```
##############################################################################
##                                                                          ##
##      Copyright (c) Microsoft Corporation 1990                            ##
##      All rights reserved.                                                ##
##                                                                          ##
##############################################################################
#
##
### This is the makefile for the LADDR compliant sample SCSI BID
##
#

.SUFFIXES:
.SUFFIXES: .c .obj .asm .inc .h .rc .lib .bas

#
#       Debug definitions - comment out the unwanted line
#
DBG= -DPRINTF           # Debug messages enabled

#DBG=                     # Debug messages disabled

#
#       Definitions for the C Compiler
#
CC=cl

CFLAGS= -c -Zp1 -Asnw -G2s -PLM $(DBG)

CINC= -I../../laddrh

#
#       Definitions for the assembler
#
ASM= masm

MFLAGS= -t

AINC= -I../../laddrinc

#
#       list of obj's for sample bid
#
OBJS=   bidstart.obj bidaer.obj bidinit.obj\
        bidintr.obj  bidreq.obj bidcomp.obj  bidsubr.obj
#
#       list of lst's for sample bid
#
LSTS=   bidstart.lst bidaer.lst bidinit.lst\
        bidintr.lst  bidreq.lst bidcomp.lst  bidsubr.lst

#
#       General mechanism for creating obj's and lst's
#
.asm.obj:
        $(ASM) $(AINC) $(MFLAGS) $(DBG) $*.asm;

.asm.lst:
        $(ASM) $(AINC) $(MFLAGS) $(DBG) -N -L $*.asm;

.c.obj:
        $(CC)  $(CINC) $(CFLAGS) $*.c

.c.lst:
        $(CC)  $(CINC) $(CFLAGS) -Fc$*.lst -Fo$*.obj $*.c
```

MAKEFILE

```
#
#       Basic dependancies and mechanism for creating the sample bid
#
samplbid.sys:   $(OBJS) samplbid.lnk samplbid.def makefile
                link @samplbid.lnk
                mapsym samplbid.map

#
#       Basic dependancies and mechanism for creating all listing files
#
listall:        $(LSTS) samplbid.lnk samplbid.def makefile
                link @samplbid.lnk
                mapsym samplbid.map

#
#       Mechanism for re-creating all the .sys and all the obj's
#
clean:
        -del *.obj
        -del *.s*
        -del *.map
        -nmake

#
#       Mechanism for refreshing the dependancies
#
depend:
        copy makefile makefile.old
        sed "/^# Dependencies follow/,/^# see depend: above/D" makefile.old > makefile
        echo # Dependencies follow >> makefile
        includes $(AINC) *.asm >> makefile
        includes $(CINC) *.c   >> makefile
        echo # IF YOU PUT STUFF HERE IT WILL GET BLASTED >> makefile
        echo # see depend: above >> makefile

# DO NOT DELETE THE FOLLOWING LINE
# Dependencies follow
bidaer.obj bidaer.lst: bidaer.asm ../../laddrinc/aep.inc \
        ../../laddrinc/dcb.inc ../../laddrinc/defs.inc \
        ../../laddrinc/devdrv.inc ../../laddrinc/ilb.inc \
        ../../laddrinc/isp.inc ../../laddrinc/med.inc ../../laddrinc/rcb.inc \
        ../../laddrinc/rlh.inc ../../laddrinc/scsidefs.inc \
        ../../laddrinc/sgd.inc ../../laddrinc/srb.inc bidsegs.inc

bidcomp.obj bidcomp.lst: bidcomp.asm ../../laddrinc/dcb.inc \
        ../../laddrinc/ddb.inc ../../laddrinc/defs.inc ../../laddrinc/rcb.inc \
        ../../laddrinc/srb.inc bidsampl.inc bidsegs.inc

bidinit.obj bidinit.lst: bidinit.asm ../../laddrinc/aep.inc \
        ../../laddrinc/ddb.inc ../../laddrinc/defs.inc \
        ../../laddrinc/devhlp.inc ../../laddrinc/ilb.inc \
        ../../laddrinc/isp.inc ../../laddrinc/srb.inc bidsampl.inc \
        bidsegs.inc

bidintr.obj bidintr.lst: bidintr.asm ../../laddrinc/ddb.inc \
        ../../laddrinc/defs.inc ../../laddrinc/devhlp.inc \
        ../../laddrinc/ilb.inc ../../laddrinc/srb.inc bidsampl.inc \
        bidsegs.inc

bidreq.obj bidreq.lst: bidreq.asm ../../laddrinc/dcb.inc \
        ../../laddrinc/ddb.inc ../../laddrinc/defs.inc ../../laddrinc/ilb.inc \
        ../../laddrinc/rcb.inc ../../laddrinc/srb.inc bidsampl.inc \
        bidsegs.inc
```

```
bidstart.obj bidstart.lst: bidstart.asm ../../laddrinc/defs.inc \
        ../../laddrinc/devdrv.inc ../../laddrinc/devhlp.inc \
        ../../laddrinc/drp.inc ../../laddrinc/ilb.inc \
        ../../laddrinc/iosdefs.inc bidinfo.inc bidsegs.inc

bidsubr.obj bidsubr.lst: bidsubr.asm ../../laddrinc/defs.inc bidsegs.inc

# IF YOU PUT STUFF HERE IT WILL GET BLASTED
# see depend: above
```

```
LIBRARY SAMPLBID
PROTMODE
```

Samplbid.def

```
bidstart bidaer bidinit bidintr bidreq bidcomp bidsubr
samplbid.sys
samplbid.map /map /noignorecase
..\..\laddrlib\doscalls.lib
samplbid.def;
```

SAMPLBID.LNK

```
        page    ,132
        title   BIDAER - OS/2 1.21 sample LADDR compliant BID
        name    BIDAER
;
;******************************************************************************
;                                                                             *
; Copyright (c) Microsoft Corporation 1990                                    *
; All Rights Reserved                                                         *
;                                                                             *
;******************************************************************************
;
; this module contains the asynchronous event routines
;
;
.286

.xlist
                include bidsegs.inc     ;code and data segment definitions
                include bpb.inc
                include defs.inc
                include devdrv.inc      ;OS/2 device driver definitions
                include ilb.inc         ;IOS linkage block
                include aep.inc         ;Async event packet
                include isp.inc         ;IOS Service Block
                include med.inc         ;Memory element descriptors
                include dcb.inc         ;Device Control Block
                include rcb.inc         ;Request control block
                include sgd.inc
                include rlh.inc         ;Request packet definitions
                include scsidefs.inc    ;SCSI definitions
                include srb.inc         ;SCSI request block
.list

; code in bidreq.asm

        extrn   BID_Request:near
        extrn   SCSI_Req:near

; code in bidinit.asm

        extrn   Identify_Bus_IF:near
        extrn   Init_Bus_IF:near
        extrn   Get_Bid_Spec_Length:near

bid_aer_frame   struc

                dd      ?               ; callers 16:16 return address.
b_a_aep_off     dd      ?               ;
b_a_aep_seg     dw      ?               ; 16:32 pointer to aep.

bid_aer_frame   ends

Code            segment

                assume  cs:CodeGroup,ds:DataGroup,es:nothing

;******************************************************************************
;                                                                             *
; Async_Request(*AEP) - Asynchronous Event Handler                            *
;                                                                             *
;       Async_Request basically jump to the appropriate routine               *
;       indicated by the AER command in the Async Event Packet (AEP).         *
;                                                                             *
;       note that all registers are volatile - the caller is                  *
;       responsible for saving and restoring any registers that               *
```

```
;       must be preserved.                                                  *
;                                                                           *
;       Entry:  16:32 pointer to AEP on stack                               *
;                                                                           *
;       Exit:                                                               *
;                                                                           *
;***************************************************************************

                public  async_request
async_request   proc    far

;
; set up the pointer to the stack frame, point to the aep, obtain the
; function code from it, call the appropriate function handler, and
; return to our caller
;

        mov     bp,sp                   ; set up pointer to stack frame.

        mov     es,[bp.b_a_aep_seg]     ; point to the aep
        mov     bx,[bp.b_a_aep_off.lo]  ;

        mov     si,es:[bx.AEP_Func]     ; pick up the aep's function code.

        cmp     si,max_func             ; is it legal?
        ja      bad_aer_cmd             ; no go report the error.

        shl     si,1                    ; convert function code to an index
        call    cs:[si+offset func_table] ; and go to the appropriate routine.

        ret                             ; return to our caller.

func_table      label   word

        dw      offset init_bi          ; 0 initialize bus interface
        dw      offset init_bi          ; 1 initialize bus interface
        dw      offset config_dcb       ; 2 configure device
        dw      offset config_dcb       ; 3 configure device
        dw      offset RCB_timeout      ; 4 handle RCB timeout
        dw      offset config_dcb       ; 5 update device
        dw      offset device_Inquiry   ; 6 device inquiry

max_func        equ     ($-func_table)/2  ; maximum supported function.

;
; report that the function code is not supported by this bid
;

bad_aer_cmd:

        int     3
        ret

async_request   endp

page

;***************************************************************************
;                                                                           *
; Init_BI - initialize bus interface                                        *
;                                                                           *
;       Entry:  ES:BX -> Async Event Packet                                 *
;                                                                           *
;       Exit:                                                               *       *
;                                                                           *
```

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

        public  init_bi
init_bi proc    near

;
; call a routine "Identify_Bus_IF"
; to determine the size of the ddb
;

        call    Identify_Bus_IF         ; Size of DDB returned in AX
;
; call IOS services to create our Driver Interface Block (ddb)
;

        sub     sp,size ISP_ddb_create  ; allocate isp from the stack.

        mov     di,sp                   ; point to the gotten isp.

        mov     ss:[di].ISP_func,ISP_create_ddb  ; construct
        mov     ss:[di].ISP_owner.segmt,cs       ; isp for
        mov     ss:[di].ISP_owner.offst,offset $ ; the build
        mov     ss:[di].ISP_ddb_size,ax          ; ddb service.

        push    ss      ; ptr to
        push    di      ; packet.

        call    cs:[ILB_Service_rtn] ; create ddb.

        mov     ax,ss:[di.ISP_ddb_ptr.loword]

        add     sp,size ISP_ddb_create+4 ; cleanup the stack and
                                         ; discard the isp.

;
; Call routine to initialize the adapter with
; a pointer to the aep on the stack
;

        push    es                      ; Pass *AEP on stack
        push    0                       ; 16:32 format
        push    bx                      ;
        push    0                       ; Pass 32 bit offset to DDB on stack
        push    ax
        call    Init_Bus_IF             ; initialize bus interface.
        add     sp,6+4                  ; clean up the stack.

        ret                             ; return to our caller.

init_bi endp

page

;*****************************************************************************
;                                                                            *
; Config_DCB - Configure a device                                            *
;                                                                            *
;       Entry:  ES:BX -> Async Event Packet                                  *
;                                                                            *
;       Exit:                                                                *       *
;                                                                            *
;*****************************************************************************

                public  config_DCB
Config_DCB      proc    near
```

```
        ;get DCB pointer from AER packet

        mov     di,es:[bx].AEP_i_d_dcb.loword   ;DS:DI -> DCB
        mov     ds,cs:[ILB_ios_mem_sel]

        ;Display dcb information to screen

        mov     dl,[di].DCB_Vendor_Id[29]       ;Store byte following Inquiry data
        add     di,DCB_Vendor_Id        ;DS:SI Vendor Id Info
        mov     byte ptr [di].29,0      ;and temporarily use as null terminator
        push    ds
        push    di
        sub     di,DCB_Vendor_Id        ;Push pointer to Inquiry string

        xor     ah,ah
        mov     al,[di].DCB_SCSI_LUN    ;Insert LUN in string
        mov     cx,offset Digit_Table
        add     cx,ax
        push    cs
        push    cx

        mov     al,[di].DCB_scsi_target ;Insert SCSI Id in string
        mov     cx,offset Digit_Table
        add     cx,ax
        push    cs
        push    cx

        push    cs
        push    offset Inq_Message
        call    cs:[ILB_dprintf_rtn]
        add     sp,16
        mov     [di].DCB_Vendor_Id[29],dl

        ;KLUDGE KLUDGE KLUDGE KLUDGE

        mov     [di].DCB_bpb.BPB_SPT,32 ;32 sectors per track
        mov     [di].DCB_bpb.BPB_head_cnt,64
        mov     [di].DCB_apparent_cyl_cnt.lo,40
        mov     [di].DCB_apparent_cyl_cnt.hi,0

        ;update BID specific length field in DCB

        call    Get_BID_Spec_Length
        mov     [di].DCB_BID_Spec_Area_Len,ax

;
; put an entry in the calldown table for us
;
        sub     sp,size ISP_calldown_insert ; allocate isp from the stack.

        mov     si,sp                   ; point to the gotten isp.

        mov     ss:[si].ISP_func,ISP_INSERT_CALLDOWN ;
        mov     ss:[si].ISP_owner.segmt,cs           ;
        mov     ss:[si].ISP_owner.offst,offset $     ;

        mov     ss:[si.ISP_i_cd_dcb.hi],0  ; put the address of our
        mov     ss:[si.ISP_i_cd_dcb.lo],di ; dcb in the isp.

        mov     ss:[si.ISP_i_cd_req.segmt],cs ; put our request routine addr
        mov     ss:[si.ISP_i_cd_req.offst],offset bid_request ; in the isp.

        mov     ss:[si.ISP_i_cd_aer.segmt],cs ; put the address of our aer
        mov     ss:[si.ISP_i_cd_aer.offst],offset async_request ; in the isp.
```

```
        mov     ax,es:[bx.AEP_DDB.lo]       ; put the address
        mov     ss:[si.ISP_i_cd_ddb.lo],ax ; of our ddb in
        mov     ss:[si.ISP_i_cd_ddb.hi],0  ; the isp.

        mov     ss:[si.ISP_i_cd_flags],DCB_dmd_srb_cdb+DCB_dmd_physical ; put
                                            ; our demand bits in the isp.

        push    ss                          ; put the 16:16 pointer to
        push    si                          ; the isp on the stack.

        call    cs:[ILB_Service_rtn]    ; put our entry in the calldown table.

        add     sp,size ISP_calldown_insert+4 ; cleanup the stack and
                                            ; discard the isp.

        ret                                 ; return to our caller.

CodeInit        segment
        Inq_Message     db      'SCSI ID %c LUN %c -> %s',CAR_RET,LINE_FEED,0
Public Digit_Table
        Digit_Table     db      '0','1','2','3','4','5','6','7','8','9'
CodeInit        ends

Config_DCB      endp

page

;*****************************************************************************
;                                                                            *
; Device_Inquiry - Return Device Info/Status                                 *
;                                                                            *
;       Entry:  ES:BX -> Async Event Packet                                  *
;                                                                            *
;       Exit:                                                                *       *
;                                                                            *
;*****************************************************************************

                public  Device_Inquiry
Device_Inquiry  proc    near


;
; Allocate and RCB and an SRB to send the Inquiry request
; through our the request path.
;
        ;Allocate an RCB

        sub     sp,size ISP_rcb_alloc   ; allocate isp from the stack.
        mov     di,sp                   ; point to the gotten isp.

        mov     ss:[di].ISP_func,ISP_create_rcb  ; construct
        mov     ss:[di].ISP_owner.segmt,cs       ; isp for
        mov     ss:[di].ISP_owner.offst,offset $ ; the build
        mov     ss:[di].ISP_rcb_size,size rcb

        push    ss      ; ptr to
        push    di      ; packet.

        call    cs:[ILB_Service_rtn] ; create srb.

        mov     ax,ss:[di.ISP_rcb_ptr.loword]

        add     sp,size ISP_rcb_alloc+4 ; cleanup the stack and
                                        ; discard the isp.
```

```
        mov     si,ax                   ;DS:SI -> RCB

        ;Put DCB pointer into RCB

        mov     ax,es:[bx].AEP_i_d_dcb.loword
        mov     ds,cs:[ILB_ios_mem_sel]
        mov     [si].RCB_physical_dcb.loword,ax

        ;Allocate an SRB

        sub     sp,size ISP_srb_alloc   ; allocate isp from the stack.
        mov     di,sp                   ; point to the gotten isp.

        mov     ss:[di].ISP_func,ISP_create_srbs ; construct
        mov     ss:[di].ISP_owner.segmt,cs       ; isp for
        mov     ss:[di].ISP_owner.offst,offset $ ; the build
        mov     ss:[di].ISP_srb_number,1

        call    Get_BID_Spec_Length
        mov     dx,ax                   ;Store away BID specific length
        add     ax,size SRB             ;Set SRB size with Private data
        add     ax,14                   ;Make room for Sense Data
        add     ax,size SGD             ;Make room for a SG descritpor
        mov     ss:[di].ISP_srb_size,ax

        push    ss      ; ptr to
        push    di      ; packet.

        call    cs:[ILB_Service_rtn] ; create srb.

        mov     ax,ss:[di.ISP_srb_ptr.loword]

        add     sp,size ISP_srb_alloc+4 ; cleanup the stack and
                                        ; discard the isp.

        mov     di,ax                   ;DS:DI -> SRB
        mov     [si].RCB_SRB_Logical.loword,ax

;
; We now have an RCB pointing to both a DCB and an SRB
; The request must now be built by filling in any fields
; needed by the SCSI state machine to process the request.
;
; The SRB has room at the end of it for a SG descriptor as
; well as for a temporary INQUIRY data buffer.
;
;
;
;       DS:SI -> RCB
;       DS:DI -> SRB
;       ES:BX -> AEP packet
;

        mov     [di].SRB_Request_State,SRB_Processing
        mov     [di].SRB_Next_Sortable.loword,0
        mov     [di].SRB_Next_NonSort.loword,0

        ;Setup Sense buffer pointer

        mov     ax,di                           ;AX SRB logical pointer
        add     ax,size SRB                     ;adjust past SRB
        add     ax,dx                           ;adjust past BID private area

        mov     cx,[di].SRB_Phys_Addr.Lo
```

```
add     cx,size SRB                     ;adjust past SRB
add     cx,dx                           ;adjust past BID private area
mov     dx,[di].SRB_Phys_Addr.Hi        ;DX:CX physical address of SRB
adc     dx,0

mov     [di].SRB_Sense_Buffer_Physical.Hi,dx
mov     [di].SRB_Sense_Buffer_Physical.Lo,cx

;Setup Scatter/Gather Descriptor

add     ax,14                           ;adjust past Sense
mov     [di].SRB_SG_Sel,ds
mov     [di].SRB_SG_Off.loword,ax

add     cx,14
adc     dx,0
mov     [di].SRB_SG_Physical.Hi,dx
mov     [di].SRB_SG_Physical.Lo,cx

;Put address of Inquiry buffer in DCB in SG descriptor

push    di
push    si
mov     si,es:[bx].AEP_i_d_dcb.loword   ;DS:SI -> DCB
mov     di,ax                           ;DS:DI -> SG descriptor
mov     dx,[si].DCB_Phys_Addr.Hi
mov     cx,[si].DCB_Phys_Addr.Lo
add     cx,DCB_inquiry_flags
mov     [di].SG_buff_ptr.Hi,dx
mov     [di].SG_buff_ptr.Lo,cx
mov     [di].SG_buff_size.Hi,0
mov     [di].SG_buff_size.Lo,DCB_INQ_DATA_LENGTH
pop     si
pop     di

mov     [di].SRB_RCB_Logical.loword,si
mov     [di].SRB_Function,SCSI_IO

mov     [di].SRB_Data_Xfer_Length.lo,DCB_INQ_DATA_LENGTH
mov     [di].SRB_Data_Xfer_Length.hi,0
mov     [di].SRB_Sense_Buffer_Length,14
mov     [di].SRB_CDB_Length,6           ;Inquiry is a 6 byte CDB
mov     [di].SRB_Num_SG_Entries,1       ;Single Scatter/Gather entry

mov     [di].SRB_CDB[0],SCSI_INQUIRY
mov     [di].SRB_CDB[2],0
mov     [di].SRB_CDB[3],0
mov     [di].SRB_CDB[4],DCB_INQ_DATA_LENGTH

push    es
push    bx
push    di

mov     dx,es:[bx].AEP_DDB.loword               ;DS:DX -> DDB
mov     si,[di].SRB_RCB_Logical.loword
mov     bx,[si].RCB_physical_dcb.loword         ;DS:BX -> DCB

mov     al,[bx].DCB_SCSI_LUN    ;Setup LUN in CDB
shl     al,5
mov     [di].SRB_CDB[1],al

push    cs              ;treat like a far call
call    SCSI_Req

pop     di
```

```
        pop     bx
        pop     es

        ;Wait for SRB to complete

SRB_Spin:
        test    [di].SRB_Request_State,SRB_Done
        jz      SRB_Spin

        cmp     [di].SRB_Request_State,SRB_Done+SRB_No_Error
        jne     Not_Present

        ;Device is present and Inquiry was successful
        ;Move Inquiry data into AEP and set Device Present in AEP

        mov     es:[bx].AEP_i_d_status,AEP_REAL_INQ_DATA
        jmp     Dev_Inq_Ret

Not_Present:

        mov     es:[bx].AEP_i_d_status,AEP_NO_INQ_DATA

Dev_Inq_Ret:

        ;Deallocate RCB and SRB

        mov     ax,[di].SRB_RCB_Logical.loword

        sub     sp,size ISP_rcb_dealloc ; allocate isp from the stack.
        mov     di,sp                   ; point to the gotten isp.

        mov     ss:[di].ISP_func,ISP_dealloc_rcb ; construct
        mov     ss:[di].ISP_owner.segmt,cs       ; isp
        mov     ss:[di].ISP_owner.offst,offset $ ;
        mov     ss:[di].ISP_rcb_ptr_da.loword,ax
        mov     ss:[di].ISP_rcb_ptr_da.hiword,0

        push    ss      ; ptr to
        push    di      ; packet.

        call    cs:[ILB_Service_rtn] ; create srb.

        add     sp,size ISP_rcb_dealloc+4 ;cleanup the stack and
                                          ;discard the isp.

        ret

Device_Inquiry  endp

page

;*************************************************************************
;                                                                        *
; RCB_Timeout - Timeout Handler                                          *
;                                                                        *
;       Entry:  ES:BX -> Async Event Packet                              *
;                                                                        *
;       Exit:                                                            *       *
;                                                                        *
;*************************************************************************

        Public  RCB_Timeout
RCB_Timeout     proc    near

        ret                             ; return to our caller.
```

```
RCB_Timeout     endp

Code    ends

end
```

```
        page    ,132
        title   BIDCOMP - OS/2 1.21 sample LADDR compliant BID
        name    BIDCOMP
;
;*****************************************************************************
;                                                                             *
; Copyright (c) Microsoft Corporation 1990                                    *
; All Rights Reserved                                                         *
;                                                                             *
;*****************************************************************************
;
; this module contains the completion routine
;
;
.286

.xlist
                include bidsegs.inc     ; code and data segment definitions.
                include bpb.inc
                include dcb.inc         ; device control block.
                include ddb.inc         ; driver data block header.
                include defs.inc        ; common definitions.
                include rcb.inc         ; request control block.
                include srb.inc         ; scsi request block.
                include bidsampl.inc    ; bid specific definitions.
.list

; code in bidintr.asm

        extrn   send_srb:near

; code in bidreq.asm

        extrn   srb_dispatch:near

Code            segment

                assume  cs:CodeGroup,ds:nothing,es:nothing

;
;*****************************************************************************
;                                                                             *
; srb_completion - handle srb completion                                      *
;                                                                             *
; on entry:     ds:si -> ddb                                                  *
;               ds:di -> srb that is completing                               *
;                                                                             *
;*****************************************************************************
;
                public  srb_completion
srb_completion  proc    near
;
; add code to move scsi status to ds:[di.SRB_Status.SCSI_Status]
;



;
; add code to check for a check condition
;



;
; add code to if a check condition exist, move sense data to sense buffer
; (which normally follows the bid specific area of the srb)
;
```

```
;
; add code to move hba status to ds:[di.SRB_Status.Adapter_Status]
;


;
; add code to set up request state in ds:[di.SRB_request_state]. typical
; values are "SRB_DONE+SRB_NO_ERROR" and "SRB_DONE+SRB_ERROR"
;


;
; add code to start the next srb with a call to "send_next_srb"
;


;
; perform completion callback on srb
;

        test    ds:[di.SRB_flags_long.lo],SRB_DISABLE_CALLBACK ; is srb callback enabled?
        jz      Completion_Return               ; no, bypass doing it.

        push    si                              ; yes, save our ddb pointer.

        push    0                               ; put a 32-bit pointer to
        push    di                              ; the srb on the stack.

        call    ds:[di.SRB_callback]            ; do the srb callback.

        add     sp,4                            ; unstack the srb pointer.

        pop     si                              ; recover our ddb pointer.

callback_done:

        or      ax,ax                           ; did the callout routine
                                                ; give us back an srb?

        jz      completion_return               ; no, go return.

        mov     di,ax                           ; yes, point to it.

        mov     bx,ds:[di.SRB_rcb_logical.lo]   ; point to
        mov     bx,ds:[bx.RCB_physical_dcb.lo]  ; its dcb.

        mov     dx,si                           ; point to our ddb.

        push    cs                              ; synthesize a far call
        call    SRB_dispatch                    ; to our srb dispatcher.

completion_return:

        ret                                     ; return to our caller.

SRB_completion endp

;
;*****************************************************************************
;                                                                            *
; send_next_srb - start the next srb on the device queue, if there is one    *
;                                                                            *
; on entry:     ds:si -> ddb                                                 *
```

```
;               ds:di -> srb that has completed                         *
;                                                                       *
;***********************************************************************
;
                public  send_next_srb
send_next_srb   proc    near

;
; dispatch next non-sortable srb if there is no error
;

        cmp     ds:[di.SRB_next_nonsort.lo],0 ; is there a non-sortable?
        je      dispatch_sortable             ; no, go try sortables.

        cmp     ds:[di.SRB_request_state],SRB_DONE+SRB_ERROR ; yes, any error?
        je      dispatch_sortable             ; yes, go try sortable.

        mov     di,ds:[di.SRB_next_nonsort.lo] ; no, point to next srb
        call    send_srb                      ; and start it.

;
; start next sortable srb
;

dispatch_sortable:

        mov     bx,ds:[di.SRB_rcb_logical.lo]     ; point to
        mov     bx,ds:[bx.RCB_physical_dcb.lo]    ; the dcb.

        DISABLE                                   ; lockout conflicting code.

        mov     di,ds:[bx.DCB_work_queue.lo] ; get first queued srb.

        or      di,di                         ; is there one?
        je      queue_empty                   ; no, go show device idle.

        mov     ax,ds:[di.SRB_next_sortable.lo] ; yes, remove it from
        mov     ds:[bx.DCB_work_queue.lo],ax ; the pending queue.

        ENABLE                                ; enable other code.

        call    send_srb                      ; start the request.

        ret                                   ; return to our caller.

queue_empty:

        and     ds:[bx.DCB_device_status],not ACTIVE ; show device is idle.

        ENABLE                                ; enable other code.

        ret                                   ; return to our caller.

send_next_srb   endp

Code            ends

                end
```

```
        page    ,132
        title   BIDINIT - OS/2 1.21 sample LADDR compliant BID
        name    BIDINIT
;
;******************************************************************************
;                                                                              *
; Copyright (c) Microsoft Corporation 1990                                     *
; All Rights Reserved                                                          *
;                                                                              *
;******************************************************************************
;
; this module contains the bus interface initialization routine
;
;
.286

.xlist
                include aep.inc         ; Async event packet.
                include bidsegs.inc     ; code and data segment definitions.
                include ddb.inc         ; driver data block header.
                include defs.inc        ; common definitions.
                include devhlp.inc      ; OS/2 Device Help definitions.
                include ilb.inc         ; IOS Linkage Block.
                include isp.inc         ; IOS Service packet(s).
                include srb.inc         ; scsi request block.
                include bidsampl.inc    ; bid specific definitions.
.list

; code in bidsubr.asm

        extrn   Get_Options:near
        extrn   Wait_Idle:near

; code in bidintr.asm

        extrn   HA_Interrupt:near

INIT_Stack      struc

IBI_BP          dw      ?       ;from push bp
IBI_Ret         dw      ?       ;return address
DDB_Off_Lo      dw      ?
DDB_Off_Hi      dw      ?
AEP_Off_Lo      dw      ?
AEP_Off_Hi      dw      ?
AEP_Sel         dw      ?

INIT_Stack      ends

DataInit        segment

                public  Timer_Status                ; should be in biddata.asm
Timer_Status    DB      ?       ;Timer Status Flag <-------------------------

DataInit        ends

CodeInit        segment

                assume  cs:CodeGroup,ds:nothing,es:DataGroup

;******************************************************************************
;                                                                             *
; Init_Bus_IF - Bus Interface Initialization                                  *                    *
;                                                                             *
;       Entry:  *DDB,*AEP passed on stack                                     *
```

```
;                                                                       *
;       Exit:                                                           *
;                                                                       *
;************************************************************************

        Public  Init_Bus_IF
Init_Bus_IF     proc    near

        ;get DDB pointer off stack

        push    bp
        mov     bp,sp                           ;Get a stack frame pointer
        mov     si,[bp].DDB_Off_Lo              ;DS:SI -> DDB
        mov     ds,cs:[ILB_ios_mem_sel] ;our ddb for display.
        mov     di,[bp].AEP_Off_Lo              ;ES:DI -> AEP
        mov     es,[bp].AEP_Sel
        pop     bp

        ;Get adapter configuration from option string

        add     di,AEP_bi_i_option      ;ES:DI -> Option string
        call    Get_Options

        ;Setup adapter based on configuration information

        mov     es,cs:[ilb_drv_data_sel]
        push    ax
        call    Adapter_Setup
        pop     ax
        ret

Init_Bus_IF     endp

;************************************************************************
;                                                                       *
; Identify_Bus_IF - Determine type of Bus Interface                     *
;                                                                       *
;       Entry:  *AEP passed on stack                                    *
;                                                                       *
;       Exit:   AX - Required size of DDB                               *
;                                                                       *
;************************************************************************

        Public Identify_Bus_IF
Identify_Bus_IF proc    near

        mov     ax,size sample_bid_ddb
        ret

Identify_Bus_IF endp

;************************************************************************
;                                                                       *
; Get_BID_Spec_Length - Get length of BID specific area in SRB          *
;                                                                       *
;       Entry:                                                          *
;                                                                       *
;       Exit:   AX - Required size of DDB                               *
;                                                                       *
;************************************************************************

        Public Get_BID_Spec_Length
Get_BID_Spec_Length     proc    near

        mov     ax,((size sample_bid_srb) - (size SRB))
```

```
        ret

Get_BID_Spec_Length     endp

page

;*****************************************************************************
;                                                                             *
; Adapter_Setup - Setup Adapter based on settings in DDB                      *
;                                                                             *
;       Entry:  DS:SI -> DDB                                                  *
;                                                                             *
;       Exit:                                                                 *
;                                                                             *
;*****************************************************************************

        Public Adapter_Setup
Adapter_Setup   proc    near

        ;Wait for Adapter Idle

Wait_For_Idle:

        call    Wait_Idle               ;Wait for adapter idle
        jnc     Adapter_Idle
        jmp     Setup_Error             ;quit if timeout occurred

Adapter_Idle:

;
; add code to get config info from adapter if necessary
;


;
; turn interrupt handler on
;

        call    Setup_IRQ
        jc      Setup_Error

        jmp     Setup_Done

Setup_Error:

        ;Indicate error to IOS somehow <-------------------------------------
        ;and continue

Setup_Done:

        call    Timer_Rel               ;Release Timer services
        ret

Adapter_Setup   endp

page

;*****************************************************************************
;                                                                             *
; Setup_IRQ - Setup Interrupt Request Handler Registration                    *
;                                                                             *
;       Entry:  DS:SI -> Driver information block (DDB)                       *
;                                                                             *
;       Exit:   C set if failed                                               *
;                                                                             *
```

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

        Public Setup_IRQ
Setup_IRQ       proc    near

        ;call IOS to register IRQ

        push    bp
        sub     sp,size ISP_IRQ_set     ;allocate isp from the stack.

        mov     bp,sp                   ;point to the isp
        mov     [bp].ISP_func,ISP_set_irq
        mov     al,[si].interrupt_level
        mov     [bp].ISP_IRQ_LEVEL,al
        mov     [bp].ISP_Share_Flag,0           ;Non-shared interrupt

        ;setup IRQ Handler so that DS:SI-> DDB on entry

        mov     word ptr [bp].ISP_IRQ_Data.segmt, ds
        mov     word ptr [bp].ISP_IRQ_Data.offst, si
        mov     [bp].ISP_IRQ_Handler.segmt, cs
        mov     [bp].ISP_IRQ_Handler.offst,offset HA_Interrupt

        push    ss                      ;Put pointer to ISP on stack
        push    bp

        call    cs:[ILB_Service_rtn]

        add     sp,size ISP_IRQ_set+4 ; cleanup the stack

        pop     bp
        ret

Setup_IRQ       endp

page

;*****************************************************************************
;                                                                            *
; Setup_Timer - Setup Timer for Initialization Timeout function             *
;                                                                            *
;       Entry:  CX - Timer duration in Milliseconds                          *
;                                                                            *
;       Exit:   C set if failed                                              *
;                                                                            *
;*****************************************************************************

        Public Setup_Timer
Setup_Timer     proc    near

        push    dx
        push    bx
        push    ds
        mov     es:Timer_Status,not TIMEOUT     ;Clear Timeout condition
        push    es                          ;Set DS to header segment
        pop     ds
        mov     ax,offset Timer_Routine  ;Request interrupt after cx ticks
        mov     bx,cx
        mov     dl,DevHlp_TickCount
        call    cs:[ILB_DevHlp]
        pop     ds
        pop     bx
        pop     dx
        ret                             ;return with status in carry bit
```

```
Setup_Timer     endp

;************************************************************************
;                                                                       *
; Timer_Rel - Release Initialization Timeout function                   *
;                                                                       *
;       Entry:                                                          *
;                                                                       *
;       Exit:   C set if failed                                         *
;                                                                       *
;************************************************************************

        Public Timer_Rel
Timer_Rel proc near

        push    dx
        push    ds
        push    es              ;Set DS to Header segment
        pop     ds
        mov     ax,offset Timer_Routine
        mov     dl,DevHlp_ResetTimer
        call    cs:[ILB_DevHlp]
        pop     ds
        pop     dx
        ret                             ;return with status in carry bit

Timer_Rel endp

;************************************************************************
;                                                                       *
; Timer_Routine - Timer Handler for Initialization                      *
;                                                                       *
;       Entry:  Interrupt Time                                          *
;                                                                       *
;       Exit:                                                           *
;                                                                       *
;************************************************************************

        Public  Timer_Routine
Timer_Routine   proc    far
assume  ds:DataGroup,es:nothing

        mov     Timer_Status,TIMEOUT    ;Indicate Timer Expired
        ret

assume  ds:nothing,es:DataGroup

Timer_Routine    endp

CodeInit        ends

end
```

```
        page    ,132
        title   BIDINTR - OS/2 1.21 sample LADDR compliant BID
        name    BIDINTR
;
;*****************************************************************************
;                                                                             *
; Copyright (c) Microsoft Corporation 1990                                    *
; All Rights Reserved                                                         *
;                                                                             *
;*****************************************************************************
;
; this module contains the bus interface interrupt handler
;
;
.286

.xlist
                include bidsegs.inc     ;code and data segment definitions
                include devhlp.inc      ;OS/2 device help definitions
                include ilb.inc         ;IOS Linkage Block
                include defs.inc        ;Common definitions
                include ddb.inc         ; driver data block header.
                include srb.inc         ; scsi request block.
                include bidsampl.inc    ; bid specific definitions.
.list

; code in bidsubr.asm

        extrn   Enable_HaInt:near
        extrn   Disable_HaInt:near
        extrn   Start_SCSI:near

; code in bidcomp.asm

        extrn   SRB_Completion:near

Code            segment
                assume  cs:CodeGroup,ds:nothing,es:nothing

;*****************************************************************************
;                                                                             *
; HA_Interrupt - Hardware Interrupt Handler                                   *
;                                                                             *
;       Entry:  DS:SI -> Driver data block                                    *
;               Interrupts Disabled                                           *
;                                                                             *
;       Exit:   Carry must be clear if interrupt was presented                *
;               by adapter identified by the DDB                              *
;                                                                             *
;       This interrupt handler does not provide for nested interrupts         *
;       on the same adapter. This situation is prevented by masking the       *
;       host adapter interrupt until the routine is ready to return.          *
;                                                                             *
;*****************************************************************************

        Public HA_Interrupt
HA_Interrupt    proc    far

        call    Disable_haint   ;Prevent further interrupts on this Adapter
        ENABLE                  ;Enable system interrupts

;
; insert code to get status from adapter and reset its pending innterrupt
; if appropriate
;
```

```
        mov     al,[si].Interrupt_level ;Issue EOI
        mov     dl,DevHlp_EOI
        call    cs:[ILB_DevHlp]

;
; insert code to process interrupt and set up registers
;

        call    SRB_Completion

        call    Enable_HaInt

        clc

        ret

HA_Interrupt endp

page

;*****************************************************************************
;                                                                            *
; Send_SRB - Send SCSI request to host adapter via the mailboxes             *
;                                                                            *
;       Entry:  DS:SI -> Driver data block                                   *
;               DS:DI -> SCSI Request Block                                  *
;                                                                            *
;*****************************************************************************

        Public Send_SRB
Send_SRB    proc    near

;
; insert code to check if srb can be started now
;


;
; insert code to set up registers
;


        call    Start_Scsi              ;Issue Start SCSI to HA

        ret

Send_SRB endp

Code ends

end
```

```
        page    ,132
        title   BIDREQ - OS/2 1.21 sample LADDR compliant BID
        name    BIDREQ
;
;******************************************************************************
;                                                                             *
; Copyright (c) Microsoft Corporation 1990                                    *
; All Rights Reserved                                                         *
;                                                                             *
;******************************************************************************
;
; this module contains the request handler routine
;
;
.286

.xlist

                include bidsegs.inc     ; code and data segment definitions
                include bpb.inc
                include dcb.inc         ; Device Control Block
                include ddb.inc         ; driver data block header.
                include defs.inc        ; General definitions
                include ilb.inc         ; IOS Linkage Block
                include rcb.inc         ; Request Control Block
                include srb.inc         ; scsi request block.
                include bidsampl.inc    ; bid specific definitions.
.list

; code in bidintr.asm

        extrn   send_srb:near

Code            segment
                assume  cs:Code,ds:nothing,es:nothing

;******************************************************************************
;                                                                             *
; BID_Request - BID Entry point                                               *
;                                                                             *
;       This is the main entry point into the BID.  SCSI requests are         *
;       passed to this entry point via a pointer to an SRB chain.             *
;                                                                             *
;       Entry:  *SRB on stack                                                 *
;                                                                             *
;       Exit:                                                                 *
;                                                                             *
;       Note:   This routine is accessed with a FAR call.                     *
;               Routine may be called in interrupt mode, so no                *
;               blocking.                                                     *
;                                                                             *
;******************************************************************************

        Public  BID_Request
BID_Request     proc    far

BR_Stack_Frame struc
        BR_BP dw ?
        BR_Ret_Address dd ?
        BR_SRB_Ptr_Off dd ?
BR_Stack_Frame ends

        push    bp
        mov     bp, sp
```

```
        mov     ds,cs:[ILB_ios_mem_sel] ;ios mem selector
        mov     di,word ptr [bp].BR_SRB_Ptr_Off.loword  ;DS:DI -> SRB

        mov     si,[di].SRB_RCB_Logical.loword
        mov     bx,[si].RCB_physical_dcb.loword         ;DS:BX -> DCB
        mov     si,[si].RCB_CalldownPtr.loword
        mov     dx,[si].DCB_CD_DDB.loword               ;DS:DX -> DDB

        pop     bp

        Public  SRB_Dispatch
SRB_Dispatch:

        ;Note: New requests at Interrupt/Completion time enter here
        ;with registers setup appropriately

        ;Do a quick check for normal function

        cmp     [di].SRB_Function,SCSI_IO
        je      SCSI_Req

        ;USe jump table to access other functions

        mov     al,[di].SRB_Function            ;Get the SRB Function
        xor     ah,ah
        cmp     ax,Req_Abort                    ;Compare to the maximum
        jg      Bad_SRB_Cmd                     ;If >, reject request
        xor     ah,ah                           ;Translate command into
        shl     ax,1                            ;word offset for jump table
        mov     si,offset BID_Table             ;Call command handler
        add     si,ax
        jmp     cs:[si]

BID_Return:

        ret

Bad_SRB_Cmd:

        int     3
        ret

BID_Request     endp

;BID Command Dispatch Table
        Public  BID_Table
BID_Table       label   word
        dw      offset  SCSI_Req                ;0 Execute SCSI I/O
        dw      offset  SCSI_Req                ;1 SCSI Device Reset
        dw      offset  SCSI_Req                ;2 SRB Abort


SUBTTL SCSI I/O Request Handler
PAGE

;*************************************************************************
;                                                                        *
; SCSI_Req - Execute SCSI I/O Request                                    *
;                                                                        *
;       Entry:  DS:DI -> SRB chain                                       *
;               DS:DX -> DDB                                             *
;               DS:BX -> DCB                                             *
;               Code may be entered in Init, Kernel or Interrupt Mode    *
;                                                                        *
;       Exit:                                                            *
```

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

        Public SCSI_Req
SCSI_Req        proc    near

        push    di              ;Store head of SRB chain

CCB_Build:

;
; add code to convert srb to adapter specific form in bid private
; area of srb
;


;Check if there is a sortable link

        cmp     [di].SRB_Next_Sortable.loword,0
        je      Check_NonSort
        mov     di,word ptr [di].SRB_Next_Sortable.loword
        jmp     CCB_Build

;Check if there is a non-sortable link

Check_NonSort:

        cmp     [di].SRB_Next_NonSort.loword,0
        je      End_Of_Chain
        mov     si,word ptr [di].SRB_Next_NonSort.loword
        jmp     CCB_Build

End_Of_Chain:

        pop     di              ;Restore head of SRB chain

;
; add code to add processed srb(s) to pending work queue anchored
; from DCB_Work_Queue
;



; Start up next request if this device is idle

        DISABLE
        test    [bx].DCB_Device_Status,ACTIVE   ;Is this device Active ?
        jnz     Already_Active
        or      [bx].DCB_Device_Status,ACTIVE   ;If not, claim semaphore

; Pull SRB at the head of the Queue

        mov     di,[bx].DCB_Work_Queue.loword
        mov     ax,[di].SRB_Next_Sortable.loword
        mov     [bx].DCB_Work_Queue.loword,ax

        ENABLE

        call    Send_SRB

Already_Active:

        ENABLE

        ret
```

```
Scsi_Req        endp

Code ends

end
```

```
        page    ,132
        title   BIDSTART - OS/2 1.21 sample LADDR compliant BID
        name    BIDSTART
;
;******************************************************************************
;                                                                             *
; Copyright (c) Microsoft Corporation 1990                                    *
; All Rights Reserved                                                         *
;                                                                             *
;******************************************************************************
;
; this module contains the start1 entry point and initialization routines
;
;
.286


        .xlist
        include bidsegs.inc     ;code and data segment definitions
        include bidinfo.inc     ;BID Identification Equates
        include defs.inc
        include devhlp.inc      ;OS/2 device help definitions
        include devdrv.inc      ;OS/2 device driver definitions
        include drp.inc         ;Device registration packet
        include ilb.inc         ;IOS linkage block
        include iosdefs.inc     ;IOS definitions
        .list

SUBTTL External Declarations
PAGE

Code    segment
        extrn   Async_Request:near
Code    ends

CodeInit segment
        extrn   DOSDevIOCTL:far
        extrn   DOSOpen:far
        extrn   DOSClose:far
CodeInit ends

SUBTTL Data Declarataions
PAGE

DataInit        segment

DevHlpFunc      DD      ?       ;Device Help function pointer

;12 byte field used for ATTACH DevHlp

IOS_Req_Real    DD      ?       ;Real Mode FAR address of IOS registration
IOS_DS_Real     DW      ?       ;Real Mode DS for IOS registration
IOS_Req_Prot    DD      ?       ;Prot Mode FAR address of IOS registration
IOS_DS_Prot     DW      ?       ;Prot Mode DS for IOS registration

;Name of the I/O Subsystem - Null terminated

IOS_Name        DB      'IOS$    ',0

;Driver registration packet

Drv_Reg_Pkt     DRP     <DRP_BID,,,,,BIDName,BIDDate,BIDTime,BIDRev,DRP_SCSI_ADDR,00h,>

;Data needed by IOCTL in Ring 3 Init
```

```
IOS_Handle      DW      ?       ;Handle for IOS$ character device
Open_Action     DW      ?       ;Result of OPEN call to get handle

DataInit        ends

Code    segment

;IOS Linkage block is located in Code segment to facilitate
;calls to IO subsystem.  Must be at offset 0 in Code segment.

ILB_Seg segment

        public  Bid_ilb
Bid_ilb         ILB     <>      ;IO Subsystem linkage block

ILB_Seg ends

Code    ends

SUBTTL Device Header
PAGE

Header segment

;*************************************************************************
;                                                                        *
; Device Header                                                          *
;                                                                        *
; Identifies device as a character device.  This character              *
; device is not functional, it merely provides a means                   *
; for the BID to get initialized.                                        *
;                                                                        *
;*************************************************************************

        Public  Next_Header,Dev_Attributes,Strat_Offset,IDC_Offset,Dev_Name
        Public  ProtMode_CS,ProtMode_DS,RealMode_CS,RealMode_DS
Next_Header     dd      -1                      ;Pointer to next device
Dev_Attributes  dw      DEVLEV_2+DEV_CHAR_DEV
Strat_Offset    dw      offset BID_strat1       ;Strategy Routine
IDC_Offset      dw      0                       ;IDC entry point
Dev_Name        db      BID_Dev_Name            ;Device Name
ProtMode_CS     dw      ?
ProtMode_DS     dw      ?
RealMode_CS     dw      ?
RealMode_DS     dw      ?
                db      8 dup (?)               ;
Header ends

SUBTTL BID Strategy Routine
PAGE

Code    segment
assume cs:CodeGroup,ds:DataGroup,es:nothing

;*************************************************************************
;                                                                        *
; BID_strat1 - BID Strategy1 Routine                                     *
;                                                                        *
;       BID_strat1 handles calls from the kernel to process request      *
;       packets. the only legal calls are for initialization requests.   *
;                                                                        *
;       Entry:  ES:BX   OS/2 Request Packet address                      *
;                                                                        *
;       Exit:   ES:BX   OS/2 Request Packet address                      *
;                       Status Byte in Packet is Updated                 *
```

```
;                                                                       *
;       Note:   All commands except INIT are returned with invalid      *
;               command error.                                          *
;                                                                       *
;************************************************************************

        public  bidstart
bidstart        proc    far

        public  BID_strat1
BID_strat1:

        mov     al,es:[bx].Pkt_BCmd     ; pick up the command.
        cmp     al,max_cmd              ; is it valid?
        ja      bad_cmd                 ; no, go report the error.

        xor     ah,ah                   ; yes, convert
        shl     ax,1                    ; it to an
        mov     si,ax                   ; index.

        call    cs:[si+offset cmd_table] ; process the command.
        jmp     strat1_done              ; go exit.

bad_cmd:
        call    not_supported           ; report the error.

strat1_done:

        ret                             ; return to our caller.

cmd_table       label   word

        dw      offset  bid_init_3      ; 00 - steady state (ring 3) init.
        dw      offset  not_supported   ; 01 - command is not supported.
        dw      offset  not_supported   ; 02 - command is not supported.
        dw      offset  not_supported   ; 03 - command is not supported.
        dw      offset  not_supported   ; 04 - command is not supported.
        dw      offset  not_supported   ; 05 - command is not supported.
        dw      offset  not_supported   ; 06 - command is not supported.
        dw      offset  not_supported   ; 07 - command is not supported.
        dw      offset  not_supported   ; 08 - command is not supported.
        dw      offset  not_supported   ; 09 - command is not supported.
        dw      offset  not_supported   ; 0a - command is not supported.
        dw      offset  not_supported   ; 0b - command is not supported.
        dw      offset  not_supported   ; 0c - command is not supported.
        dw      offset  not_supported   ; 0d - command is not supported.
        dw      offset  not_supported   ; 0e - command is not supported.
        dw      offset  not_supported   ; 0f - command is not supported.
        dw      offset  not_supported   ; 10 - generic ioctl.
        dw      offset  not_supported   ; 11 - command is not supported.
        dw      offset  not_supported   ; 12 - command is not supported.
        dw      offset  not_supported   ; 13 - command is not supported.
        dw      offset  not_supported   ; 14 - command is not supported.
        dw      offset  not_supported   ; 15 - command is not supported.
        dw      offset  not_supported   ; 16 - command is not supported.
        dw      offset  not_supported   ; 17 - command is not supported.
        dw      offset  not_supported   ; 18 - command is not supported.
        dw      offset  not_supported   ; 19 - command is not supported.
        dw      offset  not_supported   ; 1a - command is not supported.
        dw      offset  bid_init_0      ; 1b - boot time (ring 0) init.
        dw      offset  shutdown        ; 1c - prepare for shutdown.

max_cmd         equ     ($-cmd_table-2)/2

bidstart        endp
```

```
not_supported   proc    near

        int     3
        mov     es:[bx].Pkt_fsStatus,BAD_DRV_REQ or STAT_ERROR or STAT_DONE ;
                                        ; store the error code in the packet.
        ret                             ; return to our caller.

not_supported   endp

shutdown        proc    near

        cmp     byte ptr es:[bx+13],0   ; is is begin shutdown?
        jne     shutdown_complete       ; no, go process complete shutdown.
        mov     es:[bx].Pkt_fsStatus,STAT_DONE ; mark the command as done.
        ret                             ; return to our caller.

shutdown_complete:

        mov     es:[bx].Pkt_fsStatus,STAT_DONE ; mark the command as done.
        ret                             ; return to our caller.

shutdown        endp

SUBTTL Ring 0 BID Initialization
PAGE

;*************************************************************************
;                                                                        *
; BID_Init_0 - Ring 0 BID Initialization                                 *
;                                                                        *
;       BID_Init basically registers with the IOS and based on the       *
;       return code from IOS leaves all, some or none of the             *
;       driver resident.  During the registration call, the IOS may      *
;       call the BIDs Async Event Routine (AER).                         *
;                                                                        *
;       Entry:  ES:BX   OS/2 Request Packet address                      *
;                                                                        *
;       Exit:   ES:BX   OS/2 Request Packet address                      *
;                                                                        *
;       Note:   Called at Ring 0 (Base Driver).                          *
;                                                                        *
;*************************************************************************
bid_init        proc    near

        assume  cs:CodeGroup,ds:DataGroup,es:nothing

        Public  bid_init_0
bid_init_0:

        ;Save the DevHlp address

        mov     ax,word ptr es:[bx].INI_pDevHlp
        mov     word ptr DevHlpFunc,ax
        mov     ax,word ptr es:[bx].INI_pDevHlp+2
        mov     word ptr DevHlpFunc+2,ax

        call    DspBanner               ;Display ASCII message

        ;Use Attach to get IOS registration entry point

        push    bx
        mov     bx,offset IOS_Name
        mov     di,offset IOS_Req_Real
        mov     dl,DevHlp_AttachDD
```

```
        call    [DevHlpFunc]
        pop     bx
        jnc     Attach_OK               ;If IOS not found for some reason
        jmp     BID_Abort               ;just abort the driver

Attach_OK:

        call    Setup_DRP               ;Setup Driver Registration packet

        push    es                      ;Store es
        push    ds                      ;Store ds
        push    bx

        push    es                      ;pass pointer to the Init packet
        push    bx
        push    ds                      ;pass pointer to driver registration
        push    offset Drv_Reg_Pkt      ;packet (DRP)
        call    [IOS_Req_Prot]          ;call registration
        add     sp,08                   ;Clean up stack
        pop     bx
        pop     ds                      ;Restore ds
        pop     es                      ;Restore es

        jmp     Check_Reg_Status        ;Check registration status

SUBTTL Ring 3 BID Initialization
PAGE

;*****************************************************************************
;                                                                            *
; BID_init_3 - Ring 3 BID Initialization                                     *
;                                                                            *
;       BID_Init basically registers with the IOS and based on the           *
;       return code from IOS leaves all, some or none of the                 *
;       driver resident.  During the registration call, the IOS may          *
;       call the BIDs Async Event Routine (AER).                             *
;                                                                            *
;       Entry:  ES:BX   OS/2 Request Packet address                          *
;                                                                            *
;       Exit:   ES:BX   OS/2 Request Packet address                          *
;                                                                            *
;       Note:   Called at Ring 3 (Installable driver).                       *
;                                                                            *
;*****************************************************************************

        assume  cs:CodeGroup,ds:DataGroup,es:nothing

        Public  bid_init_3
bid_init_3:

        ;Save the DevHlp address

        mov     ax,word ptr es:[bx].INI_pDevHlp
        mov     word ptr DevHlpFunc,ax
        mov     ax,word ptr es:[bx].INI_pDevHlp+2
        mov     word ptr DevHlpFunc+2,ax

        call    DspBanner                       ;Display ASCII message

        ;We can't use address from ATTACH at ring 3 Init
        ;To circumvent this, we issue a specially defined IOCTL instead.

        ;Get handle for IOS$ character device

Open IOS:
```

```
        push    ds
        push    offset IOS_Name                 ;Push ptr to IOS name
        push    ds
        push    offset IOS_Handle               ;Push ptr to device handle
        push    ds
        push    offset Open_Action              ;Push ptr to action taken word
        push    0000H                           ;File size not applicable
        push    0000H                           ;and it's a DWORD
        push    0000H                           ;Normal file attributes
        push    0001H                           ;Open file if it exists
        push    0012H                           ;Open Mode
        push    0000H                           ;Reserved DWORD
        push    0000H
        call    DOSOPEN
        cmp     Open_Action,0001H               ;If IOS did not exist
        jne     IOS_Not_Found
        jmp     IOS_Found

        ;IOS not found

IOS_Not_Found:

        int     3

        jmp     BID_Abort

IOS_Found:

        ;Setup Driver Registration packet

        call    Setup_DRP

        ;Call registration function via IOCTL

        push    es                      ;Data Buffer pointer -> Init packet
        push    bx                      ;
        push    ds                      ;Parameter packet pointer ->
        push    offset Drv_Reg_Pkt      ;driver registration packet (DRP)
        push    IOS_Reg_Driver_Function ;IOCTL registration Function
        push    IOS_IOCTL_Category      ;Category
        push    IOS_Handle
        call    DOSDEVIOCTL

        ;Close IOS Character Device

        push    ax
        push    IOS_Handle
        call    DOSClose
        pop     ax

        ;Check registration status

Check_Reg_Status:

        jmp     Bid_Resident            ; <<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<< temp

        cmp     Drv_Reg_Pkt.DRP_reg_result,DRP_REMAIN_RESIDENT
        je      Bid_Resident
        cmp     Drv_Reg_Pkt.DRP_reg_result,DRP_MINIMIZE
        je      Bid_Minimize
        jmp     Bid_Abort

Bid_Resident:

        ;First copy of code. so leave driver resident
```

```
        mov     word ptr es:[bx].INI_pDevHlp,offset CodeInit
        mov     word ptr es:[bx].INI_pDevHlp+2,offset DataInit

        ;Display Installation successful message

        push    cs
        push    offset Succ_Message
        call    cs:[ILB_dprintf_rtn]
        add     sp,4

        jmp     BID_Init_Done

Bid_Minimize:

        ;Code already loaded, so minimize this copy

        mov     word ptr es:[bx].INI_pDevHlp,0
        mov     word ptr es:[bx].INI_pDevHlp+2,offset DataInit
        jmp     BID_Init_Done

Bid_Abort:

        ;Error occurred, so abort loading of BID

        mov     word ptr es:[bx].INI_pDevHlp,0
        mov     word ptr es:[bx].INI_pDevHlp+2,0
        mov     es:[bx].Pkt_fsStatus,BAD_DRV_REQ or STAT_ERROR or STAT_DONE ;
        jmp     BID_Init_Errored

BID_Init_Done:
        mov     es:[bx].Pkt_fsStatus,STAT_DONE

BID_Init_Errored:

        ret

bid_init        endp

;*****************************************************************************
;                                                                            *
; Setup_DRP - Setup Driver Registration Packet                               *
;                                                                            *
;       Set up fields in Driver registration packet which                    *
;       could not be assembled in.                                           *
;                                                                            *
;       Entry:                                                               *
;                                                                            *
;       Exit:                                                                *
;                                                                            *
;*****************************************************************************


        Public  Setup_DRP
Setup_DRP       proc    near

        ;Setup DRP pointer fields

        mov     Drv_Reg_Pkt.DRP_AER.segmt,cs
        mov     Drv_Reg_Pkt.DRP_AER.offst,offset Async_Request
        mov     Drv_Reg_Pkt.DRP_ILB.segmt,cs
        mov     Drv_Reg_Pkt.DRP_ILB.offst,offset Bid_ilb

        ;get physical address of ILB
```

```
        push    es                      ;save ptr to request pkt
        push    bx
        push    ds
        mov     si,offset Bid_ilb
        push    cs
        push    ds
        pop     es                      ;es is local data seg
        pop     ds                      ;ds:si-> ILB
        mov     dl,DevHlp_VirtToPhys
        call    dword ptr es:[DevHlpFunc]
        pop     ds                      ;restore ds to local data

        ;put physical address of ILB in DRP

        mov     Drv_Reg_Pkt.DRP_ILB_Phys.loword,bx
        mov     Drv_Reg_Pkt.DRP_ILB_Phys.hiword,ax
        pop     bx                      ;es:bx->request packet
        pop     es

        ;put DS in DRP so IOS can move into ILB CS

        mov     ax,ds
        mov     Drv_Reg_Pkt.DRP_DS,ax

        ret

Setup_DRP       endp

;******************************************************************************
;                                                                             *
; DspBanner - Display driver sign on banner                                   *
;                                                                             *
;       Entry:                                                                *
;                                                                             *
;       Exit:                                                                 *
;                                                                             *
;******************************************************************************

        Public  DspBanner
DspBanner       proc    near

        push    cs
        push    offset Sign_On_Message
        call    cs:[ILB_dprintf_rtn]
        add     sp,4
        ret

CodeInit segment

Sign_On_Message db CAR_RET,LINE_FEED
               db 'BIDSTART: OS/2 Sample SCSI BID for release 1.21'
               db CAR_RET,LINE_FEED,0

Succ_Message    db 'BIDSTART: Initialization successfully completed'
               db CAR_RET,LINE_FEED,LINE_FEED,0

CodeInit ends

DspBanner       endp

Code    ends

end
```

```
        page    ,132
        title   BIDSUBR - OS/2 1.21 sample LADDR compliant BID
        name    BIDSUBR
;
;*****************************************************************************
;                                                                             *
; Copyright (c) Microsoft Corporation 1990                                    *
; All Rights Reserved                                                         *
;                                                                             *
;*****************************************************************************
;
; this module contains assorted subroutines
;
;
.286

.xlist
        include bidsegs.inc     ;code and data segment definitions
        include defs.inc
.list

                assume  cs:CodeGroup,ds:nothing,es:DataGroup

Code    segment

;*****************************************************************************
;                                                                             *
; Enable_HaInt - Enable Host Adapter Interrupts                               *
;                                                                             *
;       Entry:  DS:SI -> Driver data block                                    *
;                                                                             *
;       Exit:   DS:SI -> Driver data block                                    *
;                                                                             *
;*****************************************************************************

        Public  Enable_HaInt
Enable_HaInt    proc    near

;
; add code to enable interrupts at the pic's (8259's)
;
        ret

Enable_HaInt    endp

;*****************************************************************************
;                                                                             *
; Disable_HaInt - Disable Host Adapter Interrupts                             *
;                                                                             *
;       Entry:  DS:SI -> Driver data block                                    *
;               Interrupts Disabled                                           *
;                                                                             *
;       Exit:   DS:SI -> Driver data block                                    *
;                                                                             *
;*****************************************************************************

Public  Disable_HaInt
Disable_HaInt   proc    near

;
; add code to disable interrupts at the pic's (8259's)
;
        ret

Disable_HaInt   endp
```

```
;*****************************************************************************
;                                                                             *
; Get_Options - Parse ASCII option string and setup HA paramters              *
;                                                                             *
;       Entry:  DS:SI -> Driver data block                                    *
;               ES:DI -> option string                                        *
;                                                                             *
;       Exit:                                                                 *
;                                                                             *
;*****************************************************************************

        Public  Get_Options
Get_Options     proc    near

;
; set our ddb up to reflect the default and optionally specified
; configuration parameters
;
        ret

Get_Options     endp

;*****************************************************************************
;                                                                             *
; Start_Scsi - Issue Start SCSI command to the host adapter                   *
;                                                                             *
;       Entry:  DS:SI -> driver data block                                    *
;                                                                             *
;       Exit:   DS:SI -> driver data block                                    *
;                                                                             *
;*****************************************************************************

        Public Start_Scsi
Start_Scsi proc  near

;
; add code to issue start command to adapter
;
        ret

Start_Scsi endp



;*****************************************************************************
;                                                                             *
; Wait_Idle - Wait for host adapter Idle                                      *
;                                                                             *
;       Entry:  DS:SI -> driver data block                                    *
;                                                                             *
;       Exit:   DS:SI -> driver data block                                    *
;               C set if failed                                               *
;                                                                             *
;*****************************************************************************

        Public  Wait_Idle
Wait_Idle       proc  near

;
; add code to wait - under timer protection - for adapter to become idle
;

        ret
```

```
Wait_Idle endp

Code    ends

end
```

```
;*****************************************************************************
;                                                                             *
; Copyright (c) Microsoft Corporation 1990                                    *
; All Rights Reserved                                                         *
;                                                                             *
;*****************************************************************************
;
; this module contains identicifation info for the sample bid
;
;

        BIDName         EQU     <'Sample SCSI BID '> ; must be 16 characters.
        BIDDate         EQU     <'04/17/90'>         ; must be 8 characters.
        BIDTime         EQU     <'12:00:00'>         ; must be 8 characters.
        BIDRev          EQU     <'0.00'>             ; must be 4 characters.

BID_Dev_Name    EQU     'SmplBid$'      ; name for device driver header.
```

BIDINFO.INC

```
;*****************************************************************************
;                                                                             *
; Copyright (c) Microsoft Corporation 1990                                    *
; All Rights Reserved                                                         *
;                                                                             *
;*****************************************************************************
;
; this module contains structures private to the sample bid
;
;

;
; ddb for sample bid
;

sample_bid_ddb  struc

                db      size DDB dup (?)        ; DDB header defined by ios.

bid_ddb_field_1 db      ?
bid_ddb_field_2 dw      ?
bid_ddb_field_3 db      ?
bid_ddb_field_4 dd      ?
interrupt_level db      ?                       ;Interrupt Channel of Adapter

sample_bid_ddb  ends

;
; srb for sample bid
;

sample_bid_srb  struc

                db      size SRB dup (?)        ; SRB header defined by ios.

bid_srb_field_1 db      ?
bid_srb_field_2 dw      ?
bid_srb_field_3 db      ?
bid_srb_field_4 dd      ?

sample_bid_srb  ends


; macro's which should be elsewhere <----------------------------------------

ENABLE  MACRO
        sti
ENDM

DISABLE MACRO
        cli
ENDM

;Flags for Device Status - should be in dcb.h <-------------------------------

ACTIVE          EQU     0001h

TIMEOUT         EQU     0FFh ; should be somewhere else <--------------------
```

Bidsampl.INC

```
;*****************************************************************************
;                                                                             *
; Copyright (c) Microsoft Corporation 1990                                    *
; All Rights Reserved                                                         *
;                                                                             *
;*****************************************************************************
;
; this module contains segment and group definitions for the sample bid
;
;

;Device header must be at offset 0 in Data segment

Header          segment word public 'DATA'      ;device header segment
Header          ends

DataInit        segment word public 'DATA'      ;throw away Init data
DataInit        ends

;IOS Linkage Block must be at offset 0 in Code segment

ILB_Seg         segment word public 'CODE'      ;ILB
ILB_Seg         ends

Code            segment word public 'CODE'      ;code segment
Code            ends

CodeInit        segment para public 'CODE'      ;throw away Init code
CodeInit        ends

DataGroup       group   Header,DataInit
CodeGroup       group   ILB_Seg,Code,CodeInit
```

BIDSEG5.INC

```
;*****************************************************************************
;       Copyright (c) Microsoft Corporation 1990
;       All rights reserved.
;
;*****************************************************************************
;*****************************************************************************
; AEP (Asynchronous Event Packet) Data Structure
;
;*****************************************************************************
AEP_PrivateLen  EQU     32        ;  Random Length Assignment
;  Definitions for AER Functions
AEP_INITIALIZE  EQU     0         ;  initialize driver/interface
AEP_RESERVED    EQU     1         ;  reserved
AEP_BOOT_COMPLETE       EQU     2         ;  booting done - switch to run time
AEP_CONFIG_PHYSICAL     EQU     3         ;  configure physical device
AEP_RCB_TIMEOUT EQU     4         ;  rcb timeout occured
AEP_CONFIG_LOGICAL      EQU     5         ;  configure logical device
AEP_DEVICE_INQUIRY      EQU     6         ;  get device indentification data
AEP_RESET_COUNTERS      EQU     7         ;  reset perfview counters
AEP_REGISTER_DONE       EQU     8         ;  registration processing complete
AEP_MAX_FUNC    EQU     8         ;  maximum legal function

AEP_header      STRUC
AEP_func        DW      ?               ;  Function Code
AEP_result      DW      ?             ;  result: zero = no error
AEP_ddb DD      ?               ;  driver data block Pointer
AEP_header      ENDS
;
;* define the aep for the initialize driver/interface event

AEP_bi_init     STRUC
AEP_filler_4    DW      ?           ;  "AEP_INITIALIZE"
AEP_filler_5    DW      ?           ;  result: zero = no error
AEP_filler_6    DD      ?           ;  driver data block pointer
AEP_bi_i_rp     DD      ?            ;  16:16 ptr to drivers init rp
AEP_bi_i_i_f    DW      ?           ;  id of initialized interface
                                        ;  isa: base address of controller
                                        ;  eisa/mca: slot number
AEP_bi_i_irq    DW      ?           ;  irq for bid to use
AEP_bi_i_ioa    DW      ?           ;  i/o address for bid to use
AEP_bi_i_length DB      ?         ;  option str length (excludes null)
AEP_bi_i_option DB      64 DUP (?)    ;  option string
AEP_bi_init     ENDS
;
;* define the aep for the boot complete IRQ event

AEP_boot_done   STRUC
AEP_filler_10   DW      ?          ;  "AEP_BOOT_COMPLETE"
AEP_filler_11   DW      ?          ;  result: zero = no error
AEP_filler_12   DD      ?          ;  driver data block pointer
AEP_boot_done   ENDS
;
;* define the aep for the configure physical device event

AEP_physical_config     STRUC
AEP_filler_7    DW      ?           ;  "AEP_CONFIG_PHYSICAL"
AEP_filler_8    DW      ?           ;  result: zero = no error
AEP_filler_9    DD      ?           ;  driver data block pointer
AEP_p_c_rp      DD      ?             ;  16:16 pointer to driver init rp
                                        ;  or zero.
AEP_p_c_dcb     DD      ?            ;  32-bit mem pool offset of DCB
AEP_p_c_length  DB      ?         ;  option str length (excludes null)
AEP_p_c_option  DB      64 DUP (?)     ;  option string
AEP_physical_config     ENDS
;
```

```
;* define the aep for the get device identification data event

AEP_inquiry_device      STRUC
AEP_filler_1    DW      ?           ;  "AEP_DEVICE_INQUIRY"
AEP_filler_2    DW      ?           ;  result: zero = no error
AEP_filler_3    DD      ?           ;  driver data block pointer
AEP_i_d_dcb     DD      ?            ;  dcb pointer
AEP_i_d_status  DB      ?          ;  ending status from device
AEP_inquiry_device      ENDS
;  AEP_i_d_status values
; #define DEVICE_NOT_PRESENT 0
; #define DEVICE_PRESENT           1    /* Device present and Inquiry data valid    */
AEP_NO_INQ_DATA EQU     0   ;  No inquiry data available
AEP_REAL_INQ_DATA       EQU     1   ;  Device present and Inquiry data valid
AEP_FAKE_INQ_DATA       EQU     2   ;  inquiry data fabricated by bid
AEP_NO_MORE_DEVICES     EQU     3   ;  no more devices of this type
;
;* define the aep for the reset counters event

AEP_counter_reset       STRUC
AEP_filler_13   DW      ?           ;  "AEP_RESET_COUNTERS"
AEP_filler_14   DW      ?           ;  result: zero = no error
AEP_filler_15   DD      ?           ;  driver data block pointer
AEP_c_r_dcb     DD      ?            ;  pointer to device control block
AEP_counter_reset       ENDS
;
;* define the aep for the rcb timeout event

AEP_rcb_timeout_occurred        STRUC
AEP_filler_16   DW      ?           ;  "AEP_RCB_TIMEOUT"
AEP_filler_17   DW      ?           ;  result: zero = no error
AEP_filler_18   DD      ?           ;  driver data block pointer
AEP_r_t_o_rcb   DD      ?           ;  pointer to offending rcb
AEP_rcb_timeout_occurred        ENDS
```

```
;******************************************************************************
;       Copyright (c) Microsoft Corporation 1990
;       All rights reserved.
;
;******************************************************************************
;******************************************************************************
; BPB (Bios Parameter Block) Data Structure
;
;******************************************************************************

BPB     STRUC
BPB_sector_size DW      ?        ;  Sector size
BPB_cluster_size        DB      ?       ;  sectors per allocation unit
BPB_reserved_sector     DW      ?    ;  DOS reserved sectors
BPB_number_fats DB      ?        ;  Number of FATS
BPB_directory_cnt       DW      ?       ;  Number of directories
BPB_sector_cnt  DW      ?         ;  Partition size in sectors
BPB_media_descriptor    DB      ?   ;  Media descriptor
BPB_fat_sectors DW      ?        ;  Fat sectors
BPB_spt DW      ?                ;  Sectors Per Track
BPB_head_cnt    DW      ?           ;  Number heads
BPB_hidden_sectors      DD      ?     ;  Hidden sectors
BPB_big_sector_cnt      DD      ?     ;  DWORD number sectors
BPB     ENDS
```

```
;*****************************************************************************
;       Copyright (c) Microsoft Corporation 1990
;       All rights reserved.
;
;*****************************************************************************
;*****************************************************************************
; DCB (Device Control Block) Data Structure
;
;*****************************************************************************
;
;* define the call down table
DCB_cd_table_depth      EQU     10      ;  allow a layering of ten deep

DCB_cd_entry    STRUC
DCB_cd_io_address       DD      ?       ;  16:16 address of request routine.
DCB_cd_aer      DD      ?               ;  16:16 address of async event rtn.
DCB_cd_ddb      DD      ?               ;  ios:32 pointer to ddb.
DCB_cd_dvt      DD      ?               ;  ios:32 pointer to dvt.
DCB_cd_flags    DW      ?             ;  demand bits - as defined below.
DCB_cd_filler   DW      ?             ;  reserved - must be zero.
DCB_cd_pv_len   DW      ?             ;  length of perfview data.
DCB_cd_pv_sel   DW      ?             ;  16-bit selector of perfview data
DCB_cd_pv_str   DD      ?             ;  32-bit offset of perfview strings
DCB_cd_pv_ptr   DD      ?             ;  32-bit offset of perfview header
                                        ;  or zero if one does not exist.
DCB_cd_entry    ENDS
;
;* define the monitor's structure

DCB_cpf_mon_req STRUC
DCB_cpfcommnd   DB      ?             ;  CODE PAGE/FONT COMMAND
DCB_cpfrsvd1    DB      ?             ;  RESERVED AREA
DCB_cpfrsvd2    DB      ?             ;  RESERVED AREA
DCB_cpfrsvd3    DB      ?             ;  RESERVED AREA
DCB_cpfreturn   DB      ?             ;  SWITCHER RETURN CODE
DCB_cpfcodepg   DB      ?             ;  CODE PAGE WORD
DCB_cpffont     DB      ?             ;  FONT WORD
DCB_cpf_mon_req ENDS
DCB_CPF_MON_REQ_LEN     EQU     SIZE    DCB_cpf_mon_req/2
DCB_MONITOR_SIZE        EQU     128
DCB_BUFFER_SIZE EQU     128
;
;* define the demand bits for use with DCB_cd_flags
DCB_dmd_srb_cdb EQU     0001H   ;  there must be an srb and cdb for
                                        ;  each rh.
DCB_dmd_rsrv_1  EQU     0002H   ;  reserved - must be zero.
DCB_dmd_logical EQU     0004H   ;  media address must be logical and
                                        ;  dcb must be for a logical device.
DCB_dmd_physical        EQU     0008H   ;  media adresses must be physical
                                        ;  and dcb must be for a physical
                                        ;  device.
DCB_dmd_small_memory    EQU     0010H   ;  data buffers must reside in
                                        ;  in low 16M
DCB_dmd_single_rh       EQU     0020H   ;  only one rh is permitted per rlh
DCB_dmd_single_sgd      EQU     0040H   ;  only one sgd is permitted per rh
DCB_dmd_word_align      EQU     0080H   ;  data buffers must be word aligned
DCB_dmd_dword_align     EQU     0080H   ;  data buffers must be double word
                                        ;  aligned
DCB_dmd_not_512 EQU     0100H   ;  secor size on the media is not
                                        ;  512 bytes
DCB_dmd_not_286 EQU     0200H   ;  code will not execute properly
                                        ;  a 286 cpu
DCB_dmd_ios_16  EQU     0400H   ;  ios memory pool pointers must be
                                        ;  16 bits or less in size
DCB_dmd_rsrv_2  EQU     7800H   ;  reserved - must be zero.
```

```
DCB_dmd_contig_sns      EQU     8000H   ;  sense data area must follow the
                                        ;  bid private area of the srb.
;
;* define the device control block (dcb)
;
;   NOTE: Any changes in this structure must also be reflected in cfr.h
;

DCB     STRUC
;  offset 0
DCB_phys_addr   DD      ?           ;  Physical Pointer to DCB
DCB_physical_dcb        DD      ?       ;  DCB for physical device
DCB_device_type DB      ?         ;  Device Type
DCB_bus_type    DB      ?           ;  Type of BUS, see below
DCB_device_name DB      6 DUP (?)     ;  ascii name of device
;  offset 10h
DCB_drive_lttr_equiv    DB      ?   ;  numeric equivalent of drive
                                        ;  letter where c = 2. set up by
                                        ;  iosserv during logical device
                                        ;  associate processing.
DCB_unit_number DB      ?         ;  either physical drive number
                                        ;  (sequential drive number or'd
                                        ;  with 80h) or unit number within
                                        ;  tsd. set up by iosbid for disk
                                        ;  physical dcb's. set up by tsdpart
                                        ;  for disk logical dcb's. set up by
                                        ;  tsdaer for cdrom physical dcb's.
                                    ;  >>>>>>> previously DCB_physical_device
DCB_controller  DB      ?           ;  controller number
                                        ;  the same as spindle number which
                                        ;  fdisk calls physical drive.
                                        ;  used by esdi bid to select drive.
                                        ;  set up by iosbid for esdi only.
DCB_abios_dev_type      DB      ?     ;  device type in abios form
;  offset 14h
DCB_abios_lid   DW      ?           ;  logical unit id for abios
DCB_scsi_target DB      ?         ;  scsi target id
                                    ;  >>>>>>> previously DCB_device_id
DCB_scsi_lun    DB      ?           ;  SCSI logical unit number
;  offset 18h
;       BYTE    DCB_scsi_channel;       /* channel (cable) within adaptor   */
DCB_reserved_00 DB      ?         ;  reserved - must be zero
DCB_esdi_unit_on_ctl    DB      ?   ;  unit within controller which is
DCB_max_sense_data_len  DB      ? ;  Maximum sense Length
DCB_SG_Max_count        DB      ?       ;  Max S/G desc count supported
;  offset 1ch
DCB_bid_spec_area_len   DW      ?  ;  Bid Specific Area Length
DCB_device_flags        DW      ?       ;  device characteristics flags
;  offset 20h
DCB_TSD_Flags   DW      ?           ;  Flags for TSD
DCB_SCSI_VSD_FLAGS      DW      ?     ;  Flags for SRB builder
DCB_BID_Specific        DB      4 DUP (?)   ;  4 more bytes for BID use
DCB_Device_Status       DW      ?       ;  Device Status for BID use
DCB_q_algo      DW      ?             ;  queuing algorithm index - see
                                        ;  values below.
DCB_q_work      DD      ?             ;  work space for queuing code
;  offset 30h
DCB_work_queue  DD      ?         ;  srb queue anchor for BID use
DCB_next_dcb_logical    DD      ?   ;  link to next DCB
DCB_dcb_size    DW      ?           ;  size of this dcb
DCB_size_cd     DW      ?           ;  size of calldown entry
DCB_ptr_cd      DD      ?           ;  ptr to start of calldown table
;  offset 40h
DCB_pv_tot_len  DD      ?         ;  size of pvw data in cd stack
DCB based start DD      ?         :  these two fields are meaningful
```

```
DCB_based_len   DD      ?           ;  only when DCB_DEV_BASED is on.
                                    ;  they are provide to help drivers
                                    ;  map multiple physical dcb's to a
                                    ;  single real drive/drive array.
                                    ;  any driver using these fields
                                    ;  MUST set DCB_DEV_BASED and may
                                    ;  not use them if that bit is
                                    ;  already set.
DCB_reserved_1  DW      ?           ;  reserved - must be zero
DCB_reserved_2  DW      ?           ;  reserved - must be zero
;  offset 50h
DCB_inquiry_flags       DB      8 DUP (?)   ;  Device Inquiry Flags
DCB_vendor_id   DB      8 DUP (?)       ;  Vendor ID string
;  offset 60h
DCB_product_id  DB      16 DUP (?)      ;  Product ID string
;  offset 70h
DCB_rev_level   DB      4 DUP (?)       ;  Product revision level
DCB_reserved_3  DD      ?           ;  reserved - must be zero
DCB_reserved_4  DD      ?           ;  reserved - must be zero
DCB_reserved_5  DD      ?           ;  reserved - must be zero
;  offset 80h
DCB_reserved_6  DB      ?           ;  reserved - must be zero
DCB_reserved_7  DB      ?           ;  reserved - must be zero
DCB_reserved_8  DB      ?           ;  reserved - must be zero
DCB_reserved_9  DB      ?           ;
                                ;  >>>>>>> previously DCB_device_unit
DCB_reserved_10 DD      ?           ;  reserved - must be zero
DCB_reserved_11 DD      ?           ;  reserved - must be zero
DCB_reserved_12 DD      ?           ;  reserved - must be zero
;  offset 90h
DCB_reserved_13 DW      ?           ;  reserved - must be zero
DCB_apparent_blk_size   DW      ?   ;  block size of device as seen by
                                    ;  tsd an above.
                                ;  >>>>>>> previously DCB_block_size
DCB_apparent_sector_cnt DD      ?   ;  number of sectors as seen by tsd
                                    ;  and above.
DCB_apparent_cyl_cnt    DD      ?   ;  number of cylinders as seen by
                                    ;  the tsd and above.
                                ;  >>>>>>> previously DCB_number_cylinders
DCB_apparent_head_cnt   DW      ?   ;  number of heads as seen by tsd
                                    ;  and above.
                                ;  >>>>>>> previously DCB_physical_nheads
DCB_apparent_spt        DW      ?       ;  number of sectors per track as
                                        ;  seen by tsd and above.
                                ;  >>>>>>> previously DCB_physical_spt
;  offset a0h
DCB_reserved_14 DW      ?           ;  reserved - must be zero
DCB_actual_blk_size     DW      ?   ;  actual block size of the device
                                    ;  as seen below the tsd.
DCB_actual_sector_cnt   DD      ?   ;  number of sectors as seen below
                                    ;  the tsd.
DCB_actual_cyl_cnt      DD      ?   ;  number of cylinders as seen
                                    ;  below the tsd.
DCB_actual_head_cnt     DW      ?   ;  number of heads as seen below
                                    ;  the tsd.
DCB_actual_spt  DW      ?           ;  number of sectors per track as
                                    ;  seen below the tsd.
;  offset b0h
DCB_cd_table    DB      SIZE DCB_cd_entry * DCB_cd_table_depth DUP (?)
DCB     ENDS
;
;* define the device control block (dcb) for disk and cd-rom device types
;
;   NOTE: Any changes in this structure must also be reflected in cfr.h
;
```

```
DCB_disk        STRUC
DCB_filler_1    DB      SIZE DCB DUP (?)          ;  standard dcb header
DCB_Write_Precomp       DW      ?       ;  Write Precompensation
DCB_partition_type      DB      ?       ;  partition type
DCB_ft_part_num DB      ?       ;  partition number as used by ft
DCB_Partition_Start     DD      ?       ;  partition start sector
DCB_Partition_End       DD      ?       ;  partition end sector
DCB_bpb DB      SIZE BPB DUP (?)                ;  bpb from the media
DCB_rbpb        DB      SIZE BPB DUP (?)              ;  recomended bpb for the partition
DCB_audio_flags DD      ?       ;  device audio sub-system state
DCB_disk        ENDS
;
;* define the device control block (dcb) for printer device types
;
;   NOTE: Any changes in this structure must also be reflected in cfr.h
;

DCB_printer     STRUC
DCB_filler_2    DB      SIZE DCB DUP (?)          ;  standard dcb header
DCB_Prt_cpqueue DD      ?       ;  address of printer queue
DCB_Prt_dosoffset       DW      ?       ;  offset of DOS request block
DCB_Prt_dosrqseg        DW      ?       ;  segment of DOS request block
DCB_Prt_monoffset       DW      ?       ;  offset of monitor thread request
DCB_Prt_monseg  DW      ?       ;  segment of monitor thread request
DCB_Prt_prtfrmctrl      DW      ?       ;  current frame control status
DCB_Prt_prtcount        DW      ?       ;  count of chars left to print
DCB_Prt_count1k DB      ?       ;  count of 128 byte blocks
DCB_Prt_accessctr       DB      ?       ;  count of processes accessing drvr
DCB_Prt_retryctr        DB      ?       ;  max number of retries allowed
DCB_Prt_cancelflags     DB      ?       ;  cancel procedure flags
DCB_Prt_cancelstatus    DB      ?       ;  printer status port data
DCB_Prt_share_interrupt DB      ? ;  printer supports interrupt share
DCB_Prt_splpid  DW      ?       ;  active spooler's process id
DCB_Prt_DosVar  DD      ?       ;  pointer to dos variables
DCB_Prt_Semiphore       DD      ?       ;  used to coordinate
DCB_Prt_BlockedCounter  DB      ? ;  number of blocked packets
DCB_Prt_erroraction     DB      ?       ;  monitor action code
DCB_Prt_monflags        DW      ?       ;  monitor flags
DCB_Prt_stratsfn        DW      ?       ;  user's system file number
DCB_Prt_reserved_0      DW      ?       ;  reserved for future use
DCB_Prt_monbuffer       DB      DCB_MONITOR_SIZE DUP (?) ;  data sent to monitor
        ;  Order of next four entries must not be changed
DCB_Prt_moncpbufsize    DW      ?       ;  size of code page buffer
DCB_Prt_moncpinflgs     DW      ?       ;  monitor code page flags
DCB_Prt_moncpinsfn      DW      ?       ;  user's code page system
DCB_Prt_moncpinbuf      DB      SIZE DCB_cpf_mon_req DUP (?) ;  10 byte buffer
        ;  Order of next five entries must not be changed
DCB_prt_delta_dcb       DW      ?       ;  offset from begining of dcb
DCB_Prt_monfinalbufsize DW      ? ;  size of final buffer
DCB_Prt_monflags0       DW      ?       ;  monitor flags
DCB_Prt_mondatinsfn     DW      ?       ;  user's data system file number
DCB_Prt_buffer  DB      DCB_BUFFER_SIZE DUP (?) ;  128 byte device buffer
DCB_Prt_monhandle       DW      ?       ;  data monitor handle
DCB_Prt_cpfhandle       DW      ?       ;  code page monitor handle
DCB_Prt_commonflags     DW      ?       ;  device driver flags
DCB_Prt_commonflags1    DW      ?       ;  device driver flags
DCB_Prt_reserved_1      DD      ?       ;  reserved for future use
DCB_Prt_reserved_2      DD      ?       ;  reserved for future use
DCB_Prt_reserved_3      DD      ?       ;  reserved for future use
DCB_Prt_reserved_4      DD      ?       ;  reserved for future use
DCB_Prt_reserved_5      DD      ?       ;  reserved for future use
;       USHORT  DCB_Prt_errorflags;     /* monitor error flags          */
;       USHORT  DCB_Prt_errorsfn;       /* monitor error function       */
;       BYTE    DCB_Prt_errorstatus;    /* monitor return code          */
```

```
DCB_printer     ENDS
;
;* define the device control block (dcb) for tape device types
;
;   NOTE: Any changes in this structure must also be reflected in cfr.h
;

DCB_tape        STRUC
DCB_filler_3    DB      SIZE DCB DUP (?)          ;  standard dcb header
DCB_Tape_Info   DD      ?
DCB_Tape_Primary_BlkLen DD      ?
DCB_Tape_Sys_File_Num   DW      ?
DCB_Tape_Frame_BlkCount DW      ?
DCB_Tape_Unit_Number    DW      ?
DCB_Tape_Format_Type    DW      ?
DCB_Tape_ECC_Type       DW      ?
DCB_Tape_Data_Blks_Cnt  DW      ?
DCB_Tape_ID     DB      15 DUP (?)
DCB_Tape_Reserve_Byte   DB      ?
DCB_tape        ENDS
;
;* define the algorithm indecees for DCB_q_algo
DCB_q_li_fo     EQU     00H             ;  last in, first out
DCB_q_fi_fo     EQU     02H             ;  first in, first out
DCB_q_priority  EQU     04H             ;  priority queuing
DCB_q_c_scan    EQU     06H             ;  special algorithm for disk
;
;* define the device type codes for use with DCB_device_type
DCB_type_disk   EQU     00H     ;  All Direct Access Devices
DCB_type_tape   EQU     01H     ;  Sequencial Access Devices
DCB_type_printer        EQU     02H     ;  Printer Device
DCB_type_processor      EQU     03H     ;  Processor type device
DCB_type_worm   EQU     04H     ;  Write Once Read Many Device
DCB_type_cdrom  EQU     05H     ;  CD ROM Device
DCB_type_scanner        EQU     06H     ;  Scanner Device
DCB_type_optical_memory EQU     07H     ;  some Optical disk
DCB_type_changer        EQU     08H     ;  Changer device e.g. juke box
DCB_type_comm   EQU     09H     ;  Communication devices
;
;* define Volume characteristics
DCB_INQ_DATA_LENGTH     EQU     size DCB_inquiry_flags + size DCB_vendor_id + size DCB_product_id + size DCB_rev_
;
;* define the bus interface type codes for use with DCB_bus_type
DCB_BUS_ESDI    EQU     00H     ;  ESDI BUS
DCB_BUS_SCSI    EQU     01H     ;  SCSI BUS
DCB_BUS_IPI     EQU     02H     ;  IPI  BUS
;
;* define the flags for use with DCB_tsd_flags
DCB_TSD_INVALID_PARTITION       EQU     1     ;  Don't trust the BPB
DCB_TSD_BAD_MBR EQU     1     ;  Bad MBR - physical device only
DCB_TSD_DISK_XLAT       EQU     2     ;  Large drive translation
DCB_TSD_MEDIA_CHANGED   EQU     4     ;  Media in drive changed
DCB_TSD_PHYS_GEOM_SET   EQU     8     ;  geometry data set in physical dcb
;
;* define the device characteristics flags for use with DCB_device_flags
DCB_DEV_RO      EQU     8000H   ;  drive is in read only mode
DCB_DEV_NFG     EQU     4000H   ;  drive is defective
DCB_LAST_ACCESS EQU     2000H   ;  used by FT to indicate this
                                ;  partition was accessed last
DCB_UNCERTAIN_MEDIA     EQU     1000H   ;  media may have been changed
DCB_DEV_FLAG_REMOV      EQU     0080H   ;  media can be removed from device
DCB_DEV_VAR_BLKSIZE     EQU     0040H   ;  device has variable block size
DCB_DEV_READ_ONLY       EQU     0020H   ;  device is read only
DCB_DEV_HWCACHE EQU     0010H   ;  device has hw cache implented
DCB_DEV_RAM_DISK        EQU     0008H   ;  seek time independent of position
```

```
DCB_DEV_LOGICAL EQU     0004H   ;  on = logical; off = physical dcb
DCB_DEV_BASED   EQU     0002H   ;  based addressing in use - see
                                ;  DCB_based_start and DCB_based_len
DCB_STATUS_UNKNOWN      EQU     0001H   ;  device status is unknown
```

```
;******************************************************************************
;       Copyright (c) Microsoft Corporation 1990
;       All rights reserved.
;
;******************************************************************************
;******************************************************************************
; DDB (Driver Data Block) Data Structure
;
;******************************************************************************

DDB     STRUC
DDB_phys_addr   DD      ?           ;  Physical address of ddb
DDB_link        DW      ?               ;  pointer to next ddb in chain
DDB_Next_DDB    DD      ?            ;  next ddb on dvt_ddb chain
DDB_Next_DDB_init       DD      ?       ;  next ddb on dvt_ddb_init chain
DDB_pv_basic    DB      8 DUP (?)         ;  perfview basic string
DDB_pv_extra    DB      8 DUP (?)         ;  perfview extra string
DDB_pv_data     DD      ?             ;  pointer to perfview data
DDB_pv_len      DW      ?              ;  length of perfview data
DDB     ENDS
```

```
;*****************************************************************************
;       Copyright (c) Microsoft Corporation 1990
;       All rights reserved.
;
;*****************************************************************************
;*****************************************************************************
; Module Name: DEFS.H - Common Definitions file
;
;*****************************************************************************
;  ascii terminal (ie debug terminal) control characters
BELL    EQU     07H             ;  bell character.
LINE_FEED       EQU     0aH             ;  line feed character.
CAR_RET EQU     0dH             ;  carriage return character.
;  debug terminal comm port addresses
COM1_PORT       EQU     03f8H           ;  com1's i/o address
COM2_PORT       EQU     02f8H           ;  com2's i/o address
;  various constants for use with 8250 type uart's
COM_DAT EQU     00H             ;
COM_IEN EQU     01H             ;  interrupt enable
COM_IER EQU     02H             ;  interrupt ID
COM_LCR EQU     03H             ;  line control registers
COM_MCR EQU     04H             ;  modem control register
COM_LSR EQU     05H             ;  line status register
COM_MSR EQU     06H             ;  modem status regiser
COM_DLL EQU     00H             ;  divisor latch least sig
COM_DLM EQU     01H             ;  divisor latch most sig
;  structure to define hibyte and lobyte within a word

w_s     STRUC
lobyte  DB      ?
hibyte  DB      ?
w_s     ENDS
;  structure to define hiword and loword within a double word

dw_s    STRUC
loword  DW      ?
hiword  DW      ?
dw_s    ENDS

dw_ss   STRUC
lo      DW      ?
hi      DW      ?
dw_ss   ENDS
;  structure to define offst and segmt within a 16:16 far pointer

FarPtr  STRUC
offst   DW      ?
segmt   DW      ?
FarPtr  ENDS
;  structure to define offst32 and segmt32 within a 16:32 far pointer

FarPtr32        STRUC
offst32 DD      ?
segmt32 DW      ?
pad32   DW      ?
FarPtr32        ENDS
```

```
;*****************************************************************************
;       Copyright (c) Microsoft Corporation 1990
;       All rights reserved.
;
;*****************************************************************************
;*****************************************************************************
; Module Name: DEVDRV.H - Standard Device Driver Definitions file
;
;*****************************************************************************
;
;***     Device Table Record
;*
;*      Devices are described by a chain of these records
;

DevHeader       STRUC
tag_Next        DD      ?           ; Pointer to next device header
tag_Att DW      ?                       ;  Attributes of the device
tag_Strat       DW      ?                   ;  Strategy entry point
tag_IDCEP       DW      ?                   ;  IDC entry point
tag_Name        DB      8 DUP (?)               ;  Name of device (only first byte used for block)
DevHeader       ENDS
;*****************************************************************************
;*
;* Packet Header structure
;*
;* All packets contain this structure at their start
;*
;*****************************************************************************

PktHdr  STRUC
PKT_cbLen       DB      ?
PKT_bUnit       DB      ?
PKT_bCmd        DB      ?
PKT_fsStatus    DW      ?
PKT_ulReserved  DD      ?
PKT_ppktQLink   DD      ?
PktHdr  ENDS
;
;* Packet structure for init command
;

InitPkt STRUC
INI_hdr         DB      SIZE PKTHDR DUP (?)  ;  packet header
INI_cUnit       DB      ?   ;  number of units returned
INI_Code                DW      ?   ;  (exit)  size of code segment
INI_Data                DW      ?   ;  (exit)  size of data segment
INI_fpBPB       DD ? ; BPB Array for logical devices
InitPkt ENDS

InitPkt2 STRUC
                                DB SIZE PKTHDR DUP (?)
                        DB ?
INI_pDevHlp     DD ?     ; Device Helper Address
INI_pchParms DD ?  ; Points to the INIT arguments
INI_bDrv                DB ? ; Drive number for the first block device unit
InitPkt2 ENDS



;
;* Packet structure for IOCTl command
;

IOCTlPkt        STRUC
IOC hdr DB      SIZE PKTHDR DUP (?)                                     ;  packet header
```

```
IOC_GIOCategory DB      ?               ;  Category Code
IOC_GIOFunction DB      ?               ;  Function code
IOC_GIOParaPack DD      ?               ;  pointer to parameter packet
IOC_GIODataPack DD      ?               ;  pointer to data packet
IOC_GIOSFN      DW      ?                       ;  (used by Spooler?)
IOC_GIOParaLen  DW      ?                ;  length of parameter packet
IOC_GIODataLen  DW      ?                ;  length of data packet
IOCTlPkt        ENDS
;
;* Packet structure for Strat1 I/O commands
;

ReqPkt  STRUC
REQ_hdr DB      SIZE PKTHDR DUP (?)                             ;  packet header
REQ_IOMedia     DB      ?                               ;  Media Descriptor Byte
REQ_IOpData     DD      ?                   ;  Data Buffer Address
REQ_IOCount     DW      ?                           ;  Block Count
REQ_IOStart     DD      ?                   ;  IO Start Block
ReqPkt  ENDS
;*****************************************************************************
;*
;* Register Packet structure
;*
;* Used by CallMasm() to pass register values to/from C code
;*
;*****************************************************************************

_REGS struc
   REG_AX                       DW                      ?
   REG_BX                       DW                      ?
   REG_CX                       DW                      ?
   REG_DX                       DW                      ?
   REG_ES                       DW                      ?
   REG_DS                       DW                      ?
_REGS ENDS


;  far ptr to Global Info Segment
;  typedef struct InfoSegGDT far *pSIS
;
;***     Device Driver Type defininitions
;*
;
DEV_CIN EQU     0001H   ;    0  Device is console in
DEV_COUT        EQU     0002H   ;    1  Device is console out
DEV_NULL        EQU     0004H   ;    2  Device is the Null device
DEV_CLOCK       EQU     0008H   ;    3  Device is the clock device
DEV_FCNLEV      EQU     0380H   ;  9-7  Device function level
DEV_30  EQU     0800H   ;   11  Accepts Open/Close/Removable Media
DEV_SHARE       EQU     1000H   ;   12  Device wants FS sharing checking
DEV_NON_IBM     EQU     2000H   ;   13  Device is a non IBM device.
DEV_IDC EQU     4000H   ;   14  Device accepts IDC request
DEV_CHAR_DEV    EQU     8000H   ;   15  Device is a character device
SizeOfMemoryBlock       EQU     8000H    ;  Size of Memory Block
;
;*      Level definitions for devices
;
DEVLEV_0        EQU     0000H  ;  DOS 3.0 and before (NEEDS TO BE FIXED)
DEVLEV_1        EQU     0080H  ;  OS/2 v1.0
DEVLEV_2        EQU     0100H  ;  OS/2 v1.2 (new gen ioctl iface)
;
;*      Return Code values
;
STAT_ERROR      EQU     8000H
STAT DRV ERR    EOU     4000H
```

```
STAT_BUSY       EQU     0200H
STAT_DONE       EQU     0100H
STAT_ERR_FIELD  EQU     00ffH
DEV_IN_USE      EQU     0014H
INV_PARM        EQU     0013H
NOMONSUPPORT    EQU     0012H
IO_CALL_INT     EQU     0011H
UNC_MEDIA       EQU     0010H
CHANGE_DISK     EQU     000dH
GEN_FAILURE     EQU     000cH
READ_FAULT      EQU     000bH
WRITE_FAULT     EQU     000aH
SEC_NOT_FOUND   EQU     0008H
UNKNOWN_MEDIA   EQU     0007H
SEEK_ERROR      EQU     0006H
BAD_DRV_REQ     EQU     0005H
CRC_ERROR       EQU     0004H
UNKNOWN_CMD     EQU     0003H
DEV_NOT_RDY     EQU     0002H
UNKNOWN_UNIT    EQU     0001H
WRITE_PROTECT   EQU     0000H
;
;*  device commands
;
CMDInit EQU     0       ;  INIT command
CMDMedChk       EQU     1       ;  Media Check
CMDBldBPB       EQU     2       ;  build BPB
CMDIOCTLR       EQU     3       ;  reserved for 3.x compatability
CMDINPUT        EQU     4       ;  read data from device
CMDNDR  EQU     5       ;  non-destructive read
CMDInputS       EQU     6       ;  input status
CMDInputF       EQU     7       ;  input flush
CMDOUTPUT       EQU     8       ;  write data to device
CMDOUTPUTV      EQU     9       ;  write data and verify
CMDOutputS      EQU     10      ;  output status
CMDOutputF      EQU     11      ;  output flush
CMDIOCTLW       EQU     12      ;  reserved for 3.x compatability
CMDOpen EQU     13      ;  device open
CMDClose        EQU     14      ;  device close
CMDRemMed       EQU     15      ;  is media removable
CMDGenIOCTL     EQU     16      ;  Generic IOCTL
CMDResMed       EQU     17      ;  reset media uncertain
CMDGetLogMap    EQU     18
CMDSetLogMap    EQU     19
CMDDeInstall    EQU     20      ;  De-Install driver
CMDPartfixeddisks       EQU     22      ;  Partitionable Fixed Disks
CMDGetfd_logunitsmap    EQU     23      ;  Get Fixed Disk/Logical Unit Map
CMDInputBypass  EQU     24      ;  cache bypass read data
CMDOutputBypass EQU     25      ;  cache bypass write data
CMDOutputBypassV        EQU     26      ;  cache bypass write data and verify
CMDInitBase     EQU     27      ;  INIT command for base DDs
CMDShutdown     EQU     28
CMDGetDevSupport        EQU     29      ;  query for extended capability
CMDAddOnPrep    EQU     97      ;  Prepare for add on
CMDStar EQU     98      ;  start console output
CMDStop EQU     99      ;  stop console output
```

```
;*****************************************************************************
;       Copyright (c) Microsoft Corporation 1990
;       All rights reserved.
;
;*****************************************************************************
;*****************************************************************************
; DEVHLP.H - defines the devhlp names and numbers
;
;*****************************************************************************
DevHlp_SchedClock       EQU     0       ;   0   Called each timer tick
DevHlp_DevDone  EQU     1       ;   1   Device I/O complete
DevHlp_Yield    EQU     2       ;   2   yield CPU if resched set
DevHlp_TCYield  EQU     3       ;   3   yield to time critical  task
DevHlp_ProcBlock        EQU     4       ;   4   Block on event
DevHlp_ProcRun  EQU     5       ;   5   Unblock process
DevHlp_SemRequest       EQU     6       ;   6   claim a semaphore
DevHlp_SemClear EQU     7       ;   7   release a semaphore
DevHlp_SemHandle        EQU     8       ;   8   obtain a semaphore handle
DevHlp_PushRequest      EQU     9       ;   9   Push the request
DevHlp_PullRequest      EQU     10      ;   A   Pull next request from Q
DevHlp_PullParticular   EQU     11      ;   B   Pull a specific request
DevHlp_SortRequest      EQU     12      ;   C   Push request in sorted order
DevHlp_AllocReqPacket   EQU     13      ;   D   allocate request packet
DevHlp_FreeReqPacket    EQU     14      ;   E   free request packet
DevHlp_QueueInit        EQU     15      ;   F   Init/Clear char queue
DevHlp_QueueFlush       EQU     16      ;  10   flush queue
DevHlp_QueueWrite       EQU     17      ;  11   Put a char in the queue
DevHlp_QueueRead        EQU     18      ;  12   Get a char from the queue
DevHlp_Lock     EQU     19      ;  13   Lock segment
DevHlp_Unlock   EQU     20      ;  14   Unlock segment
DevHlp_PhysToVirt       EQU     21      ;  15   convert physical address to virtual
DevHlp_VirtToPhys       EQU     22      ;  16   convert virtual address to physical
DevHlp_PhysToUVirt      EQU     23      ;  17   convert physical to LDT
DevHlp_AllocPhys        EQU     24      ;  18   allocate physical memory
DevHlp_FreePhys EQU     25      ;  19   free physical memory
DevHlp_SetROMVector     EQU     26      ;  1A   set a ROM service routine vector
DevHlp_SetIRQ   EQU     27      ;  1B   set an IRQ interrupt
DevHlp_UnSetIRQ EQU     28      ;  1C   unset an IRQ interrupt
DevHlp_SetTimer EQU     29      ;  1D   set timer request handler
DevHlp_ResetTimer       EQU     30      ;  1E   unset timer request handler
DevHlp_MonitorCreate    EQU     31      ;  1F   create a monitor
DevHlp_Register EQU     32      ;  20   install a monitor
DevHlp_DeRegister       EQU     33      ;  21   remove a monitor
DevHlp_MonWrite EQU     34      ;  22   pass data records to monitor
DevHlp_MonFlush EQU     35      ;  23   remove all data from stream
DevHlp_GetDOSVar        EQU     36      ;  24   Return pointer to DOS variable
DevHlp_SendEvent        EQU     37      ;  25   an event occurred
DevHlp_ROMCritSection   EQU     38      ;  26   ROM Critical Section
DevHlp_VerifyAccess     EQU     39      ;  27   Verify access to memory
DevHlp_RAS      EQU     40      ;  28   Put info in RAS trace buffer
DevHlp_ABIOSGetParms    EQU     41      ;  29   Get ABIOS Calling Parms
DevHlp_AttachDD EQU     42      ;  2A   Attach to a device driver
DevHlp_InternalError    EQU     43      ;  2B   Signal an internal error
DevHlp_ModifyPriority   EQU     44      ;  2C   Undocumented (used by PM)
DevHlp_AllocGDTSelector EQU     45      ;  2D   Allocate GDT Selectors
DevHlp_PhysToGDTSelector        EQU     46      ;  2E   Convert phys addr to GDT sel
DevHlp_RealToProt       EQU     47      ;  2F   Change from real to protected mode
DevHlp_ProtToReal       EQU     48      ;  30   Change from protected to real mode
DevHlp_EOI      EQU     49      ;  31   Send EOI to PIC
DevHlp_UnPhysToVirt     EQU     50      ;  32   mark completion of PhysToVirt
DevHlp_TickCount        EQU     51      ;  33   modify timer
DevHlp_GetLIDEntry      EQU     52      ;  34   Obtain Logical ID
DevHlp_FreeLIDEntry     EQU     53      ;  35   Release Logical ID
DevHlp_ABIOSCall        EQU     54      ;  36   Call ABIOS
DevHlp_ABIOSCommonEntry EQU     55      ;  37   Invoke Common Entry Point
```

```
DevHlp_GetDeviceBlock   EQU     56      ;  38   Get ABIOS Device Block
                                        ;  39   Reserved for Profiling Kernel
DevHlp_RegisterStackUsage       EQU     58      ;  3A   Register for stack usage
DevHlp_LogEntry EQU     59      ;  3B   Place data in log buffer
DevHlp_VideoPause       EQU     60      ;  3C Video pause on/off                - D607
DevHlp_Save_Message     EQU     61      ;  3D   Save msg in SysInit Message Table
DevHlp_RegisterPDD      EQU     62      ;  3E   Register PDD entry point with
                                        ;       VDM manager for later PDD-VDD
                                        ;       communication
DevHlp_RegisterBeep     EQU     63      ;  3F   register PTD beep service
                                        ;       entry point with kernel
DevHlp_Beep     EQU     64      ;  40   preempt beep service via PTD
DevHlp_FreeGDTSelector  EQU     65      ;  41   Free allocated GDT selector
DevHlp_PhysToGDTSel     EQU     66      ;  42   Convert Phys Addr to GDT sel
                                        ;       with given access
                                        ;       BUGBUG: TEMPORARY!!!
DevHlp_VMLock   EQU     67      ;  43   Lock linear address range
DevHlp_VMUnlock EQU     68      ;  44   Unlock address range
DevHlp_VMAlloc  EQU     69      ;  45   Allocate memory
DevHlp_VMFree   EQU     70      ;  46   Free memory or mapping
DevHlp_VMProcessToGlobal        EQU     71      ;  47   Create global mapping to process
                                        ;       memory
DevHlp_VMGlobalToProcess        EQU     72      ;  48   Create process mapping to global
                                        ;       memory
DevHlp_VirtToLin        EQU     73      ;  49 Convert virtual address to linear
DevHlp_LinToGDTSelector EQU     74      ;  4A Convert linear address to virtual
DevHlp_GetDescInfo      EQU     75      ;  4B Return descriptor information
DevHlp_LinToPageList    EQU     76      ;  4C build pagelist array from lin addr
DevHlp_PageListToLin    EQU     77      ;  4D map page list array to lin addr
DevHlp_PageListToGDTSelector    EQU     78      ;  4E map page list array to GDT sel.
DevHlp_RegisterTmrDD    EQU     79      ;  4F Register TMR Device Driver.
DevHlp_RegisterPerfCtrs EQU     80      ;  50 Register device driver perf. ctrs (PVW).
DevHlp_AllocateCtxHook  EQU     81      ;  51 Allocate a context hook
DevHlp_FreeCtxHook      EQU     82      ;  52 Free a context hook
DevHlp_ArmCtxHook       EQU     83      ;  53 Arm a context hook
```

```
;*****************************************************************************
;       Copyright (c) Microsoft Corporation 1990
;       All rights reserved.
;
;*****************************************************************************
;*****************************************************************************
; DRP (Driver Registration Packet) Data Structure
;
;*****************************************************************************
;  C Definition
;  Assembly Definition

DRP             STRUC

DRP_layer               dw      ?       ; layer code - see values below.
DRP_ds                  dw      ?       ; data group selector.
DRP_aer                 dd      ?       ; 16:16 ptr to async event routine.
DRP_ilb                 dd      ?       ; 16:16 address of ilb.
DRP_ilb_phys            dd      ?       ; physical address of ilb.
DRP_ascii_name          db      'WAMIC BID String'
DRP_create_date         db      '09/20/90'
DRP_create_time         db      '12:00:00'
DRP_rev_level           db      '0.00'
DRP_feature_code        dd      ?
DRP_if_requirements     dw      ?         ; i/f requirements - values below.
DRP_reg_result          dw      ?
DRP_pv_ptr_off          dd      ?       ; 16:32 pointer to perfview data. must
DRP_pv_ptr_sel          dw      ?       ; be zero if no perfview data exists.
DRP_filler_1            dw      ?       ; reserved - must be zero.
DRP_pv_len                      dw      ?       ; 32-bit size of perfview data. must
                                ; be zero if no perfview data exists.
DRP_pv_basic_str        db      '        '; first eight bytes of name of file
                                  ; which contains the perfview ascii
                                  ; strings for the basic (usually
                                  ; microsoft defined) pv cells. must
                                  ; be zero if no pv data exists. the
                                  ; file type is assumed to be ".BPS".
DRP_pv_extra_str        db      '        '; first eight bytes of name of file
                                  ; which contains the perfview ascii
                                  ; strings for the extrat (usually
                                  ; oem/ihv defined) pv cells. must be
                                  ; zero if no extra pv data exists.
                                  ; the file type is assumed to be
                                  ; ".BPS".

DRP             ENDS


;
;* Feature Code Definitions
DRP_SCSI_ADDR   EQU     1         ;   on = bid uses scsi addressing conventions
DRP_SCSI_UP     EQU     1         ;   on = bid does scsi addressing and the
                                  ;        rom on the adaptor scans up.
DRP_ESDI_SCAN   EQU     2         ;   on = bid uses esdi addressing conventions
DRP_SCAN_DOWN   EQU     4         ;   on = bios scans targets from 7 to 0
                                  ;  off = bios scans targets from 0 to 7
DRP_SCSI_DN     EQU     4         ;   on = bid does scsi addressing and the
                                  ;        rom on the adaptor scans down.
DRP_ABIOS_SCAN  EQU     8         ;   on = bid uses abios addressing/scanning.
;
;* I/F Requirements
DRP_IF_ISA      EQU     0001H   ;  on = driver supports isa platforms
DRP_IF_EISA     EQU     0002H   ;  on = driver supports eisa platforms
DRP_IF_MCA      EQU     0004H   ;  on = driver supports mca platforms
DRP_IF_STD      EQU     00ffH   ;  on = drvr supports all standard platforms
```

```
;
;* driver layer
DRP_IOC EQU     0       ;  this is the ios configuration table
DRP_TSD EQU     1       ;  driver is in type specific layer
DRP_VSD_1       EQU     2       ;  driver is in vendor enhancement layer 1
DRP_VSD_2       EQU     3       ;  driver is in vendor enhancement layer 2
DRP_VSD_3       EQU     4       ;  driver is in vendor enhancement layer 3
DRP_VSD_4       EQU     5       ;  driver is in vendor enhancement layer 4
DRP_VSD_5       EQU     6       ;  driver is in vendor enhancement layer 5
DRP_VSD_6       EQU     7       ;  driver is in vendor enhancement layer 6
DRP_VSD_7       EQU     8       ;  driver is in vendor enhancement layer 7
DRP_VSD_8       EQU     9       ;  driver is in vendor enhancement layer 8
                                ;  vsd level 8 is intended for scsi'izers
DRP_VSD_9       EQU     10       ;  driver is in vendor enhancement layer 9
DRP_PSD EQU     11       ;  driver is in path selection layer
DRP_BID EQU     12       ;  driver is in  bus interface layer
DRP_NON_COM     EQU     13      ;  driver is non compliant
DRP_IORUN       EQU     14      ;  driver is the laddr run layer
DRP_LAYER_MAX   EQU     14      ;  maximum valid layer number
;
;* Registration Results
DRP_REMAIN_RESIDENT     EQU     1       ;  Driver should remain resident
DRP_MINIMIZE    EQU     2       ;  Driver should minimize
DRP_ABORT       EQU     3       ;  Driver should not load
DRP_INVALID_LAYER       EQU     4       ;  bad layer number - abort driver
```

```
;*****************************************************************************
;       Copyright (c) Microsoft Corporation 1990
;       All rights reserved.
;
;*****************************************************************************
;*****************************************************************************
;
; ILB (IOS Linkage Block)
;
;
;*****************************************************************************
;
;*  ILB_flag equates
ILB_286_Mode    EQU     1
;
;*  ILB field lengths
ILB_NAME_LEN    EQU     16
ILB_DATE_LEN    EQU     8
ILB_TIME_LEN    EQU     8
ILB_REV_LEN     EQU     4
ILB_FC_LEN      EQU     4

ILB     STRUC
ILB_strat1_block        DD      ?       ;  strat 1 entry point into ios
ILB_strat2_block        DD      ?       ;  strat 2 entry point into ios
ILB_strat1_char DD      ?        ;  strat 1 entry point into ios
ILB_strat2_char DD      ?        ;  strat 2 entry point into ios
ILB_queue_srb   DD      ?          ;  queue srb entry point into ios
ILB_service_rtn DD      ?        ;  service rtn entry point into ios
ILB_dprintf_rtn DD      ?        ;  dprintf rtn entry point into ios
ILB_eoi_rtn     DD      ?            ;  eoi rtn entry point into ios
ILB_PhysToGDTSel_rtn    DD      ?   ;  PhysToGDTSel rtn entry into ios
ILB_AllocGdtSel_rtn     DD      ?    ;  AllocGdtSel  rtn entry into ios
ILB_ProcBlock_rtn       DD      ?      ;  ProcBlock entry point into ios
ILB_ProcRun_rtn DD      ?        ;  ProcRun entry point into ios
ILB_PhysToVirt_rtn      DD      ?     ;  PhysToVirt entry point into ios
ILB_VirtToPhys_rtn      DD      ?     ;  VirtToPhys entry point into ios
ILB_AllocPhys_rtn       DD      ?      ;  AllocPhys entry point into ios
ILB_Yield_rtn   DD      ?          ;  Yield rtn entry point into ios
ILB_Lock_rtn    DD      ?           ;  Lock entry point into ios
ILB_UnLock_rtn  DD      ?         ;  UnLock entry point into ios
ILB_reserved_12 DD      ?        ;  reserved for future use
ILB_reserved_13 DD      ?        ;  reserved for future use
ILB_reserved_14 DD      ?        ;  reserved for future use
ILB_reserved_15 DD      ?        ;  reserved for future use
ILB_reserved_16 DD      ?        ;  reserved for future use
ILB_reserved_17 DD      ?        ;  reserved for future use
ILB_reserved_18 DD      ?        ;  reserved for future use
ILB_reserved_19 DD      ?        ;  reserved for future use
ILB_reserved_20 DD      ?        ;  reserved for future use
ILB_devhlp      DD      ?            ;  devhelp entry point into ios
ILB_trace_rtn   DD      ?          ;  trace rtn entry point into ios
ILB_dvt DD      ?               ;  32 bit offset of this drivers dvt
ILB_reserved_30 DD      ?        ;  reserved for future use
ILB_reserved_31 DD      ?        ;  reserved for future use
ILB_runtime_cs  DW      ?         ;  driver's run time code selector
ILB_drv_data_sel        DW      ?       ;  driver's data group selector
ILB_ios_mem_sel DW      ?        ;  selector for ios's memory pool
ILB_flags       DW      ?            ;  flags - see defines above
ILB_platform    DW      ?          ;  platform (machine) type mask
ILB_ver_major   DB      ?          ;  Major version number of os/2
ILB_ver_minor   DB      ?          ;  Minor version number of os/2
ILB_reserved_43 DB      ?        ;  reserved for future use
ILB_reserved_44 DB      ?        ;  reserved for future use
ILB reserved 45 DB      ?        :  reserved for future use
```

```
ILB_first_drive DB      ?       ;  unit number of first drive
ILB_reserved_46 DW      ?       ;  reserved: used to be info_seg_sel
ILB     ENDS
```

```
;*****************************************************************************
;       Copyright (c) Microsoft Corporation 1990
;       All rights reserved.
;
;*****************************************************************************
;*****************************************************************************
; Module Name: IOSDEFS.H - IOS Private Definitions file
;
;*****************************************************************************
IOS_IOCTl_Category      EQU     90H
IOS_Reg_Driver_Function EQU     50H
;*****************************************************************************
; C Macros for IOS
;
;*****************************************************************************
```

```
;*****************************************************************************
;       Copyright (c) Microsoft Corporation 1990
;       All rights reserved.
;
;*****************************************************************************
;*****************************************************************************
; ISP (IOS Services Packet) Data Structure
;
;*****************************************************************************
;
;* define the ios service function code
ISP_CREATE_DDB  EQU     0       ;  create ddb
ISP_CREATE_DCB  EQU     1       ;  create dcb (Physical)
ISP_CREATE_RCB  EQU     2       ;  create rcb
ISP_CREATE_XCB  EQU     2       ;  create xcb
ISP_ALLOC_MEM   EQU     3       ;  allocate memory
ISP_SET_IRQ     EQU     4       ;  set irq
ISP_CREATE_SRBS EQU     5       ;  ************* temp *************
ISP_ALLOC_SRBS  EQU     5       ;  allocate srb chain
ISP_DEALLOC_RCB EQU     6       ;  deallocate rcb
ISP_DEALLOC_XCB EQU     6       ;  deallocate xcb
ISP_DEALLOC_MEM EQU     7       ;  deallocate memory
ISP_LOG_DCB_CREATE      EQU     8       ;  create logical dcb
ISP_INSERT_CALLDOWN     EQU     9       ;  insert entry in calldown table
ISP_ASSOCIATE_DCB       EQU     10       ;  associate dcb & relative drive #
ISP_GET_DCB     EQU     11       ;  return DCB for logical volume
ISP_PUT_ILR     EQU     12       ;  put ilr in log buffer
ISP_UNSET_IRQ   EQU     13       ;  deregister the irq
ISP_GET_FIRST_NEXT_DCB  EQU     14       ;  get first/next DCB
;
;* define the basic ios service packet

ISP     STRUC
ISP_func        DW      ?               ;  function to be performed
ISP_owner       DD      ?              ;  cs:ip of caller
ISP_result      DW      ?             ;  result: zero = no error
ISP     ENDS
;
;* define the ios service packet for the create ddb function

ISP_ddb_create  STRUC
ISP_filler_1    DW      ?           ;  "ISP_CREAT_DDBb"
ISP_filler_2    DD      ?           ;  cs:ip of caller
ISP_filler_3    DW      ?           ;  result: zero = no error
ISP_ddb_size    DW      ?           ;  size of ddb to create
ISP_ddb_ptr     DD      ?            ;  32-bit offset to ddb
ISP_ddb_pv_sel  DW      ?         ;  selector of pvw strings
ISP_ddb_pv_str  DD      ?         ;  32-bit offset to pvw strings
ISP_ddb_pv_data DD      ?        ;  32-bit offset to pvw data
ISP_ddb_pv_len  DW      ?         ;  length of perfview data
ISP_ddb_create  ENDS
;
;* define the ios service packet for the create dcb function

ISP_dcb_create  STRUC
ISP_filler_4    DW      ?           ;  "ISP_CREATE_DCB"
ISP_filler_5    DD      ?           ;  cs:ip of caller
ISP_filler_6    DW      ?           ;  result: zero = no error
ISP_dcb_size    DW      ?           ;  size of dcb to create
ISP_dcb_ptr     DD      ?            ;  32-bit offset to dcb
ISP_dcb_create  ENDS
;
;* define the ios service packet for the create rcb function

ISP RCB alloc   STRUC
```

```
ISP_filler_7    DW      ?           ;  "ISP_CREATE_RCB"
ISP_filler_8    DD      ?           ;  cs:ip of caller
ISP_filler_9    DW      ?           ;  result: zero = no error
ISP_RCB_size    DW      ?           ;  size of RCB to allocate
ISP_RCB_ptr     DD      ?           ;  32-bit offset to RCB
ISP_RCB_alloc   ENDS
;
;* define the ios service packet for the create xcb function

ISP_xcb_alloc   STRUC
ISP_filler_7a   DW      ?           ;  "ISP_CREATE_XCB"
ISP_filler_8a   DD      ?           ;  cs:ip of caller
ISP_filler_9a   DW      ?           ;  result: zero = no error
ISP_xcb_size    DW      ?           ;  size of xcb to allocate
ISP_xcb_ptr     DD      ?           ;  32-bit offset to xcb
ISP_xcb_alloc   ENDS
;
;* define the ios service packet for the allocate memory function

ISP_mem_alloc   STRUC
ISP_filler_10   DW      ?           ;  "ISP_ALLOC_MEM"
ISP_filler_11   DD      ?           ;  cs:ip of caller
ISP_filler_12   DW      ?           ;  result: zero = no error
ISP_mem_size    DW      ?           ;  byte size of memory to allocate
ISP_mem_ptr     DD      ?           ;  32-bit offset to memory block
ISP_mem_type    DW      ?           ;  med type code for memory block
ISP_mem_alloc   ENDS
;
;* define the ios service packet for the set irq function (and unset)

ISP_IRQ_set     STRUC
ISP_filler_13   DW      ?           ;  "ISP_SET_IRQ"
ISP_filler_14   DD      ?           ;  cs:ip of caller
ISP_filler_15   DW      ?           ;  result: zero = no error
ISP_IRQ_LEVEL   DB      ?           ;  Interrupt level
ISP_Share_Flag  DB      ?           ;  IRQ share flag
ISP_IRQ_Data    DD      ?           ;  IRQ Data
ISP_IRQ_Handler DD      ?           ;  Address of IRQ service routine
ISP_IRQ_set     ENDS
;
;* define the ios service packet for the allocate srb chain function

ISP_srb_alloc   STRUC
ISP_filler_16   DW      ?           ;  "ISP_ALLOC_SRBS"
ISP_filler_17   DD      ?           ;  cs:ip of caller
ISP_filler_18   DW      ?           ;  result: zero = no error
ISP_srb_number  DW      ?           ;  number of chained srbs to alloc
ISP_srb_size    DW      ?           ;  size of srb to allocate
ISP_srb_ptr     DD      ?           ;  32-bit offset to 1st srb
ISP_srb_alloc   ENDS
;
;* define the ios service packet for the de-allocate rcb function

ISP_rcb_dealloc STRUC
ISP_filler_20   DW      ?           ;  "ISP_DEALLOC_RCB"
ISP_filler_21   DD      ?           ;  cs:ip of caller
ISP_filler_22   DW      ?           ;  result: zero = no error
ISP_rcb_ptr_da  DD      ?           ;  32-bit offset to rcb to dealloc
ISP_rcb_dealloc ENDS
;
;* define the ios service packet for the de-allocate xcb function

ISP_xcb_dealloc STRUC
ISP_filler_20a  DW      ?           ;  "ISP_DEALLOC_XCB"
ISP_filler_21a  DD      ?           ;  cs:ip of caller
```

```
ISP_filler_22a  DW      ?         ;  result: zero = no error
ISP_xcb_ptr_da  DD      ?         ;  32-bit offset to xcb to dealloc
ISP_xcb_dealloc ENDS
;
;* define the ios service packet for the de-allocate memory function

ISP_mem_dealloc STRUC
ISP_filler_23   DW      ?         ;  "ISP_DEALLOC_MEM"
ISP_filler_24   DD      ?         ;  cs:ip of caller
ISP_filler_25   DW      ?         ;  result: zero = no error
ISP_mem_ptr_da  DD      ?         ;  32-bit offset to memry to dealloc
ISP_mem_dealloc ENDS
;
;* define the ios service packet for the create logical dcb from physical
;* dcb function

ISP_dcb_log_create      STRUC
ISP_filler_26   DW      ?         ;  "ISP_LOG_DCB_CREATE"
ISP_filler_27   DD      ?         ;  cs:ip of caller
ISP_filler_28   DW      ?         ;  result: zero = no error
ISP_dcb_phys    DD      ?          ;  32-bit offset of physical dcb
ISP_dcb_log     DD      ?          ;  32-bit offset to dcb
ISP_dcb_log_create      ENDS
;
;* define the ios service packet for the insert calldown entry function

ISP_calldown_insert     STRUC
ISP_filler_29   DW      ?         ;  "ISP_INSERT_CALLDOWN"
ISP_filler_30   DD      ?         ;  cs:ip of caller
ISP_filler_31   DW      ?         ;  result: zero = no error
ISP_i_cd_dcb    DD      ?          ;  32-bit offset to dcb
ISP_i_cd_req    DD      ?          ;  16:16 pointer to request routine
ISP_i_cd_aer    DD      ?          ;  16:16 pointer to aer
ISP_i_cd_ddb    DD      ?          ;  32-bit offset to ddb
ISP_i_cd_pv_sel DW      ? ;  16-bit selector of perfview data
ISP_i_cd_pv_str DD      ?        ;  32-bit offset to pvw strings
ISP_i_cd_pv_ptr DD      ?        ;  32-bit offset to perfview hdr -
                                     ;  zero if none.
ISP_i_cd_pv_len DW      ?        ;  length of perfview data - zero
                                     ;  if none.
ISP_i_cd_flags  DW      ?         ;  demand flag bits
ISP_calldown_insert     ENDS
;
;* define the ios service packet for the associate dcb and relative drive
;* number function

ISP_dcb_associate       STRUC
ISP_filler_32   DW      ?         ;  "ISP_ASSOCIATE_DCB"
ISP_filler_33   DD      ?         ;  cs:ip of caller
ISP_filler_34   DW      ?         ;  result: zero = no error
ISP_d_a_dcb     DD      ?          ;  32-bit offset of dcb
ISP_d_a_drive   DB      ?         ;  relative drive number
ISP_dcb_associate       ENDS
;
;* define the ios service packet for the associate dcb and relative drive
;* number function

ISP_dcb_get     STRUC
ISP_filler_35   DW      ?         ;  "ISP_GET_DCB"
ISP_filler_36   DD      ?         ;  cs:ip of caller
ISP_filler_37   DW      ?         ;  result: zero = no error
ISP_g_d_dcb     DD      ?          ;  32-bit offset of dcb
ISP_g_d_drive   DB      ?         ;  relative drive number
ISP_dcb_get     ENDS
;
```

```
;* define the ios service packet for the log ilr function

ISP_ILR_PUT     STRUC
ISP_filler_38   DW      ?           ;  "ISP_PUT_ILR"
ISP_filler_39   DD      ?           ;  cs:ip of caller
ISP_filler_40   DW      ?           ;  result: zero = no error
ISP_ilr_sel     DW      ?             ;  16-bit selector of ilr
ISP_ilr_offset  DD      ?         ;  32-bit offset of ilr
ISP_ILR_PUT     ENDS
;
;* define the ios service packet for the get first/next DCB function

ISP_GET_FRST_NXT_DCB    STRUC
ISP_filler_41   DW      ?           ;  "ISP_GET_DCB"
ISP_filler_42   DD      ?           ;  cs:ip of caller
ISP_filler_43   DW      ?           ;  result: zero = dcb found
ISP_gfnd_dcb_offset     DD      ?    ;  32-bit offset of DCB from which
                                       ;  to get next or zero to get first.
ISP_gfnd_found_dcb      DD      ?     ;  if result = zero: dcb found by
                                       ;  IOS. undefined if result not zero
ISP_gfnd_dcb_type       DB      ?      ;  acceptable device type or 0ffh
                                       ;  for any type. see DCB.H for
                                       ;  valid device type codes.
ISP_GET_FRST_NXT_DCB    ENDS
```

```
;*****************************************************************************
;       Copyright (c) Microsoft Corporation 1990
;       All rights reserved.
;
;*****************************************************************************
;*****************************************************************************
; MED (Memory Element Descriptor) Data Structure
;
;*****************************************************************************

MED     STRUC
MED_Physical    DD      ?           ;  Physical Pointer to MED Struct
MED_Owner       DD      ?              ;  CS:IP of requestor
MED_Next        DD      ?               ;  Next Logical MED
MED_Adjacent    DD      ?           ;  Next Physical MED
MED_Type        DW      ?               ;  Type of data structure
MED_Size        DW      ?               ;  Size of piece (including med)
MED_Flags       DW      ?              ;  Flags
MED_CRC16       DW      ?              ;  16 bit Cyclic Redundancy Check
MED     ENDS
;
;* WAMIC Data Structure defines (MED_Flags)
MED_FREE_POOL   EQU     0001H           ;  piece is free and in the pool
MED_FREE_FAST   EQU     0002H           ;  piece is free for quick allocate
MED_ALLOC       EQU     0004H           ;  this piece is allocated
;
;* WAMIC Data Structure defines (MED Type)
MED_FREE_TYPE   EQU     0               ;  piece is not being used - do not
                                        ;  use this as a test for the piece
                                        ;  being free. use the flags

MED_RCB EQU     1               ;  RCB DataType
MED_RLH EQU     2               ;  RLH DataType
MED_RH  EQU     3               ;  RH  DataType
MED_SRB EQU     4               ;  SRB DataType
MED_ILB EQU     5               ;  ILB DataType
MED_AEP EQU     6               ;  AEP DataType
MED_DCB EQU     7               ;  DCB DataType
MED_DDB EQU     8               ;  ddb DataType
MED_DRP EQU     9               ;  DRP DataType
MED_DVT EQU     10               ;  DVT datatype
MED_IRB EQU     11               ;  IRB datatype
MED_MISC        EQU     12              ;  Misc datatype
MED_ILR EQU     13              ;  ios log record
MED_IOC EQU     14              ;  ios configuration file
MED_QLINK       EQU     15              ;  qlink
MED_NEW_RCB     EQU     16              ;  type used to alloc rcb's due to timer issues
;
;* constants for use only by the memory allocation/deallocation routines
MED_GRAN_DELTA  EQU     16              ;  delta from one granule size to
                                        ;  the next. this number must be a
                                        ;  power of two.
MED_FAST_GRANS  EQU     64              ;  number of granules in the fast
                                        ;  allocation table.
MED_SHIFT_CNT   EQU     4 -2            ;  shift count equivalent to
                                        ;  dividing by MED_GRAN_DELTA and
                                        ;  multiplying by the number of
                                        ;  bytes in a dword (ie, four).
```

```
;*****************************************************************************
;       Copyright (c) Microsoft Corporation 1990
;       All rights reserved.
;
;*****************************************************************************
;*****************************************************************************
; RCB (Request Control Block) Data Structure
;
;*****************************************************************************
RCB_CALLBACK_TABLE_DEPTH        EQU     6
;
;* CallBack Table Entry

RCB_CallBack_Entry      STRUC
RCB_CB_Address  DD      ?         ;  16:16 Call Back Address
RCB_CB_DDB_Offset       DD      ?       ;  DDB 32 bit Offset
RCB_CB_DDB_Segment      DW      ?     ;  DDB 16 bits Segment
RCB_CallBack_Entry      ENDS
;
;* RCB Typedef

RCB     STRUC
RCB_Phys_Addr   DD      ?                  ;  Physical Pointer to RCB Struct
RCB_physical_dcb        DD      ?               ;  32 bit pointer to physical DCB
RCB_original_dcb        DD      ?               ;  ptr to dcb associated with unit
                                                                        ;  designated by original strat1/
                                                                        ;  request. note that this could
                                                                        ;  either a logical or physical c

RCB_SRB_Logical DD      ?        ;  SRB Logical Pointer
RCB_SRB_Physical        DD      ?       ;  SRB Physical Pointer
RCB_RLH_Segment DW      ?        ;  RLH Segment (may be GDT)
RCB_RLH_Offset  DD      ?         ;  RLH Offset (32 bit)
RCB_RLH_Physical        DD      ?       ;  RLH Physical Pointer
RCB_RCB_Logical_FLink   DD      ?  ;  RCB Logical - Forward Link
RCB_RCB_Physical_FLink  DD      ? ;  RCB Physical -Forward Link
RCB_RCB_Logical_BLink   DD      ?  ;  RCB Logical - Backward Link
RCB_Exception_Routine   DD      ?  ;  Cur RCB Owner Except Handler Addr
RCB_rp  DD      ?                 ;  16:16 pointer of associated rp
RCB_Number_RH_in_RLH    DW      ?   ;  Number of RH's in current RLH
RCB_Function    DW      ?           ;  RCB Function
RCB_Flags       DW      ?             ;  RCB Flags
RCB_timer       DW      ?             ;  RCB current timeout value
RCB_timer_orig  DW      ?         ;  RCB original timeout value
RCB_Status      DW      ?            ;  RCB Status Word
RCB_Countdown   DW      ?          ;  Counter for tracking RH completion
RCB_ft_prim_part        DB      ?       ;  Primary partition number for FT
RCB_ft_sec_part DB      ?        ;  Secondary partition number for FT
RCB_ft_req_handle       DW      ?       ;  FT Request Handle
RCB_ft_prim_rcb DD      ?        ;  32-bit offset for FT primary RCB
RCB_ft_sec_rcb  DD      ?         ;  32-bit offset of FT secondary RCB
RCB_CallDownPtr DD      ?        ;  ptr to next calldown routine
RCB_CallBackPtr DD      ?        ;  ptr to current callback address
RCB_CallBackTable       DB      SIZE RCB_CallBack_Entry * RCB_CALLBACK_TABLE_DEPTH DUP (?)
                                                                        ;  Call Back Stack Addres

RCB     ENDS
;
;* RCB Function Code
RCB_EXECUTE_IO  EQU     001Q    ;  Execute IO FN Code
RCB_RESET       EQU     002Q    ;  Reset FN Code
RCB_ABORT       EQU     003Q    ;  Abort FN Code
;
;* RCB Flags
CALLBACK_ENABLE EQU     0001H   ;  CallBack Address is valid
NON_SORTABLE_RCB        EQU     0002H   ;  Non Sortable RCB
RCB SKIP RLH RH CB      EOU     0004H   ;  inhibit rh and rlh callbacks in
```

```
                                        ;  layers other than that which set
                                        ;  this flag and store status in
                                        ;  RH_alt_status.
;
;* RCB Request Control Flags (RLH_Request_Control)
;
;*****************************************************************
;*
;* Please use RLH Defines .... as seen in RLH.H                    *
;* Please use RLH Defines .... as seen in RLH.H                    *
;* Please use RLH Defines .... as seen in RLH.H                    *
;* Please use RLH Defines .... as seen in RLH.H                    *
;* Please use RLH Defines .... as seen in RLH.H                    *
;* Please use RLH Defines .... as seen in RLH.H                    *
;*
;*****************************************************************
```

```
;*****************************************************************************
;       Copyright (c) Microsoft Corporation 1990
;       All rights reserved.
;
;*****************************************************************************
;*****************************************************************************
; RLH (STRAT2 packet) Data Structure
;
;*****************************************************************************
;  equivalent names:
;
;  old rlh.h/strat2.h   old rlx.h              new rlh.h/rlz.h
;  ==================   ==================     ==================
;  rlh_count            rlx_count_lo           RLH_count_lo
;  rlh_short_count      rlx_count_lo           RLH_count_lo
;  rlh_notify_address   rlx_notify_addr        RLH_notify_addr
;  rlh_request_control  rlx_request_ctl        RLH_request_ctl
;  rlh_block_dev_unit   rlx_drive_number       RLH_drive_number
;  rlh_lst_status       rlx_status             RLH_status
;  rlh_y_done_count     rlx_done_count         RLH_done_count
;  rlh_count_done       rlx_done_count         RLH_done_count
;  rlh_queued           n/a                    RLH_next_rh_numb
;
;  rh_length            rx_length              RH_delta_next
;  rh_old_command       rx_func                RH_func
;  rh_command_code      rx_command             RH_sub_func
;  rh_head_offset       rx_delta_rlh           RH_delta_rlh
;  rh_req_control       rx_req_control         RH_control
;  rh_priority          rx_priority            RH_priority
;  rh_status            rx_status              RH_status
;  rh_error_code        rx_error_code          RH_error_code
;  rh_notify_address    rx_notify_addr         RH_notify_addr
;  rh_hint_pointer      n/a                    n/a
;  rh_waiting           rx_waiting             RH_waiting
;  rh_ft_orig_pkt       rx_ft_orig_pkt         RH_ft_orig_pkt
;  rh_physical_rba      rx_physical_rba        RH_physical_rba
;  pb_start_block       rx_start_block         RH_starting_rba
;  pb_block_count       rx_block_count         RH_block_count
;  pb_blocks_xferred    rx_blocks_xferrd       RH_blocks_xferrd
;  pb_rw_flags          rx_rw_flags            RH_rw_flags
;  pb_sg_desc_count     rx_sg_desc_count       RH_sg_count
;  pb_sg_desc_count2    n/a                    n/a
;                       rx_sg_buff_ptr         RH_sg.SG_buff_ptr
;                       rx_sg_buff_size        RH_sg.SG_buff_size
;  pb_sg_array_offset                          RH_sg
;
;  pb_read_x                                   RH_read_x
;  pb_write_x                                  RH_write_x
;  pb_writev_x                                 RH_writev_x
;  pb_prefetch_x                               RH_prefetch_x
;  verify_x                                    RH_verify_x
;
;  rw_cache_writethru                          RH_rw_cache_write_thru
;  rw_cache_req                                RH_rw_cache_req
;

RLH     STRUC
RLH_count_lo    DW      ?     ;  number of requests in Req List
RLH_reserved_1  DW      ?   ;  reserved - must be zero
RLH_notify_addr DD      ?  ;  16:16 address of notification routine
RLH_request_ctl DW      ?  ;  bitfield of flags defined below
RLH_drive_number        DB      ? ;  logical unit number of volume
RLH_status      DB      ?       ;  overall status for Req List
RLH_done_count  DW      ?   ;  count of request completed (internal)
RLH_next_rh_numb        DW      ? :  number of next rh toprocess (internal)
```

```
RLH     ENDS
RLH_SIZE        EQU     size RLH

RLH_special     STRUC
RLH_reserved_11 DW      ?  ;  number of requests in Req List
RLH_reserved_12 DW      ?  ;  reserved - must be zero
RLH_reserved_13 DD      ?  ;  16:16 address of notification routine
RLH_reserved_14 DW      ?  ;  bitfield of flags defined below
RLH_reserved_15 DB      ?  ;  logical unit number of volume
RLH_reserved_16 DB      ?  ;  overall status for Req List
RLH_reserved_17 DW      ?  ;  count of request completed (internal)
RLH_reserved_18 DW      ?  ;  number of next rh to process (internal)
RLH_reserved_19 DW      ?  ;  offset of the next request
RLH_reserved_20 DB      ?  ;  reserved, always 1Ch, same offset
                              ;  as command code in OS/2 req header
RLH_reserved_21 DB      ?  ;  Pinball command request code
RLH_reserved_22 DD      ?  ;  offset from begin of Req List Header
RLH_reserved_23 DB      ?  ;  control flags bits defined below
RLH_reserved_24 DB      ?  ;  Priority of request defined below
RLH_reserved_25 DB      ?  ;  status bitfield defined below
RLH_reserved_26 DB      ?  ;  Pinball errors defined below
RLH_reserved_27 DD      ?  ;  16:16 address called when done
RLH_reserved_28 DD      ?  ;  reserved - must be zero
RLH_rcb_ptr     DD      ?     ;  pointer to owning rcb (internal)
RLH_special     ENDS

RH      STRUC
RH_delta_next   DW      ?    ;  offset of the next request
RH_func DB      ?          ;  reserved, always 1Ch, same offset
                              ;  as command code in OS/2 req header
RH_sub_func     DB      ?     ;  Pinball command request code
RH_delta_rlh    DD      ?     ;  offset from begin of Req List Header
RH_control      DB      ?      ;  control flags bits defined below
RH_priority     DB      ?      ;  Priority of request defined below
RH_status       DB      ?       ;  status bitfield defined below
RH_error_code   DB      ?    ;  Pinball errors defined below
RH_notify_addr  DD      ?   ;  16:16 address called when done
RH_reserved_1   DD      ?    ;  reserved - must be zero
RH_waiting      DD      ?      ;  Waiting queue link pointer (internal)
RH_alt_status   DB      ?    ;  used instead of RH_status when
                              ;  RCB_SKIP_RLH_RH_CB is on (internal).
RH_reserved_4   DB      3 DUP (?) ;  reserved - must be zero (internal)
RH_physical_rba DD      ?  ;  logical plus parttion offset (internal)
RH_starting_rba DD      ?  ;  start block for data transfer
RH_block_count  DD      ?   ;  number of blocks to transfer
RH_blocks_xferrd        DD      ? ;  number of blocks transferred
RH_rw_flags     DW      ?      ;  command specific control flags
RH_sg_count     DW      ?      ;  number of SG descriptors
RH_reserved_2   DW      ?    ;  reserved - must be zero    (internal)
RH_reserved_3   DW      ?    ;  reserved for dd alignment  (internal)
RH_sg   DB      SIZE SGD DUP (?)            ;  first scatter/gather descriptor
RH      ENDS
;
;* rh for use with ioctl's

RH_IOCTL        STRUC
RH_filler_1     DW      ?      ;  offset of the next request
RH_filler_2     DB      ?      ;  reserved, always 1Ch, same offset
                              ;  as command code in OS/2 req header
RH_filler_3     DB      ?      ;  Pinball command request code
RH_filler_4     DD      ?      ;  offset from begin of Req List Header
RH_rp_16_cat    DB      ?     ;  ioctl category code
RH_rp_16_func   DB      ?    ;  ioctl function code
RH_filler_5     DB      ?      ;  status bitfield defined below
RH filler 6     DB      ?      :  Pinball errors defined below
```

```
RH_filler_7     DD      ?       ;  16:16 address called when done
RH_rp_16_parm_ptr       DD      ? ;  16:16 parameter packet address
RH_rp_16_data_ptr       DD      ? ;  16:16 data buffer address
RH_filler_8     DD      ?       ;  32b ptr to original request(internal)
RH_filler_9     DD      ?       ;  logical plus parttion offset (internal)
RH_filler_10    DD      ?      ;  start block for data transfer
RH_parm_lock    DD      ?      ;  lock handle for request parm packet
RH_data_lock    DD      ?      ;  lock handle for request data buffer
RH_filler_13    DW      ?      ;  command specific control flags
RH_filler_14    DW      ?      ;  number of SG descriptors
RH_filler_15    DW      ?      ;  reserved - must be zero    (internal)
RH_filler_16    DW      ?      ;  reserved for dd alignment  (internal)
RH_filler_17    DB      SIZE SGD DUP (?)     ;  first scatter/gather descriptor
RH_IOCTL        ENDS
;
;* rlh control bit definitions for RLH_request_ctl
RLH_REQ_FROM_PB EQU     0001H   ;  Request came directly from Pinball
RLH_SINGLE_REQ  EQU     0002H   ;  Single request in list
RLH_EXE_REQ_SEQ EQU     0004H   ;  Requests to be executed in sequence
RLH_ABORT_ERR   EQU     0008H   ;  Abort on error
RLH_NOTIFY_ERR  EQU     0010H   ;  Notify immediately on error
RLH_NOTIFY_DONE EQU     0020H   ;  Notify on completion
;
;* rlh status bit definitions for high nibble of RLH_status
;*       - high nibble indicates error status of reqs in list
RLH_NO_ERROR    EQU     00H  ;  No error
RLH_REC_ERROR   EQU     10H  ;  Recoverable error has occurred
RLH_UNREC_ERROR EQU     20H  ;  Unrecoverable error has occurred
RLH_UNREC_ERROR_RETRY   EQU     30H  ;  Unrecoverable error after retry
;
;* rlh status bit definitions for low nibble of RLH_status
;*       - low nibble indicates completion status of reqs
RLH_NO_REQ_QUEUED       EQU     00H  ;  No requests queued
RLH_REQ_Not_QUEUED      EQU     01H  ;  Some, but not all, requests queued
RLH_ALL_REQ_QUEUED      EQU     02H  ;  All requests queued
RLH_ALL_REQ_DONE        EQU     04H  ;  All requests done or aborted
RLH_SEQ_IN_PROCESS      EQU     08H  ;  Requests being processed in sequence
RLH_ABORT_PENDINGS      EQU     08H  ;  Abort list processing in progress
;
;* values for the high nibble of RH_status
RH_NO_ERROR     EQU     00H  ;  No error
RH_RECOV_ERROR  EQU     10H  ;  A recoverable error has occurred
RH_UNREC_ERROR  EQU     20H  ;  An unrecoverable error has occurred
RH_UNREC_ERROR_RETRY    EQU     30H  ;  An unrecoverable error with retry
RH_ABORTED      EQU     40H  ;  The request was aborted
;
;* values for the low nibble of RH_status
RH_NOT_QUEUED   EQU     00H  ;  not yet queued
RH_QUEUED       EQU     01H  ;  queued and waiting
RH_PROCESSING   EQU     02H  ;  in process
RH_DONE EQU     04H  ;  done
;
;* values for RH_sub_func
RH_READ_X       EQU     1EH     ;  pinball read
RH_WRITE_X      EQU     1FH     ;  pinball write
RH_WRITEV_X     EQU     20H     ;  pinball write/verify
RH_PREFETCH_X   EQU     21H     ;  pinball prefetch read
RH_VERIFY_X     EQU     22H     ;  verify command
RH_IOCTL_X      EQU     10H     ;  IOCTL
;
;* values for RH_control
RH_PB_REQUEST   EQU     01H  ;  Request came directly from Pinball
RH_NOTIFY_ERROR EQU     10H  ;  Notify on Error
RH_NOTIFY_DONE  EQU     20H  ;  Notify on completion
;
```

```
;* values for RH_rw_flags
RH_RW_CACHE_WRITE_THRU  EQU     0001H   ;  Cache write thru
RH_RW_CACHE_REQ EQU     0002H   ;  Cache the request
;
;* Priorities
PRIO_PREFETCH   EQU     00H     ;  Prefetch requests.
PRIO_LAZY_WRITE EQU     01H     ;  Lazy writer.
PRIO_PAGER_READ_AHEAD   EQU     02H     ;  Read ahead, lo priority pager I/O
PRIO_BACKGROUND_USER    EQU     04H     ;  Background synchronous user I/O.
PRIO_FOREGROUND_USER    EQU     08H     ;  Foreground synchronous user I/O.
PRIO_PAGER_HIGH EQU     10H     ;  High priority pager I/O.
PRIO_URGENT     EQU     80H     ;  Urgent (e.g. power fail).
```

```
;*****************************************************************************
;       Copyright (c) Microsoft Corporation 1990
;       All rights reserved.
;
;*****************************************************************************
;*****************************************************************************
;       General SCSI Definitions
;
;*****************************************************************************
CDB_6_byte      EQU     6       ;  Length of 6 byte CDB
CDB_10_byte     EQU     10      ;  Length of 10 byte CDB
SCSI_Test_Unit_Ready    EQU     00H     ;  Test Unit Ready
SCSI_Rezero_Unit        EQU     01H     ;  CD-ROM rezero unit
SCSI_Rewind     EQU     01H     ;  Tape Rewind
SCSI_Request_Sense      EQU     03H     ;  Request Sense Command
SCSI_Read_Blk_Limits    EQU     05H     ;  Read Block Limits
SCSI_Req_Aux_Sense      EQU     06H     ;  Request Auxiliary Sense
SCSI_Read_6     EQU     08H     ;  SCSI 6 byte Read
SCSI_Write_6    EQU     0AH     ;  SCSI 6 byte Write
SCSI_Write_Filemarks    EQU     10H     ;  Tape Write Filemarks
SCSI_Space      EQU     11H     ;  Tape Space
SCSI_Inquiry    EQU     12H     ;  Inquiry command
SCSI_Recover_Buffer     EQU     14H     ;  Tape Recover Buffer
SCSI_Mode_Select        EQU     15H     ;  Mode Select
SCSI_Reserve_Unit       EQU     16H     ;  Tape Reserve Unit
SCSI_Release_Unit       EQU     17H     ;  Tape Release Unit
SCSI_Erase      EQU     19H     ;  Tape Erase
SCSI_Mode_Sense EQU     1AH     ;  Mode Sense
SCSI_Start_Stop_Unit    EQU     1BH     ;  Start/Stop Unit
SCSI_Load_Unload        EQU     1BH     ;  Tape Load/Unload Media
SCSI_Lock_Unlock        EQU     1EH     ;  Lock/Unlock drive door
SCSI_Read_Capacity      EQU     25H     ;  Read Capacity
SCSI_Read_10    EQU     28H     ;  SCSI 10 byte Read
SCSI_Write_10   EQU     2AH     ;  SCSI 10 byte Write
SCSI_Seek_10    EQU     2BH     ;  SCSI 10 byte Seek
SCSI_Locate     EQU     2BH     ;  Tape Locate
SCSI_Write_Verify_10    EQU     2EH     ;  SCSI 10 byte Write w/Verify
SCSI_Verify_10  EQU     2FH     ;  SCSI 10 byte Verify
SCSI_Read_Sub_Chan      EQU     42H     ;  Read Sub-Channel (CD-ROM)
SCSI_Read_TOC   EQU     43H     ;  Read Table of Contents
SCSI_Play_MSF   EQU     47H     ;  Play Audio - MSF format
SCSI_Pause_Resume       EQU     4BH     ;  Pause/Resume Audio Play
; SCSI Status Codes
CHECK_CONDITION EQU     2Q      ;  SCSI Check condition
TARGET_BUSY     EQU     08      ;  SCSI Busy status
; Sense Key Values on Check Condition
SENSE_NO_SENSE  EQU     00H     ;  No error occured
RECOVERED_ERROR EQU     01H     ;  Recovered Error
NOT_READY       EQU     02H     ;  Device not ready
MEDIUM_ERROR    EQU     03H     ;  Medium error detected
HARDWARE_ERROR  EQU     04H     ;  Hardware error detected
ILLEGAL_REQUEST EQU     05H     ;  Illegal request
UNIT_ATTENTION  EQU     06H     ;  Unit Attention
ABORTED_ERROR   EQU     0BH     ;  Aborted error detected
```

scsidefs

```
;******************************************************************************
;       Copyright (c) Microsoft Corporation 1990
;       All rights reserved.
;
;******************************************************************************
;******************************************************************************
; SGD - scatter/gather descriptor
;
;******************************************************************************
;  equivalent names:
;
;  old rlh.h/strat2.h   old rlx.h              new sgd.h
;  ==================   ==================     ==================
;  sg_bufferptr                                sg_buffer_ptr
;  sg_buffersize                               sg_buffer_size

SGD     STRUC
SG_buff_ptr     DD      ?     ;  32 bit physcial pointer to the buffer
SG_buff_size    DD      ?     ;  size of the buffer in bytes
SGD     ENDS
```

```
;*****************************************************************************
;       Copyright (c) Microsoft Corporation 1990
;       All rights reserved.
;
;*****************************************************************************
;*****************************************************************************
; SRB (SCSI Request Block) Data Structure
;
;*****************************************************************************
Max_CDBLen      EQU     24              ;  size of the cdb in the srb
;
;* SRB Status Block Area

SRBStat STRUC
SCSI_Status     DB      ?             ;  SCSI Bus Status
Adapter_Status  DB      ?           ;  HBA Adapter Status
Number_of_Retries       DW      ?       ;  Number of Retries
Residual_Length DD      ?         ;  Residual Length
SRBStat ENDS
;
;* Adapter Status Definitions
No_adapter_error        EQU     0         ;  No adapter error occurred
Selection_Timeout       EQU     1         ;  No response on selection
HW_Failure      EQU     2         ;  Adapter H/W failed
Data_Overrun    EQU     3         ;  data overrun
Data_Underrun   EQU     4         ;  data underrun
;
;* SCSI Status Definitions
Good_Status     EQU     00H     ;  no error reported
Check_condition EQU     02H     ;  Check condition reported
Condtion_met    EQU     04H     ;  Condtion met, Good status
Busy    EQU     08H     ;  Target Busy
Intermidiate_Good       EQU     14H     ;  Good status during likned CDBs
Reservation_conflict    EQU     18H     ;  Reservation conflict
Queue_Full      EQU     28H     ;  Queue for the device is full
;
;* Request State Definitions SRB_Request_State - low nibble
SRB_NOT_QUEUED  EQU     00H             ;  not yet queued
SRB_QUEUED      EQU     01H             ;  queued and waiting
SRB_PROCESSING  EQU     02H             ;  in process
SRB_DONE        EQU     04H             ;  done
;
;* Request State Definition  SRB_Request_State - high nibble
SRB_NO_ERROR    EQU     00H         ;  No error
SRB_ERROR       EQU     30H         ;  An SRB Completed with error
SRB_ABORTED     EQU     40H         ;  The request was aborted
;
;* define the cdb (command descriptor block) area

CDB     STRUC
CDB_Command     DB      ?             ;  CDB Command Byte
CDB_Stuff       DB      Max_CDBLen-1 DUP (?) ;  CDB area
CDB     ENDS
;
;* define the srb

SRB     STRUC
SRB_phys_addr   DD      ?           ;  Physical Pointer to SRB
SRB_filler_1    DW      ?           ;  padding for dword alignment
SRB_function    DB      ?           ;  SRB Function
SRB_request_state       DB      ?       ;  SRB State
SRB_rcb_logical DD      ?         ;  Logical RCB Pointer
SRB_flags_long  DD      ?         ;  SRB Flags
SRB_builder_area        DD      ?       ;  DSD/VSD Work Area
SRB next sortable       DD      ?       :  Next Sortable SRB linkage
```

```
SRB_next_nonsort        DD      ?       ;  Next Non-sortable SRB linkage
SRB_callback    DD      ?           ;  16:16 Callback Routine
SRB_sg_physical DD      ?        ;  Scatter/Gather Physical Pointer
SRB_data_xfer_length    DD      ?   ;  Data Transfer Length
SRB_sense_buffer_physical       DD      ? ;  Sense Buffer pointer physical
SRB_sense_buffer_length DB      ? ;  Length of Sense Buffer
SRB_cdb_length  DB      ?         ;  CDB Length
SRB_num_sg_entries      DW      ?     ;  Number of SG Entries
SRB_dcb DD      ?                ;  32 bit ptr to associated dcb
SRB_status      DB      SIZE SRBStat DUP (?)            ;  SRB Status
SRB_cdb DB      SIZE CDB DUP (?)                ;  CDB
SRB_rba DD      ?                ;  starting relative block address
SRB_owner_linkage       DD      ?      ;  SRB Owner Linkage field
SRB_sg_off      DD      ?             ;  16:32 Scatter Gather pointer
SRB_sg_sel      DW      ?
SRB_filler_6    DB      ?           ;  padding for dword alignment
SRB_filler_7    DB      ?           ;  padding for dword alignment
SRB_rh_off      DD      ?             ;  16:32 RH pointer
SRB_rh_sel      DW      ?
SRB_priority    DB      ?           ;  request's priority
SRB_filler_8    DB      ?           ;  padding for dword alignment
SRB     ENDS
;
;* SRB Function Definitions
SCSI_IO EQU     0        ;  Execute SCSI I/O
Dev_Reset       EQU     1        ;  Reset SCSI device
Req_Abort       EQU     2        ;  Abort SCSI Request
SRB_SCSI_IO     EQU     0        ;  Execute SCSI I/O
SRB_DEV_RESET   EQU     1        ;  Reset SCSI device
SRB_REQ_ABORT   EQU     2        ;  Abort SCSI Request
SRB_IOCTL       EQU     4   ;  RLH contains IOCTL
;
;* SRB Flag Definitions
SRB_DISABLE_CALLBACK    EQU     0008H  ;  Callback on Completion Disabled
SRB_DISABLE_AUTOSENSE   EQU     0020H  ;  Auto Request Sense Disabled
SRB_XFER_FROM_HOST      EQU     0080H  ;  Direction of Xfer - From Host
SRB_XFER_TO_HOST        EQU     0040H  ;  Direction of Xfer - To Host
SRB_DISABLE_DISCONNECT  EQU     8000H  ;  Disconnection Disabled
SRB_STATUS_ONLY_ON_ERROR        EQU     1000H  ;  Update Status ONLY on Error
```