F : D6 07 9C                          call SaveErrState
1057/     782 : BF                                iret
1058/     783 :                         ; State 3, 4, 8, 9: decrement MSB and return
1059/     783 : 00 12                   T0_Int_3489     dec T0_MSB
1060/     785 : 6B 04                                   jr Z, T0_Int1
1061/     787 : 56 FA EF                                and IRQ,#0EFh           ; acknowledge T0 int
1062/     78A : BF                                      iret
1063/     78B : D6 07 9C                T0_Int1         call SaveErrState
1064/     78E : BF                                      iret
1065/     78F :
1066/     78F :
1067/     78F :                         ; *****************************************************************
1068/     78F :                         ;  interrupt 5: track pulse counter
1069/     78F :                         ; *****************************************************************
1070/     78F :                         ;
1071/     78F : A6 11 06                T1_Interrupt    cp SM_State,#6
1072/     792 : 6B 04                                   jr Z, T1_Int_6
1073/     794 : D6 07 9C                                call SaveErrState
1074/     797 : BF                                      iret
1075/     798 :                         ; State 6
1076/     798 :                         T1_Int_6        ; (code deleted)
1077/     798 : D6 07 9C                                call SaveErrState
1078/     79B : BF                                      iret
1079/     79C :
1080/     79C :                         ; save error status
1081/     79C : 56 F1 03                SaveErrState    and TMR,#003h           ; load & enable T0
1082/     79F : E4 00 17                                ld StatusByte0,P0
1083/     7A2 : E4 01 18                                ld StatusByte1,P1
1084/     7A5 : E4 02 19                                ld StatusByte2,P2
1085/     7A8 : E4 FA 1A                                ld StatusByte3,IRQ
1086/     7AB : E4 11 1B                                ld SM_FaultState,SM_State       ; remember faulty state
1087/     7AE : E6 11 0A                                ld SM_State,#10         ; set Error State
1088/     7B1 : 56 03 DF                                and P3,#0FFh-SrvoRdy    ; clear ServoRdy
1089/     7B4 : 46 03 40                                or P3,#SrvoErr          ;  and set ServoErr
1090/     7B7 : 56 FA CF                                and IRQ,#0CFh           ; acknowledge T0 and T1 int
1091/     7BA : AF                                      ret
1092/     7BB :
1093/     7BB :
1094/     7BB :
1095/     7BB : =>FALSE                         if $>ROMsize
1096/     7BB :                                         error "\aROM size exceeded !!!"
1097/     7BB : =>TRUE                          elseif
1098/     7BB : FF FF FF FF FF FF                       DB ROMsize-$ dup(0FFh)  ; pad with $FF's
                FF FF FF FF FF FF
                FF FF FF FF FF FF
                FF FF FF FF FF FF
                FF FF FF FF FF FF
                FF FF FF FF FF FF
                FF FF FF FF FF FF
                FF FF FF FF FF FF
                FF FF FF FF FF FF
                FF FF FF FF FF FF
                FF FF FF FF FF FF
                FF FF FF
1099/     800 : [1095]                          endif
1100/     800 :                                 end
```

symbol table (* = unused):

ABORTVECTOR	431 C	*ACCESSMODE	80 -		
ACCFORWARD	4 -	*ARCHITECTURE	i386-unknown-win32 -		
*BIGENDIAN	0 -	*BRANCHEXT	0 -		
BYTEPTR	9 -	CALCCHKSUM	5A3 C		
CALLOFFSET	4EA C	CALLOFFSET1	520 C		
CALLOFFSET1_LP	526 C	CALLOFFSET2	518 C		
CALLOFFSET3	510 C	CALLOFFSET4	58A C		
CALLOFFSET_ERR	593 C	*CASESENSITIVE	0 -		
CLR_ODDEVEN	6DD C	CLR_OE1	6E5 C		
CMDBYTE0	4 -	CMDBYTE0H	1E -		
CMDBYTE1	5 -	CMDBYTE2	6 -		
CMDBYTE3	7 -	CMDBYTE4	8 -		
CMDREJECT	40D C	CMDREJECT1	407 C		
CMDVECTORS	413 C	CMD_ABORT	3C2 C		
CMD_ACCESS	1DA C	CMD_DECODE	3E0 C		
CMD_HOME	39 C	CMD_HOME1	4C C		
CMD_HOME2	5B C	CMD_READSTATUS	433 C		
CMD_RECAL	65 C	CMD_SETOFFSET	193 C		
COMPTSIO	601 C	*CONSTPI	3.141592653589793 -		
CO_AUTO_DOWN	54D C	CO_AUTO_DOWN1	566 C		
CO_AUTO_DOWN2	55F C	CO_AUTO_UP	56A C		
CO_AUTO_UP1	586 C	CO_AUTO_UP2	57F C		
CO_AUTO_UP3	56D C	CURVETABLE	709 C		
*DATE	7/31/2013 -	DELAY	675 C		
DELAYH	1D -	DELAYL	1C -		
DELAYLOOP1	67B C	DELAYLOOP2	678 C		
*FALSE	0 -	FAULTCTR	1F -		
FLAGS	FC -	*FULLPMMU	1 -		
GETCURVEVALUE	729 C	GETCV_1	759 C		
GETCV_2	742 C	GETCV_3	75E C		
GETCV_4	748 C	*HAS64	1 -		
*HASDSP	0 -	*HASFPU	0 -		
*HASPMMU	0 -	HW_INTERRUPT	75F C		
IMR	FB -	*INEXTMODE	0 -		
*INLWORDMODE	0 -	*INMAXMODE	0 -		
*INSRCMODE	0 -	*INSUPMODE	0 -		
*INT_VEC0	0 C	*INT_VEC1	2 C		
*INT_VEC2	4 C	*INT_VEC3	6 C		
*INT_VEC4	8 C	*INT_VEC5	A C		
*IPR	F9 -	IRQ	FA -		
L684	684 C	L69C	69C C		
L69F	69F C	L6AC	6AC C		
L6B6	6B6 C	L6BE	6BE C		
L6E6	6E6 C	L6F3	6F3 C		
L6FE	6FE C	L702	702 C		
LASTPCMD0	25 -	LASTPCMD1	26 -		
LASTPCMD2	27 -	LASTPCMD3	28 -		
LASTPCMDABORT	29 -	LASTRCMD0	21 -		
LASTRCMD1	22 -	LASTRCMD2	23 -		
LASTRCMD3	24 -	*LISTON	1 -		
LOADCMDADRS	3FE C	*MACEXP	1 -		
MOMCPU	8601 -	*MOMCPUNAME	Z8601 -		
MUX1D0	2 -	MUX1D1	4 -		
MUX1ENA	8 -	MUX2D0	40 -		
MUX2D1	80 -	MUX3D0	40 -		
*NESTMAX	100 -	ODDEVEN	80 -		
OFFSETDACSIGN	20 -	OFFSETDACSTRB	10 -		
OFFSETDACZERO	1F -	OFFSETFLAG	2A -		
OFSFORWARD	80 -	*OFSREADDAC	20 -		
OFSSETAUTO	40 -	P0	0 -		
P01M	F8 -	P1	1 -		
P1_MASK	16 -	P2	2 -		
P2M	F6 -	P3	3 -		
P3M	F7 -	*PACKING	0 -		
*PADDING	1 -	PARKHEADS	1 -		
*PORT21	2 -	PORT23	8 -		
PORT24	10 -	PORT31	2 -		
*PORT32	4 -	*PORT33	8 -		
POSERR_CMP	4 -	PRE0	F5 -		
PRE1	F3 -	RECALMODE	20 -		
*RELAXED	0 -	ROMSIZE	800 -		
RP	FD -	RXCMDBYTES	5B2 C		
SAR_ANDMASK	2B -	SAR_ORMASK	2C -		
SAVEERRSTATE	79C C	SAVESIODATA	5DA C		
SENDSTATUS	452 C	*SERVO_SI	1 -		
*SERVO_SO	80 -	SETTLINGMODE	40 -		
SET_ODDEVEN	6E2 C	SIO	F0 -		
SIOBAUD	20 -	SIORDY	10 -		
SM_COMSTATE	5BB C	SM_FAULTSTATE	1B -		
SM_ST10_1	39D C	SM_ST2_1	A6 C		
SM_ST2_2	A0 C	SM_ST3_LP	BD C		
SM_ST4_LP	E4 C	SM_ST5_LP	12E C		
SM_ST6_1	165 C	SM_ST6_2	18E C		
SM_ST6_3	1B8 C	SM_ST7_1	1BA C		
SM_ST7_2	1BE C	SM_ST8_1	1F2 C		
SM_ST8_10	2D6 C	SM_ST8_11	2EE C		
SM_ST8_12	1F8 C	SM_ST8_2	215 C		
SM_ST8_3	227 C	SM_ST8_4	22A C		
SM_ST8_5	281 C	SM_ST8_6	268 C		
SM_ST8_7	27B C	SM_ST8_8	299 C		
SM_ST8_9	290 C	SM_ST8_LP1	2A5 C		
SM_ST8_LP2	2A0 C	SM_ST9_1	309 C		
SM_ST9_2	320 C	SM_ST9_3	353 C		
SM_ST9_LP	33B C	SM_STATE	11 -		
SM_STATE_10	37C C	SM_STATE_2	7C C		
SM_STATE_5	107 C	SM_STATE_6	14E C		
SM_STATE_6A	199 C	SM_STATE_9	2F5 C		
SPH	FE -	SPL	FF -		
SRVOERR	40 -	SRVORDY	20 -		
STA57K6	80 -	*START	C C		
STATE5_ERROR	36A C	STATUSBYTE0	17 -		
STATUSBYTE1	18 -	STATUSBYTE2	19 -		
STATUSBYTE3	1A -	STATUSVECTORS	456 C		
STATUS_0	476 C	STATUS_1	478 C		
STATUS_2	486 C	STATUS_3	494 C		
STATUS_4	4A2 C	STATUS_5	4B0 C		
STATUS_6	4BE C	STATUS_7	4CC C		
STATUS_8	4DB C	*STDDEFZ8INC	1 -		
T0	F4 -	T0_INT1	78B C		
T0_INTERRUPT	76B C	T0_INT_3489	783 C		
T0_MSB	12 -	T1	F2 -		
T1_INTERRUPT	78F C	T1_INT_6	798 C		

```
 TEMP1 :                          13 - |  TEMP2 :                          14 - |
 TEMP3 :                          15 - |  TESTCHKSUM :                   5F1 C |
*TIME :                    16:13:37 - |  TMR :                           F1 - |
*TRUE :                           1 - |  TXSTATUSBYTES :                614 C |
 TXSTB_LP1 :                    64F C |  TXSTB_LP2 :                    64C C |
 U2GFAST :                         1 - |  U2HFAST :                       20 - |
*VERSION :                     142F - |  WAITSIOIRQ :                   5D5 C |
 ZERORAM_LP :                     22 C |

    239 symbols
     43 unused symbols
```

```
  codepages:
  ----------

STANDARD (0 changed characters)


0.03 seconds assembly time

   1158 lines source file
      2 passes
      0 errors
      0 warnings
```