Below is the actual header from the binary code module
in the 12/21/2004 release of the system software. All
binary commands listed are implemented and most are
fully tested at this point. To my knowledge all notations
are correct.

Headers from each command are also included

The headers refer to command station internal assembly language
code that will mean nothing to your code but may still offer some '
insight proper usage of each binary command.

The file below is best viewed with a mono-spaced font
such as Courier

```
.PAGE
.SUBTITLE       "FILE: BIN232.ASM"
;*****************************************************************************
;
;               The RS-232 port binary commands are designed to work
;               in a computer friendly mode.
;
;               Command format is:  <cmd number> <data> <data> ...
;
;               Commands range from 0x80 to 0xBF
;
;               The number of bytes in the command is determined by
;               using the lookup table "BIN_TABLE". the number returned from
;               the table is inclusive of the command number.
;
;               Commands and formats supported:
;
;
;
; CMD  FORMAT   DESCRIPTION                # OF BYTES   RESPONSES
;                                            RETURNED
;-----------------------------------------------------------------------------
; 0x80          NOP, dummy instruction          (1)     !
;-----------------------------------------------------------------------------
; 0x81 xx xx yy assign loco                     (1)     !,1,2
;-----------------------------------------------------------------------------
; 0x82          read clock                      (2)     <hours>
;                                                       <minutes>
;-----------------------------------------------------------------------------
; 0x83          Clock stop                      (1)     !
;-----------------------------------------------------------------------------
; 0x84          Clock start                     (1)     !
;-----------------------------------------------------------------------------
; 0x85 xx xx    Set clock hr/min                (1)     !,3
;-----------------------------------------------------------------------------
; 0x86 xx       Set clock 12/24                 (1)     !,3
;               0=12 hr 1=24 hr
;-----------------------------------------------------------------------------
; 0x87 xx       Set clock ratio                 (1)     !,3
;-----------------------------------------------------------------------------
; 0x88 xx xx    Dequeue packet by loco addr     (1)     !,1,2
;-----------------------------------------------------------------------------
; 0x89          Enable main trk, kill prog      (1)     !
;-----------------------------------------------------------------------------
; 0x8A yy       Return status of AIU yy         (4)     <current hi byte>
;                                                       <current lo byte>
;                                                       <change hi byte>
;
```

```
;                                                                <change lo byte>
;-------------------------------------------------------------------------------
; 0x8B          Kill main trk, enable prog       (1)     !
;-------------------------------------------------------------------------------
; 0x8C          Dummy instruction returns "!"    (3)     !,0x0D,0x0A
;               followed by CR/LF
;-------------------------------------------------------------------------------
; 0x8D xxxx mm  Set speed mode of loco xxxx      (1)     !,1,3
;               to mode mm, 1=14, 2=28, 3=128            <speed mode, 0 to 3>
;-------------------------------------------------------------------------------
; 0x8E aaaa nn <16 data bytes>                   (1)     !,4
;               Write nn bytes, start at aaaa
;               Must have 16 data bytes, pad
;               them out to 16 if necessary
;-------------------------------------------------------------------------------
; 0x8F aaaa     Read 16 bytes, start at aaaa     (16)    16 bytes
;-------------------------------------------------------------------------------
; 0x90 cc xx... Send 16 char message to Cab cc   (1)     !,2
;               LCD line 3.  xx = 16 ASCII char
;-------------------------------------------------------------------------------
; 0x91 cc xx... Send 16 char message to Cab cc   (1)     !,2
;               LCD line 4.  xx = 16 ASCII char
;-------------------------------------------------------------------------------
; 0x92 cc xx... Send 8 char message to Cab cc    (1)     !,2
;               LCD line 2 right.  xx = 8 char
;-------------------------------------------------------------------------------
; 0x93 ss xx xx xx                               (1)     !
;               Queue 3 byte packet to TEMP_Q
;               Send ss times
;-------------------------------------------------------------------------------
; 0x94 ss xx xx xx xx                            (1)     !
;               Queue 4 byte packet to TEMP_Q
;               Send ss times
;-------------------------------------------------------------------------------
; 0x95 ss xx xx xx xx xx                         (1)     !
;               Queue 5 byte packet to TEMP_Q
;               Send ss times
;-------------------------------------------------------------------------------
; 0x96 ss xx xx xx xx xx xx                      (1)     !
;               Queue 6 byte packet to TEMP_Q
;               Send ss times
;-------------------------------------------------------------------------------
; 0x97 aaaa xx  Write 1 byte,to aaaa             (1)     !
;-------------------------------------------------------------------------------
; 0x98 aaaa xx xx                                (1)     !
;               Write 2 bytes to aaaa
;-------------------------------------------------------------------------------
; 0x99 aaaa <4 data bytes>                       (1)     !
;               Write 4 bytes to aaaa
;-------------------------------------------------------------------------------
; 0x9A aaaa <8 data bytes>                       (1)     !
;               Write 8 bytes to aaaa
;-------------------------------------------------------------------------------
; 0x9B yy       Return status of AIU yy          (2)     <current hi byte>
;               (short form of command 0x8A)             <current lo byte>
;-------------------------------------------------------------------------------
; 0x9C xx       Execute macro number xx          (1)     !,0,3
;-------------------------------------------------------------------------------
; 0x9D aaaa     Read 1 bytes at memory addr aaaa (1)     1 byte
;-------------------------------------------------------------------------------
; 0x9E          Enter Programming track mode     (1)     ! = success
;                                                        3 = short circuit
;-------------------------------------------------------------------------------
```

```
; 0x9F          Exit Programming track mode      (1)     ! = success
;-------------------------------------------------------------------------
; 0xA0 aaaa xx  Program CV aa with data xx in    (1)     ! = success
;               paged mode                               0 = program track no
enabled
;-------------------------------------------------------------------------
; 0xA1 aaaa     Read CV aa in paged mode         (2)     !,0,3
;               NOTE: cv data followed ! for ok,
;                     0xff followed by 3 for can't read
;-------------------------------------------------------------------------
; 0xA2 aaaa nn dd  Locomotive speed, function,   (1)     !,3
;                  or consist command
;-------------------------------------------------------------------------
; 0xA3 xx xx    Queue 3 byte packet to TRK_Q     (1)     !,1
;               (replaces any packet with same
;               address if it exists)
;-------------------------------------------------------------------------
; 0xA4 xx xx... Queue 4 byte packet to TRK_Q     (1)     !,1
;               (replaces any packet with same
;               address if it exists)
;-------------------------------------------------------------------------
; 0xA5 xx xx... Queue 5 byte packet to TRK_Q     (1)     !,1
;               (replaces any packet with same
;               address if it exists)
;-------------------------------------------------------------------------
; 0xA6 rr xx    Program register rr with data xx (1)     ! = success
;               in register mode                         0 = program track no
enabled
;-------------------------------------------------------------------------
; 0xA7 rr       Read register rr in register mode(2)     !,3
;               NOTE: cv data followed ! for ok,         0 = program track no
enabled
;                     0xff followed by 3 for can't read
;-------------------------------------------------------------------------
; 0xA8 aaaa xx  Program CV aaaa with data xx     (1)     ! = success
;               in direct mode                           0 = program track no
enabled
;-------------------------------------------------------------------------
; 0xA9 aaaa     Read CV aaaa in direct mode      (2)     !,3
;               NOTE: cv data followed ! for ok,         0 = program track no
enabled
;                     0xff followed by 3 for can't read
;-------------------------------------------------------------------------
; 0xAA          Return software revision number  (3)     <data1>,<data2>,<data3>
;               FORMAT: VV.MM.mm
;-------------------------------------------------------------------------
; 0xAB          Perform a soft reset of command  (0)     Returns nothing
;               station (like cycling the power)
;-------------------------------------------------------------------------
; 0xAC          Perform a hard reset of command  (0)     Returns nothing
;               station (reset to factory defaults)
;-------------------------------------------------------------------------
;
;
;
;
;NOTE: a single byte of 0 will be returned of not in programming mode
;      for commands 0xA0,0xA1 and 0xA6- 0xA9
;
;
;
;
;              Errors returned:    '0'= command not supported
;                                  '1'= loco address out of range
;                                  '2'= cab address out of range
```

```
;                                '3'= data out of range
;                                '4'= byte count out of range
;                                '!'= command completed successfully
;
;*****************************************************************************


               ;command jump table
B_TBL          JP       PR_BIN_OK       ;0x80 128 NOP... just prints '!'
               JP       BCMD_81         ;0x81 129 assign loco to cab
               JP       BCMD_82         ;0x82 130 read clock
               JP       BCMD_83         ;0x83 131 Clock stop
               JP       BCMD_84         ;0x84 132 Clock resume (start)
               JP       BCMD_85         ;0x85 133 Set clock hours, minutes
               JP       BCMD_86         ;0x86 134 Set clock 12/24 hours
               JP       BCMD_87         ;0x87 135 Set clock speed (ratio)
               JP       BCMD_88         ;0x88 136 Dequeue by loco address
               JP       BCMD_89         ;0x89 137 Enable main, kill prg trk
               JP       BCMD_8A         ;0x8A 138 Return AIU status
               JP       BCMD_8B         ;0x8B 139 Kill main, enable prg trk
               JP       BCMD_8C         ;0x8C 140 Return !,CR,LF
               JP       BCMD_8D         ;0x8D 141 Set speed mode of loco
               JP       BCMD_8E         ;0x8E 142 Write 16 bytes to RAM
               JP       BCMD_8F         ;0x8F 143 Return 16 bytes from RAM
               JP       BCMD_90         ;0x90 144 Write 16 char to LCD line 3
               JP       BCMD_91         ;0x91 145 Write 16 char to LCD line 4
               JP       BCMD_92         ;0x92 146 Write 8 char to LCD line 2, right
               JP       BCMD_93         ;0x93 147 Queue 3 byte TEMP_Q packet
               JP       BCMD_94         ;0x94 148 Queue 4 byte TEMP_Q packet
               JP       BCMD_95         ;0x95 149 Queue 5 byte TEMP_Q packet
               JP       BCMD_96         ;0x96 150 Queue 6 byte TEMP_Q packet
               JP       BCMD_97         ;0x97 151 Write 1 byte to RAM
               JP       BCMD_98         ;0x98 152 Write 2 bytes to RAM
               JP       BCMD_99         ;0x99 153 Write 4 bytes to RAM
               JP       BCMD_9A         ;0x9A 154 Write 8 bytes to RAM
               JP       BCMD_9B         ;0x9B 155 Short form of CMD 0x8A
               JP       BCMD_9C         ;0x9C 156 Execute route macro
               JP       BCMD_9D         ;0x9D 157 Return 1 byte from RAM
               JP       BCMD_9E         ;0x9E 158 Enter program track mode
               JP       BCMD_9F         ;0x9F 159 Exit program track mode
               JP       BCMD_A0         ;0xA0 160 Write a CV in  paged mode
               JP       BCMD_A1         ;0xA1 161 Read a CV in paged mode
               JP       BCMD_A2         ;0xA2 162 Locomotive speed/function command
               JP       BCMD_A3         ;0xA3 163 Queue 3 byte TRK_Q packet
               JP       BCMD_A4         ;0xA4 164 Queue 4 byte TRK_Q packet
               JP       BCMD_A5         ;0xA5 165 Queue 5 byte TRK_Q packet
               JP       BCMD_A6         ;0xA6 166 Write in register mode
               JP       BCMD_A7         ;0xA7 167 Read in register mode
               JP       BCMD_A8         ;0xA8 168 Write in direct mode
               JP       BCMD_A9         ;0xA9 169 Read in direct mode
               JP       BCMD_AA         ;0xAA 170 Return C/S software version
               JP       BCMD_AB         ;0xAB 171 Command station soft reset
               JP       BCMD_AC         ;0xAC 172 Command station hard reset
               JP       BERR_0          ;0xAD 173
               JP       BERR_0          ;0xAE 174
               JP       BERR_0          ;0xAF 175
               JP       BERR_0          ;0xB0 176
               JP       BERR_0          ;0xB1 177
               JP       BERR_0          ;0xB2 178
               JP       BERR_0          ;0xB3 179
               JP       BERR_0          ;0xB4 180
               JP       BERR_0          ;0xB5
```

```
                JP      BERR_0          ;0xB6
                JP      BERR_0          ;0xB7
                JP      BERR_0          ;0xB8
                JP      BERR_0          ;0xB9
                JP      BERR_0          ;0xBA
                JP      BERR_0          ;0xBB
                JP      BERR_0          ;0xBC
                JP      BERR_0          ;0xBD
                JP      BERR_0          ;0xBE
                JP      BERR_0          ;0xBF
                JP      BERR_0          ;0xC0 this command ALWAYS BERR_0
                JP      BERR_0          ;0xC1 this command ALWAYS BERR_0
                JP      BERR_0          ;0xC2 this command ALWAYS BERR_0
                JP      BERR_0          ;0xC3 this command ALWAYS BERR_0


;*********************************************************************
;
;               BIN_TABLE contains the expected byte count of
;               each binary mode command
;
;*********************************************************************

BIN_TABLE       DB      1               ;0x80 NOP
                DB      4               ;0x81 ASSIGN
                DB      1               ;0x82 READ CLOCK
                DB      1               ;0x83 STOP CLOCK
                DB      1               ;0x84 START CLOCK
                DB      3               ;0x85 SET CLOCK H/M
                DB      2               ;0x86 SET 12/24 HOUR MODE
                DB      2               ;0x87 SET CLOCK RATIO
                DB      3               ;0x88 DEQUEUE TRK_Q PACKET
                DB      1               ;0x89 CONNECT MAIN,DISCONNECT PROG TRK
                DB      2               ;0x8A RETURN AIU STATUS, LONG FORM
                DB      1               ;0x8B DISCONNECT MAIN, CONNECT PROG TRK
                DB      1               ;0x8C RETURN "!",CR,LF
                DB      4               ;0x8D SET SPEED MODE OF LOCO
                DB      20              ;0x8E WRITE BYTES TO RAM
                DB      3               ;0x8F READ 16 BYTES FROM RAM
                DB      18              ;0x90 WRITE LCD LINE 3
                DB      18              ;0x91 WRITE LCD LINE 4
                DB      10              ;0x92 WRITE LCD LINE 2, RIGHT
                DB      5               ;0x93 3 BYTE PACKET TO TEMP_Q
                DB      6               ;0x94 4 BYTE PACKET TO TEMP_Q
                DB      7               ;0x95 5 BYTE PACKET TO TEMP_Q
                DB      8               ;0x96 6 BYTE PACKET TO TEMP_Q
                DB      4               ;0x97 Write 1 byte to RAM
                DB      5               ;0x98 Write 2 bytes to RAM
                DB      7               ;0x99 Write 4 bytes to RAM
                DB      11              ;0x9A Write 8 bytes to RAM
                DB      2               ;0x9B Short form of AIU status
                DB      2               ;0x9C Execute route macro
                DB      3               ;0x9D read1byte from RAM
                DB      1               ;0x9E Enter program track mode
                DB      1               ;0x9F Exit program track mode
                DB      4               ;0xA0 Write a CV in  paged mode
                DB      3               ;0xA1 Read a CV in paged mode
                DB      5               ;0xA2 Locomotive command
                DB      4               ;0xA3 3 BYTE PACKET TO TRK_Q
                DB      5               ;0xA4 4 BYTE PACKET TO TRK_Q
                DB      6               ;0xA5 5 BYTE PACKET TO TRK_Q
                DB      3               ;0xA6 Write in register mode
                DB      2               ;0xA7 Read in register mode
```

```
                DB      4               ;0xA8 Write in direct mode
                DB      3               ;0xA9 Read in direct mode
                DB      1               ;0xAA Return C/S software version
                DB      1               ;0xAB Command station soft reset
                DB      1               ;0xAC Command station hard reset
                DB      1               ;0xAD
                DB      1               ;0xAE
                DB      1               ;0xAF
                DB      1               ;0xB0
                DB      1               ;0xB1
                DB      1               ;0xB2
                DB      1               ;0xB3
                DB      1               ;0xB4
                DB      1               ;0xB5
                DB      1               ;0xB6
                DB      1               ;0xB7
                DB      1               ;0xB8
                DB      1               ;0xB9
                DB      1               ;0xBA
                DB      1               ;0xBB
                DB      1               ;0xBC
                DB      1               ;0xBD
                DB      1               ;0xBE
                DB      1               ;0xBF
                DB      1               ;0xC0
                DB      1               ;0xC1
                DB      1               ;0xC2
                DB      1               ;0xC3

;*************************************************************************
;
;               BCMD_81 assigns a loco to a cab
;               Loco address for this command is always 2 bytes. The first
;               byte is zero in the case of a short address. If the address
;               is long then bits 6,7 of first byte must be set to 1
;
;               Command Format:  0x81 xxxx cc
;                        xxxx = loco address
;                        cc   = cab number (0x01 to 0x3f)
;
;               Returns: ! = success
;                        1 = bad loco address
;                        2 = bad cab address
;
;*************************************************************************

;*************************************************************************
;
;               BCMD_82 returns the fast clock to the RS232 port
;               in binary mode
;
;               Format:           0x82
;
;               Returns 2 bytes:  HOURS
;                                 MINUTES
;
;*************************************************************************

;*************************************************************************
;
;               BCMD_83 stops the scale time clock
;
;               Format:  0x83  Clock stop
```

```
;
;               Returns: ! = success
;
;*******************************************************************************
;
;*******************************************************************************
;
;               BCMD_84 starts the scale time clock
;
;               FORMAT:  0x84    Clock resume (start)
;
;               Returns: ! = success
;
;*******************************************************************************
;
;*******************************************************************************
;
;               BCMD_85 sets the scale time clock
;
;               FORMAT:    0x85 xx xx    Set clock hours, minutes
;
;               Returns:   ! = success
;                          3 = hours or minutes data too large
;
;*******************************************************************************
;
;*******************************************************************************
;
;               BCMD_86 sets the scale time 12/24 hour format
;
;               Format:  0x86 xx     Set clock 12/24 hours
;                             xx=0 means 12 hour clock
;                             xx=1 means 24 hour clock
;
;               Returns:   ! = success
;                          3 = 12/24 format not 0 or 1
;
;*******************************************************************************
;
;*******************************************************************************
;
;               BCMD_87 sets the scale time clock ratio
;
;               Format:  0x87 xx       Set clock speed (ratio)
;
;               Returns:   ! = success
;                          3 = Ratio greater than 25 or zero
;
;*******************************************************************************
;
;*******************************************************************************
;
;               BCMD_88 reads loco address from BIN_BUFF, finds the
;               corresponding entry in TRK_Q and deletes the packet
;               from the TRK_Q.
;
;               The loco address for this command is always 2 bytes. The
;               first byte is zero in the case of a short address. If the
;               address is long then bits 6,7 of first byte must be set to 1
;
;               The loco address is used for comparison to see if a packet
;               should be replaced.
;
```

```
;               Command Format:  0x88 xxxx
;                                xxxx = 8/14 bit address
;
;               returns:         ! if success
;                                1 if bad loco address
;                                3 if packet not found
;
;*******************************************************************************

;*******************************************************************************
;
;               BCMD_89 connects main, disconnects programming track.
;
;               Command Format:  0x89
;
;               returns:         ! (always success)
;
;*******************************************************************************

;*******************************************************************************
;
;               BCMD_8A returns status of Auxilary Input Unit.
;               Returns four bytes.  The first 2 bytes are a bit map
;               of the 14 AIU inputs.  The last 2 bytes are a bit map
;               of any changes since this command was last given.
;               If the cab is greater than 63 it will be "forced" to 0.
;               The first time this command is given for a cab after the
;               command station is powered up or reset the change bytes
;               will be 0x3fff.
;
;               format:  0x8A cc   (cc=cab number 0-63)
;
;               returns 4 bytes : current high byte (AIU inputs 9-14)
;                                 current low byte  (AIU inputs 1-8)
;                                 change high byte  1= input changed since
;                                                   last read, 0=no change
;                                 change low byte   1=input changed since
;                                                   last read, 0=no change
;
;*******************************************************************************


;*******************************************************************************
;
;               BCMD_8B disconnects main track and clears all packet queues
;               note: programming track is enabled.
;
;               Format:  0x8B
;
;               Returns: ! (always success)
;
;*******************************************************************************

;*******************************************************************************
;
;               BCMD_8C prints "!",CR,LF to the RS-232 port in Binary Mode
;
;*******************************************************************************

;*******************************************************************************
;
;               BCMD_8D - sets the speed mode of loco dddd
;
```

```
;              Command Format:  0x8D xxxx mm
;                               xxxx = 8/14 bit address
;                                (short address 1st byte=0)
;                                (long address 1st byte bits 6,7 are set to 1)
;                               mm = mode    1=14spd, 2=28spd, 3=128spd
;
;              returns:         ! if success
;                               1 if bad loco address
;                               3 if bad speed mode, only 1,2 or 3 accepted
;
;*******************************************************************************

;*******************************************************************************
;
;              BCMD_8E writes bytes to a command station RAM address
;              16 data bytes are always required by the command. Only
;              the specified number will be written.   You must "pad"
;              out the command with extra bytes to meet the 16 byte
;              requirement even if all are not to be written.
;
;              Command Format: "0x8E aaaa nn <16 data bytes>"
;                               aaaa= RAM address, high byte first
;                               nn  = number of bytes to actually write
;
;              Returns:         !,4
;
;*******************************************************************************

;*******************************************************************************
;
;              BCMD_8F returns 16 bytes from a RAM address
;
;              Command Format: "0x8F aaaa "
;                               aaaa= RAM address, high byte first
;
;              Returns:         16 data bytes
;
;*******************************************************************************

;*******************************************************************************
;
;              BCMD_90 sends a message LCD line 3 of a cab
;              The characters to be printed are NOT range checked.
;              They must be in the range of 0x20 to 0x5F
;
;              Command Format:  0x90 cc <16 ASCII characters>
;                               cc  = cab number (0-63)
;
;              Returns:         ! for success
;                               2 bad cab address
;
;*******************************************************************************

;*******************************************************************************
;
;              BCMD_91 sends a message LCD line 4 of a cab
;              The characters to be printed are NOT range checked.
;              They must be in the range of 0x20 to 0x5F
;
;
;              Command Format:  0x91 cc <16 ASCII characters>
;                               cc  = cab number (0-63)
;
```

```
;               Returns:         ! for success
;                                2 bad cab address
;
;*******************************************************************************

;*******************************************************************************
;
;               BCMD_92 sends a message LCD the right hand side of
;               line 2 of a cab.
;               The characters to be printed are NOT range checked.
;               They must be in the range of 0x20 to 0x5F
;
;
;               Command Format:  0x92 cc <8 ASCII characters>
;                                cc  = cab number (0-63)
;
;               Returns:         ! for success
;                                2 bad cab address
;
;*******************************************************************************

;*******************************************************************************
;
;               BCMD_93 reads a 3 byte packet to put in TEMP_Q.
;               Packet addresses are NOT verified so be careful!
;               A "send times" of zero is OK... the packet
;               will not be sent.  If "send times" of 255 is requested
;               it will be adjusted to 254, due to a system limitation.
;               The packet checksum must be included with the packet data
;
;               Command Format:  0x93 <send times> <3 packet bytes>
;
;               Returns:         ! = success
;
;*******************************************************************************

;*******************************************************************************
;
;               BCMD_94 reads a 4 byte packet from BIN_BUFF to put in TEMP_Q.
;               Packet addresses are NOT verified so be careful!
;               A "send times" of zero is OK... the packet
;               will not be sent.  If "send times" of 255 is requested
;               it will be adjusted to 254, due to a system limitation.
;               The packet checksum must be included with the packet data
;
;               Command Format:  0x94 <send times> <4 packet bytes>
;
;               Returns:         ! = success
;
;*******************************************************************************

;*******************************************************************************
;
;               BCMD_95 reads a 5 byte packet from BIN_BUFF to put in TEMP_Q.
;               Packet addresses are NOT verified so be careful!
;               A "send times" of zero is OK... the packet
;               will not be sent.  If "send times" of 255 is requested
;               it will be adjusted to 254, due to a system limitation.
;               The packet checksum must be included with the packet data
;
;
;               Command Format:  0x95 <send times> <5 packet bytes>
;
```

```
;               Returns:         ! = success
;
;*****************************************************************************
;
;*****************************************************************************
;
;               BCMD_96 reads a 6 byte packet from BIN_BUFF to put in TEMP_Q.
;               Packet addresses are NOT verified so be careful!
;               A "send times" of zero is OK... the packet
;               will not be sent.  If "send times" of 255 is requested
;               it will be adjusted to 254, due to a system limitation.
;               The packet checksum must be included with the packet data
;
;               Command Format: 0x96 <sendtimes> <6 packet bytes>
;
;               Returns:        ! = success
;
;*****************************************************************************
;
;*****************************************************************************
;
;               BCMD_97 writes 1 byte to a command station RAM address
;
;               Command Format:  0x97 aaaa xx
;                                aaaa= RAM address, high byte first
;                                xx  = byte to write
;
;               Returns:         !
;
;*****************************************************************************
;
;*****************************************************************************
;
;               BCMD_98 writes 2 bytes to a command station RAM address
;
;               Command Format:  0x98 aaaa xx xx
;                                aaaa= RAM address, high byte first
;                                xx  = bytes to write
;
;               Returns:         !
;
;*****************************************************************************
;
;*****************************************************************************
;
;               BCMD_99 writes 4 bytes to a command station RAM address
;
;               Command Format: "0x99 aaaa <4 data bytes>"
;                                aaaa= RAM address, high byte first
;
;               Returns:         !
;
;*****************************************************************************
;
;*****************************************************************************
;
;               BCMD_9A writes 8 bytes to a command station RAM address
;
;               Command Format: "0x9A aaaa <8 data bytes>"
;                                aaaa= RAM address, high byte first
;
;               Returns:         !
;
```

```
;*************************************************************************

;*************************************************************************
;
;               BCMD_9B This is a short form of CMD 0x8A.  It returns
;               only the first 2 bytes of command 0x8A.
;               The 2 bytes are a bit map of the 14 AIU inputs.
;               If the cab is greater than 63 it will be "forced" to 0.
;
;               format:  0x9B cc   (cc=cab number 0-63)
;
;               returns 2 bytes : current high byte (AIU inputs 9-14)
;                                 current low byte  (AIU inputs 1-8)
;
;*************************************************************************

;*************************************************************************
;
;               BCMD_9C executes a previously defined macro for route control
;
;               Command Format:  0x9C xx    (xx = macro number, 0->255)
;
;               Returns:         ! = success
;                                0 = "empty" macro
;
;*************************************************************************

;*************************************************************************
;
;               BCMD_9D returns 1 byte from a RAM address
;
;               Command Format: "0x9D aaaa "
;                                aaaa= RAM address, high byte first
;
;               Returns:         1 data byte
;
;*************************************************************************

;*************************************************************************
;
;               BCMD_9E Enters Program track mode.  Power is removed
;               from mainline and applied to program track.  The queues
;               are formatted to send reset packets
;
;               Command Format: 0x9E
;
;               Returns: ! = success
;
;*************************************************************************

;*************************************************************************
;
;               BCMD_9F Returns from Program track mode.  Power is restored
;               to mainline and removed from program track.  The queues
;               are reinitialized for normal operation
;
;               Command Format: 0x9F
;
;               Returns: ! = success
;
;*************************************************************************

;*************************************************************************
```

```
;
;               BCMD_A0 Writes a CV in paged mode
;
;               Command Format: 0xA0 aaaa nn
;                                    aaaa = CV address (high byte first)
;                                    nn = cv data
;
;               Returns: ! = success
;                        0 = not in program track mode
;
;*******************************************************************************

;*******************************************************************************
;
;               BCMD_A1 Reads a CV in paged mode
;
;               Command Format: 0xA1 aa aa  aaaa = CV address (high byte first)
;
;               Returns: CV value followed by ! for success
;                        0xff followed by 3 for can't read CV
;                        0 = not in program track mode
;
;*******************************************************************************

;*******************************************************************************
;
;               BCMD_A2 sends a speed or function packet to a locomotive.
;               First ix is set to the computer cab context page.  This is
;               done so that type c cab commands can be called.  These
;               commands are all followed by a jump to PAGE_DONE (the normal
;               exit for the cab ping handler when a command is finished).
;               PAGE_DONE assumes that a local stack has been setup and the
;               main stack pointer must be restored.  By setting up the
;               computer cab with its own context page and local stack all
;               normal cab routines can be used.
;
;               The loco address is built up in (ix+ADDR_H) and (ix+ADDR_L)
;               on the computer cab context page.
;
;               Op_1 and data_1 are saved into (ix+P_RESP1) and (ix+P_RESP2)
;               respectively.
;
;               Then C_PROCESS_LOC is called.  When PAGE_DONE returns to
;               this code a normal binary command exit is performed
;
;               Command Format:  0xA2 <addr_h> <addr_l> <op_1> <data_1>
;
;               Addr_h and Addr_l are the loco address in DCC format.
;               If a long address is in use, bits 6 and 7 of the high byte are set.
;                 Ex:  Long address  3 = 0xc0 0x03
;                      Short address 3 = 0x00 0x03
;
;
;                op_1       data_1    Operation description
;
;                01         0-7f      Reverse 28 speed command
;                02         0-7f      Forward 28 speed command
;                03         0-7f      Forward 128 speed command
;                04         0-7f      Reverse 128 speed command
;                05         0         Estop reverse command
;                06         0         Estop forward command
;                07         0-1f      Function group 1 (same format as FG1 DCC packet
;                08         0-0f      Function group 2 (same format as FG2 DCC packet
```

```
;               09          0-0f       Function group 3 (same format as FG3 DCC packet
;               0a          0-7f       Set reverse consist address for lead loco
;               0b          0-7f       Set forward consist address for lead loco
;               0c          0-7f       Set reverse consist address for rear loco
;               0d          0-7f       Set forward consist address for rear loco
;               0e          0-7f       Set reverse consist address for additional loco
;               0f          0-7f       Set forward consist address for additional loco
;               10          0          Del loco from consist
;               11          0          Kill consist
;               12          0-9        Change momentum level for loco or consist
;               13-7f       reserved   reserved
;
;             Returns:          ! = success
;                               1 = bad loco address
;
;*****************************************************************************

;*****************************************************************************
;
;             BCMD_A3 reads a 3 byte packet from BIN_BUFF and puts it in
;             TRK_Q.  TRK_Q is only for speed and direction commands to
;             a loco.  Do not send anything else as any other command
;             will replace the existing speed command for that loco.
;             TRK_Q is limited to loco addresses.
;             The loco address is used for comparison to see if a packet
;             should be replaced.
;             The packet checksum must be included with the packet data
;
;             Command Format:  0xA3 <data> <data> <data>
;
;             Returns:          ! = success
;                               1 = bad loco address
;
;*****************************************************************************

;*****************************************************************************
;
;             BCMD_A4 reads a 4 byte packet from BIN_BUFF and puts it in
;             TRK_Q.  TRK_Q is only for speed and direction commands to
;             a loco.  Do not send anything else as any other command
;             will replace the existing speed command for that loco.
;             TRK_Q is limited to loco addresses.
;             The loco address is used for comparison to see if a packet
;             should be replaced.
;             The packet checksum must be included with the packet data
;
;             Command Format:  0xA4 <data> <data> <data> <data>
;
;             Returns:          ! = success
;                               1 = bad loco address
;
;*****************************************************************************

;*****************************************************************************
;
;             BCMD_A5 reads a 5 byte packet from BIN_BUFF and puts it in
;             TRK_Q.  TRK_Q is only for speed and direction commands to
;             a loco.  Do not send anything else as any other command
;             will replace the existing speed command for that loco.
;             TRK_Q is limited to loco addresses.
;             The loco address is used for comparison to see if a packet
;             should be replaced.
;             The packet checksum must be included with the packet data
```

```
;
;               Command Format:  0xA5 <data> <data> <data> <data> <data>
;
;               Returns:         ! = success
;                                1 = bad loco address
;
;*******************************************************************************

;*******************************************************************************
;
;               BCMD_A6 Writes a register
;
;               Command Format: 0xA8 rr xx   rr=register xx=data
;
;               Returns: ! = success
;                        0 = not in program track mode
;
;*******************************************************************************

;*******************************************************************************
;
;               BCMD_A7 read a register
;
;               Command Format: 0xA7 rr    rr=register
;
;               Returns: CV data followed by ! for success
;                        0xff followed by 3 for can't read
;                        0 = not in program track mode
;
;*******************************************************************************

;*******************************************************************************
;
;               BCMD_A8 Writes a CV in direct mode
;
;               Command Format: 0xA8 aaaa xx  address aaaa, data xx
;
;               Returns: ! = success
;                        0 = not in program track mode
;
;*******************************************************************************

;*******************************************************************************
;
;               BCMD_A9 Reads a CV in direct mode
;
;               Command Format: 0xA9 aa aa  aaaa = CV address (high byte first)
;
;               Returns: CV data followed by ! for success
;                        0xff followed by 3 for can't read CV
;                        0 = not in program track mode
;
;*******************************************************************************

;*******************************************************************************
;
;               BCMD_AA returns 3 bytes of software revision number
;
;               Command Format:  0xAA
;
;               Returns:         1st byte: version number
;                                2nd byte: major revision number
;                                3rd byte: minor revision number
```

```
;
;*************************************************************************
;
;*************************************************************************
;
;
;               BCMD_AB Soft reset of command station.  Sets command
;               Station to power up condition.
;
;               Command Format:  0xAB
;
;               Returns:         nothing
;
;*************************************************************************
;
;*************************************************************************
;
;
;               BCMD_AC Hard reset of command station.   Clears all
;               RAM and resets command station to original factory
;               defaults.  All stored information is destroyed.
;                Note: the baud rate will be set to 9600
;
;               Command Format:  0xAC
;
;               Returns:         nothing
;
;*************************************************************************
```