# Example Code

Use the examples in this section as guides in creating your own modules; they should not be considered the most current software. Software for your individual system may be different.

## Init Module

```
Microware OS-9/68020 Resident Macro Assembler V2.9  90/09/10  19:55  Page     1
Init: OS-9 Configuration Module -
00001                    nam      Init: OS-9 Configuration Module
00048 *
00049  00000016 Edition    equ      22           current edition number
00050
00051  00000c00 Typ_Lang   set      (Systm<<8)+0
00052  00008000 Attr_Rev   set      (ReEnt<<8)+0
00053                    psect    init,Typ_Lang,Attr_Rev,Edition,0,0
00054
00055 * Configuration constants (default; changable in "systype.d" file)
00056 *
00057 * Constants that use VALUES (e.g. CPUTyp set 68020) may appear anywhere
00058 * in the "systype.d" file.
00059 * Constants that use LABELS (e.g. Compat set ZapMem) MUST appear OUTSIDE
00060 * the CONFIG macro and must be conditionalized such that they are
00061 * only invoked when this file (init.a) is being assembled.
00062 * If they are placed inside the CONFIG macro, then the over-ride will not
00063 * take effect.
00064 * If they are placed outside the macro and not conditionalized then
00065 * "illegal external reference" errors will result when making other files.
00066 * The label _INITMOD provides the mechanism to ensure that the desired
00067 * operations will result.
00068 *
00069 * example systype.d setup:
00070 *
```

```
00071 * CONFIG macro
00072 *   <body of macro>
00073 *  endm
00074 *  Slice set 10
00075 *  ifdef _INITMOD
00076 * Compat set ZapMem patternize memory
00077 *  endc
00078 *
00079
00080 * flag reading init module (so that local labels can be over-ridden)
00081  00000001 _INITMOD   equ    1            flag reading init module
00082
00083 000109a0 CPUTyp     set    68000       cpu type (68008/68000/68010/etc.)
00084 00000001 Level      set    1           OS-9 Level One
00085 00000002 Vers       set    2           Version 2.4
00086 00000004 Revis      set    4
00087 00000001 Edit       set    1           Edition
00088 00000000 IP_ID      set    0           interprocessor identification code
00089 00000000 Site       set    0           installation site code
00090 00000080 MDirSz     set    128         initial mod directory size (unused)
00091 00000020 PollSz     set    32          IRQ polling table size (fixed)
00092 00000020 DevCnt     set    32          device table size (fixed)
00093 00000040 Procs      set    64          initial process table size
00094 00000040 Paths      set    64          initial path table size
00095 00000002 Slice      set    2           ticks per time slice
00096 00000080 SysPri     set    128         initial system priority
00097 00000000 MinPty     set    0           initial sys min executable priority
00098 00000000 MaxAge     set    0           initial sys max natural age limit
00099 00000000 MaxMem     set    0           top of RAM (unused)
00100 00000000 Events     set    0           initial event table size (div by 8)
00101 00000000 Compat     set    0           version smoothing byte
00102 00000400 StackSz    set    1024        IRQ Stack Size in bytes (must be 1k
                                             <= StackSz < 256k)
00103 00000000 ColdRetrys set    0           number of retries for coldstart's
                                            "chd" before failing
00104
00105 * Compat flag bit definitions
00106 00000001 SlowIRQ    equ    1           save all regs during IRQ processing
00107 00000002 NoStop     equ    1<<1        don't use 'stop' instruction
00108 00000004 NoGhost    equ    1<<2        don't retain Sticky memory modules
00109 00000008 NoBurst    equ    1<<3        don't enable 68030 cache burst mode
00110 00000010 ZapMem     equ    1<<4        wipe out mem that is allocated/freed
00111 00000020 NoClock    equ    1<<5        don't start sys clock during coldstart
00112
00113 * Compat2 flag bit definitions
00114 00000001 ExtC_I     equ    1<<0        ext instruction cache is coherent
00115 00000002 ExtC_D     equ    1<<1        external data cache is coherent
00116 00000004 OnC_I      equ    1<<2        on-chip inst cache is coherent
00117 00000008 OnC_D      equ    1<<3        on-chip data cache is coherent
00118 00000080 DDIO       equ    1<<7        don't disable data caching when in I/O
00119
00120                use    defsfile (any above defs may be overridden in
                            defsfile)
00001
00002                use    ../DEFS/oskdefs.d
```

```
00001                   opt     -l
00003                   use     ./systype.d
00001 *
00002 * System Definitions for MVME147 System
00003 *
00004 * VERSION FOR DELTA
00005                   opt     -l
00004
00005
00121
00132
00133 * Configuration module body
00134 0000 0000          dc.l   MaxMem        (unused)
00135 0004 0020          dc.w   PollSz        IRQ polling table
00136 0006 0020          dc.w   DevCnt        device table size
00137 0008 0040          dc.w   Procs         initial process table size
00138 000a 0040          dc.w   Paths         initial path table size
00139 000c 0076          dc.w   SysParam      param string for first executable mod
00140 000e 0070          dc.w   SysStart      first executable module name offset
00141 0010 008b          dc.w   SysDev        system default device name offset
00142 0012 008f          dc.w   ConsolNm      standard I/O pathlist name offset
00143 0014 009b          dc.w   Extens        Customization module name offset
00144 0016 0095          dc.w   ClockNm       clock module name offset
00145 0018 0014          dc.w   Slice         number of ticks per time slice
00146 001a 0000          dc.w   IP_ID         interprocessor identification
00147 001c 0000          dc.l   Site          installation site code
00148 0020 0062          dc.w   MainFram      installation name offset
00149 0022 0001          dc.l   CPUTyp        specific 68000 family proc in use
00150 0026 0102          dc.b   Level,Vers,Revis,Edit OS-9 Level
00151 002a 0054          dc.w   OS9Rev        OS-9 revision string offset
00152 002c 0080          dc.w   SysPri        initial system priority
00153 002e 0000          dc.w   MinPty        initial sys min executable priority
00154 0030 0000          dc.w   MaxAge        maximum system natural age limit
00155 0032 0000          dc.l   MDirSz        module directory size (unused)
00156 0036 0000          dc.w   Events        initial event table size (no.r of
                                entries)
00157 0038   10          dc.b   Compat        version change smooth byte
00158 0039   83          dc.b   Compat2       version change smooth byte #2
00159 003a 00b6          dc.w   MemList       memory definitions
00160 003c 0400          dc.w   StackSz/4     IRQ stack size (in longwords)
00161 003e 0000          dc.w   ColdRetrys    coldstart's "chd" retry count
00162 0040 0000          dc.w   0,0,0,0,0     reserved
00163 004a 0000          dc.w   0,0,0,0,0     reserved
00164
00165 * Configuration name strings
00166 0054 4f53 OS9Rev     dc.b     "OS-9/68K V",Vers+'0',".",Revis+'0',0
00167
00168 * The remaining names are defined in the "systype.d" macro
00169                   CONFIG
00170 0062 4465+MainFram   dc.b     "Delta MVME147",0
00172 0070 7368+SysStart   dc.b     "shell",0      name of initial module to execute
00173 0076=7461+SysParam   dc.b     "tapestart; ex sysgo",C$CR,0
Init: OS-9 Configuration Module -
```

```
00174 008b 2f64+SysDev     dc.b     "/dd",0        initial system disk pathlist
00184 008f 2f74+ConsolNm   dc.b     "/term",0      console terminal pathlist
00185 0095 746b+ClockNm    dc.b     "tk147",0      clock module name
00186 009b 4f53+Extens     dc.b     "OS9P2 ssm syscache" include mmu, caching.
00188 00b5   00+           dc.b     0
00189  000000b6+           align
00190          +MemList
00191                  MemType  SYSRAM,250,B_USER,ProbeSize,CPUBeg,BootMemEnd,OnBoard,CPUBeg+TRANS
00192 00b6=0000+           dc.w     SYSRAM,250,B_USER,ProbeSize>>4 type, priority,
                        access, search block size
00193 00be 0000+           dc.l     CPUBeg,BootMemEnd low, high limits (where it
                        appears on local address bus)
00194 00c6 00fa+           dc.w     OnBoard,0      offset to description string
                                 (zero if none), reserved
00195 00ca 0000+           dc.l     CPUBeg+TRANS,0,0 address translation adjustment
                                  (for DMA, etc.), reserved
00199  MemType  SYSRAM,240,B_USER+B_PARITY,ProbeSize,BootMemEnd,UserMemEnd,OffBoard,0
00200 00d6=0000+           dc.w     SYSRAM,240,B_USER+B_PARITY,ProbeSize>>4 type,
                        priority, access, search block size
00201 00de 0040+           dc.l     BootMemEnd,UserMemEnd low, high limits (where it
                        appears on local address bus)
00202 00e6 0107+           dc.w     OffBoard,0     offset to description string
                                 (zero if none), reserved
00203 00ea 0000+           dc.l     0,0,0          address translation adjustment
                                 (for DMA, etc.), reserved
00207 00f6 0000+           dc.l     0              terminate list
00208 00fa 6f6e+OnBoard    dc.b     "on-board ram",0
00209 0107 766d+OffBoard   dc.b     "vme bus ram",0
00210
00214
00218
00219 * define default caching modes (CPUTyp and system specific)
00220 * NOTE:  the following rules should be applied in determining
00221 *        the "coherency" of a cache and setting up the Compat2
00222 *        cache function flags:
00223 *
00224 *         - if the cache does not exits, then it is always coherent.
00225 *         - the on-chip cache coherency is not changable, except
00226 *           for the 68040.  If a 68040 system is used with
00227 *           bus-snooping disabled, then that fact should be registered
00228 *           by the user defining the label NoSnoop040 in their local
00229 *           "systype.d" file.
00230 *         - the coherency of external caches is indicated by the
00231 *           SnoopExt definition.  If the external caches are
00232 *           coherent or non-existant, then the label SnoopExt
00233 *           should be defined in "systype.d".
00234 *         - the kernel will disable data caching when calling a file
00235 *           manager, unless the "NoDataDis" label is defined.
00236 *           Disabling data caching is required for systems that have
00237 *           drivers that use dma and don't perform any explicit data
00238 *           cache flushing.  If your system does NOT use dma drivers,
00239 *           or the drivers care for the cache, then the NoDataDis
00240 *           label should be defined in "systype.d".
00241 *
```

```
00243
00246 * external caches are coherent or absent
00247  00000003 ExtCache   equ      ExtC_I!ExtC_D
00252
00261  00000003 Compat2    set      ExtCache       68030 on-chip caches are NOT snoopy
00270
00271 * add "don't disable data cache when in I/O" to Compat2
00273  00000083 Compat2    set      Compat2!DDIO
00275
00277
00278  00000114           ends
Errors: 00000
Memory used: 45k
Elapsed time: 6 second(s)
```

## *Sysgo Module*

```
Microware OS-9/68000 Resident Macro Assembler V1.6  86/11/04   Page 1  sysgo.a
Sysgo - OS-9/68000 Initial (startup) module
00001                    nam      Sysgo
00002                    ttl      OS-9/68000     Initial (startup) module
00003
00015  00000004 Edition    equ      4            current edition number
00016
00017  00000101 Typ_Lang   set      (Prgrm<<8)+Objct
00018  00000000 Attr_Rev   set      0            (non-re- entrant)
00019                    psect    sysgo,Typ_Lang,Attr_Rev,Edition,128,Entry
00020
00021                    use      defsfile
00022
00023                    vsect
00024  00000000          ds.b     255          stack space
00025  00000000          ends
00026
00027 0000=4e40 Intercpt   os9      F$RTE         return from intercept
00028
00029 0004 41fa Entry      lea      Intercpt(pc),a0
00030 0008=4e40           os9      F$Icpt
00031 000c 41fa           lea      CmdStr(pc),a0  default execution dir ptr
00032 0010 7004           moveq    #Exec_,d0      execution mode
00033 0012=4e40           os9      I$ChgDir       chg exec dir (ignore errs)
00034 0016 640c           bcc.s    Entry10        continue if no error
00035 0018 7001           moveq    #1,d0          std output path
00036 001a 721a           moveq    #ChdErrSz,d1   size
00037 001c 41fa           lea      ChdErrMs(pc),a0 "Help, I can't find CMDS"
00038 0020=4e40           os9      I$WritLn       output error message
00039
00040 * Process startup file
00041 0024 7000 Entry10    moveq    #0,d0          std input path
00042 0026=4e40           os9      I$Dup          clone it
00043 002a 3e00           move.w   d0,d7          save cloned path number
00044 002c 7000           moveq    #0,d0          std input path
00045 002e=4e40           os9      I$Close
00046 0032 303c           move.w   #Read_,d0
00047 0036 41fa           lea      Startup(pcr),a0 "startup" pathlist
00048 003a=4e40           os9      I$Open         open startup file
00049 003e 640e           bcc.s    Entry15        continue if no error
00050 0040 7001           moveq    #1,d0          std output path
00051 0042 7220           moveq    #StarErSz,d1   size of startup error msg
00052 0044 41fa           lea      StarErMs(pc),a0 "Can't find 'startup'"
00053 0048=4e40           os9      I$WritLn       output error message
00054 004c 6032           bra.s    Entry25
00055
00056 004e 7000 Entry15    moveq    #0,d0          any type module
00057 0050 7200           moveq    #0,d1          no add'l default mem size
00058 0052 7406           moveq    #StartPSz,d2   sz of startup shell params
00059 0054 7603           moveq    #3,d3          copy three std I/O paths
00060 0056 7800           moveq    #0,d4          same priority
00061 0058 41fa           lea      ShellStr(pcr),a0 shell name
```

```
00062 005c 43fa          lea     StartPrm(pcr),a1 initial parameters
00063 0060=4e40          os9     F$Fork        fork shell
00064 0064 6410          bcc.s   Entry20       continue if no error
00065 0066 7001          moveq   #1,d0         std output path
00066 0068 7219          moveq   #FrkErrSz,d1  size
00067 006a 41fa          lea     FrkErrMs(pc),a0 "oh no, can't fork Shell"
00068 006e=4e40          os9     I$WritLn      output error message
00069 0072=4e40          os9     F$SysDbg      crash system
00070
00071 0076=4e40 Entry20  os9     F$Wait        wait for death,ignore error
00072 007a 7000          moveq   #0,d0         std input path
00073 007c=4e40          os9     I$Close       close redirected "startup"
00074 0080 3007 Entry25  move.w  d7,d0
00075 0082=4e40          os9     I$Dup         restore original std input
00076 0086 3007          move.w  d7,d0
00077 0088=4e40          os9     I$Close       remove cloned path
00078
00079 008c 7000 Loop     moveq   #0,d0         any type module
00080 008e 7200          moveq   #0,d1         default memory size
00081 0090 7401          moveq   #1,d2         one parameter byte (CR)
00082 0092 7603          moveq   #3,d3         copy std I/O paths
00083 0094 7800          moveq   #0,d4         same priority
00084 0096 41fa          lea     ShellStr(pcr),a0 shell name
00085 009a 43fa          lea     CRChar(pcr),a1 null paramter string
00086 009e=4e40          os9     F$Fork        fork shell
00087 00a2 650a          bcs.s   ForkErr       abort if error
00088 00a4=4e40          os9     F$Wait        wait for it to die
00089 00a8 6504          bcs.s   ForkErr
00090 00aa 4a41          tst.w   d1            zero status?
00091 00ac 67de          beq.s   Loop          loop if so
00092 00ae=4e40 ForkErr  os9     F$PErr        print error message
00093 00b2 60d8          bra.s   Loop
00094
00095 00b4 7368 ShellStr dc.b    "shell",0
00096 00ba=5379 FrkErrMs dc.b    "Sysgo can't fork 'shell'",C$CR
00097  00000019 FrkErrSz equ     *-FrkErrMs
00098
00099 00d3 434d CmdStr   dc.b    "CMDS",0
00100 00d8=5379 ChdErrMs dc.b    "Sysgo can't chx to 'CMDS'",C$CR
00101  0000001a ChdErrSz equ     *-ChdErrMs
00102
00103 00f2 7374 Startup  dc.b    "startup",0
00104 00fa=5379 StarErMs dc.b    "Sysgo can't open 'startup' file",C$CR
00105  00000020 StarErSz equ     *-StarErMs
00106
00107 011a 2d6e StartPrm dc.b    "-npxt"
00108 011f=  00 CRChar   dc.b    C$CR
00109  00000006 StartPSz equ     *-StartPrm
00110  00000120          ends
00111
Errors: 00000
Memory used: 31k
Elapsed time: 21 second(s)
```

## *Signals: Example Program*

The following program demonstrates a subroutine that reads a \n terminated string from a terminal with a ten second timeout between the characters. This program is designed to illustrate signal usage; it does not contain any error checking.

The _ss_ssig(path, value) library call notifies that operating system to send the calling process a signal with signal code value when data is available on path. If data is already pending, a signal is sent immediately. Otherwise, control returns to the calling program and the signal is sent when data arrives.

```
#include <stdio.h>
#include <errno.h>

#define TRUE 1
#define FALSE 0

#define GOT_CHAR 2001
short dataready;    /* flag to show that signal was received */

/* sighand - signal handling routine for this process */
sighand(signal)
register int signal;
{
    switch(signal) {
        /* ^E or ^C? */
        case 2:
        case 3:
            _errmsg(0,"termination signal received\n");
            exit(signal);
        /* Signal we're looking for? */
        case GOT_CHAR:
            dataready = TRUE;
            break;
        /* Anything else? */
        default:
            _errmsg(0,"unknown signal received ==> %d\n",signal);
            exit(1);
    }
}

main()
{
    char buffer[256];          /* buffer for typed-in string */

    intercept(sighand);        /* set up signal handler */

    printf("Enter a string:\n"); /* prompt user */

    /* call timed_read, returns TRUE if no timeout, -1 if timeout */
    if (timed_read(buffer) == TRUE)
        printf("Entered string = %s\n",buffer);
    else
```

```
        printf("\nType faster next time!\n");
}

int timed_read(buffer)
register char *buffer;
{
    char c = '\0';          /* 1 character buffer for read */
    short timeout = FALSE;    /* flag to note timeout occurred on read */
    int pos = 0;            /* position holder in buffer */

    /* loop until <return> entered or timeout occurs */
    while ( (c != '\n')  &&  (timeout == FALSE) ) {
        sigmask(1);          /* mask signals for signal setup */
        _ss_ssig(0,GOT_CHAR);  /* set up to have signal sent */
        sleep(10);           /* sleep for 10 seconds or until signal */

/* NOTE: we had to mask signals before doing _ss_ssig() so we did not get the
signal between the time we _ss_ssig()'ed and went to sleep.  */

        /* Now we're awake, determine what happened */
        if (!dataready)
            timeout = TRUE;
        else {
            read(0,&c,1);       /* read the ready byte */
            buffer[pos] = c;     /* put it in the buffer */
            pos++;              /* move our position holder */
            dataready = FALSE;   /* mark data as read */
        }
    }
    /* loop has terminated, figure out why */
    if (timeout)
        return -1;          /* there was a timeout so return -1 */
    else {
        buffer[pos] = '\0';   /* null terminate the string */
        return TRUE;
    }
}

#asm
* C binding for sigmask(value)
sigmask: move.l d1,-(sp)       save d1 on the stack
 move.l d0,d1                  get the passed parameter in the right place
 clr.l d0                    make d0 = 0
 os9 F$SigMask                 make the system call to mask signals
 bcc.s ret                   if no error...
 move.l #-1,d0                 return -1 to user
 move.l d1,errno(a6)           fill errno with error number
ret move.l (sp)+,d1            restore d1 from the stack
 rts                       return to user
#endasm
```

## *Alarms: Example Program*

Compile the following example program with this command:

```
$ cc deton.c
```

The complete source code for the example program is as follows:

```
/*---------------------------------------------------------------------------|
|       Psect Name:deton.c                                                   |
|       Function: demonstrate alarm to time out user input                   |
|---------------------------------------------------------------------------*/
@_sysedit: equ 1

#include <stdio.h>
#include <errno.h>

#define TIME(secs) ((secs << 8) | 0x80000000)
#define PASSWORD "Ripley"

/*---------------------------------------------------------------------------*/
sighand(sigcode)
{
        /* just ignore the signal */
}

/*---------------------------------------------------------------------------*/
main(argc,argv)
int     argc;
char    **argv;
{
   register int    secs = 0;
   register int    alarm_id;
   register char   *p;
   register char   name[80];

   intercept(sighand);
   while (--argc)
      if (*(p = *(++argv)) == '-') {
         if (*(++p) == '?')
            printuse();
         else exit(_errmsg(1, "error: unknown option - '%c'\n", *p));
      } else if (secs == 0)
            secs = atoi(p);
         else exit(_errmsg(1, "unknown arg - \"%s\"\n", p));

   secs = secs ? secs : 3;
   printf("You have %d seconds to terminate self-destruct...\n", secs);


   /* set alarm to time out user input */
   if ((alarm_id = alm_set(2, TIME(secs))) == -1)
      exit(_errmsg(errno, "can't set alarm - "));
```

```
   if (gets(name) != 0)
      alm_delete(alarm_id);     /* remove the alarm; it didn't expire */
   else printf("\n");

   if (_cmpnam(name, PASSWORD, 6) == 0)
      printf("Have a nice day, %s.\n", PASSWORD);
   else printf("ka BOOM\n");

   exit(0);
}

/*---------------------------------------------------------------------------*/
/* printuse() - print help text to standard error                            */
printuse()
{
   fprintf(stderr, "syntax: %s [seconds]\n", _prgname());
   fprintf(stderr, "function: demonstrate use of alarm to time out I/O\n");
   fprintf(stderr, "options: none\n");
   exit(0);
}
```

## *Events: Example Program*

The following program uses a binary semaphore to illustrate the use of events. To execute this example:

- Type the code into a file called sema1.c.
- Copy sema1.c to sema2.c.
- Compile both programs.
- Run both programs with this command: sema1 & sema2

The program creates an event with an initial value of 1 (free), a wait increment of -1, and a signal increment of 1. Then, the program enters a loop which waits on the event, prints a message, sleeps, and signals the event. After ten times through the loop, the program unlinks itself from the event and deletes the event from the system.

```
#include <stdio.h>
#include <events.h>
#include <errno.h>

char *ev_name = "semaevent";     /* name of event to be used */
int ev_id;     /* id that will be used to access event */

main()
{
    int count = 0;  /* loop counter */

    /* create or link to the event */
    if ((ev_id = _ev_link(ev_name)) == -1)
        if ((ev_id = _ev_creat(1,-1,1,ev_name)) == -1)
            exit(_errmsg(errno,"error getting access to event - "));

    while (count++ < 10) {
        /* wait on the event */
        if (_ev_wait(ev_id, 1, 1) == -1)
            exit(_errmsg(errno,"error waiting on the event - "));

        _errmsg(0,"entering \"critical section\"\n");

        /* simulate doing something useful */
        sleep(2);

        _errmsg(0,"exiting \"critical section\"\n");

        /* signal event (leaving critical section) */
       if (_ev_signal(ev_id, 0) == -1)
            exit(_errmsg(errno,"error signalling the event - "));

        /* simulate doing something other than critical section */
        sleep(1);
    }
```

```
    /* unlink from event */
    if (_ev_unlink(ev_id) == -1)
        exit(_errmsg(errno,"error unlinking from event - "));

    /* delete event from system if this was the last process to unlink from it */
    if (_ev_delete(ev_name) == -1 && errno != E_EVBUSY)
        exit(_errmsg(errno,"error deleting event from system - "));

    _errmsg(0,"terminating normally\n");
}
```

## C Trap Handler

Use the following makefile to make the example C trap handler and test programs:

```
# makefile - Used to make the example C trap handler and test program.

CFLAGS = -sqgixt=/dd
RDIR   = RELS
TRAP   = ctrap
TEST   = traptst

# Dependencies for making the entire example.

ctrap.example: $(TRAP) $(TEST)
   touch ctrap.example

# Dependencies for making the ctrap trap handler.

$(TRAP): tstart.r $(TRAP).r
   chd $(RDIR);\
   l68 tstart.r $(TRAP).r -l=/dd/lib/cio.l -l=/dd/lib/clib.l -l=/dd/lib/sys.l\
           -o=$(TRAP) -g

# Dependencies for making the traptst test program.

$(TEST): $(TEST).r

$(TEST).r: $(TEST).c
   cc -gim=2k $(TEST).c -r=$(RDIR)
```

The complete source for the C trap handler startup routines (tstart.a) is as follows:

```
*************************************************
*
* tstart.a - C trap handler startup routines.
*
         nam     tstart C trap handler interface
         use     /dd/defs/oskdefs.d

*SYSTRAP    equ     1       define if trap should execute in system state

MaxParams   equ     20      maximum number of "C" style parameters allowed

      ifdef   SYSTRAP
AttrRevs    set     (ReEnt+SupStat)<<8  (system state)
      else
AttrRevs    set     (ReEnt)<<8          (user state)
      endc

TypeLang    set     (TrapLib<<8)+Objct
         psect   traphand,TypeLang,AttrRevs,0,0,TrapEnt
         dc.l    TrapInit
         dc.l    TrapTerm
```

```
**********************************
* Subroutine TrapInit
*   Trap handler initialization entry point
*
* Passed: d0.w  = User Trap number (1-15)
*         d1.l  = (optional) additional static storage
*         d2-d7 = caller's registers at time of trap
*         (a0)  = trap handler module name pointer
*         (a1)  = trap handler execution entry point
*         (a2)  = trap module pointer
*         a3-a5 = caller's registers at time of trap
*         (a6)  = trap handler static storage pointer
*         (a7)  = trap init stack frame pointer
*
* Returns: d0.l = "C" trapinit return value
*          (a0) = updated trap handler name pointer
*          (a1) = trap handler execution entry point
*          (a2) = trap module pointer
*          cc   = carry set, d1.w = error code if error
*          Other values returned are dependent on the trap handler
*
* The user stack looks like this:
*          .-------------------------.
*        +8 |    caller's return PC   |
*          |------------+------------|
*        +4 |    0000    |    0000    |
*          |------------|------------|
*          |   caller's a6 register  |
*   (usp)-> -------------------------
*
* NOTE: In system state, (a7)=system stack pointer.  This has a reasonable
*       amount of stack space (~1K).  No assumptions about where it is
*       should be made.

TrapInit:   bra     TrapEnt     call "C" trap handler (with func. code zero)

**********************************
* Subroutine TrapEnt
*   User Trap entry point
*
* Passed: d0-d7 = caller's registers
*         a0-a5 = caller's registers
*         (a6)  = trap handler static storage pointer
*         (a7)  = trap entry stack frame pointer
*         usp   = undisturbed user stack (in system state)
*
* Returns: cc   = carry set, d1.w=error code if error
*          Other values returned are dependent on the trap handler
*
* The system stack looks like this:
*          .-------------------------.
*        +8 |   caller's return PC    |
*          |-------------+-----------
```

```
*       +6 |   vector #  |
*          |-------------|
*       +4 |  func code  |
*          |-------------+------------.
*          |  caller's (a6) register  |
*   (a7)->  -------------------------

         org     0          stack offset definitions
S_CParams   do.l    MaxParams
S_a0        do.l    1          caller's a0 reg
S_a1        do.l    1          caller's a1 reg
S_a6        do.l    1          caller's a6 reg
S_func      do.w    1            trap function code
S_vect      do.w    1            user trap exception offset
S_cleanup   equ     .
S_pc        do.l    1          return pc

TrapEnt:    movem.l a0-a1,-(a7)        save regs
         lea     -MaxParams*4(a7),a7 allocate parameter space
         lea     S_CParams(a7),a1    ptr to C parameter area

      ifdef   SYSTRAP
         move    usp,a0      caller's parameters are on user stack ptr
         adda.l  #12,a0      above two rts pc's
      else
         lea     S_pc+16(a7),a0  caller's remaining C parameters ptr
      endc

         moveq   #MaxParams-1,d1 number of (potential) parameters
Trap10      move.l  (a0)+,(a1)+     copy caller's params from user stack
         dbra    d1,Trap10
         moveq   #0,d0           sweep reg
         move.w  S_func(a7),d0   1st param = func
         move.l  S_a6(a7),d1     2nd param = caller's (a6)
         bsr     ctrap           execute C traphandler
Trap90      movea.l S_a6(a7),a6     restore caller's a6
         lea     S_cleanup(a7),a7    discard scratch
         rts                     return to user program

*********************************
* Subroutine TrapTerm
*   Terminate trap handler servicing.
*
* As of this release (OS-9 V2.3) the trap termination entry point
* is never called by the OS-9 kernel.  Documentation details will
* be available when a working implementation exists.

TrapTerm:   move.w  #1<<8+199,d1    never called; so if it gets here...
         OS9     F$Exit          crash program (Error 001:199)

         ends
```

The complete source for the example C trap handler library (ctrap.c) is as follows:

```
/*****************************************
 *
 * ctrap.c - Example C trap handler library.
 *
 * ctrap(func, a6, p1, p2, ...)
 */

int ctrap(func, a6, p1, p2, p3, p4)
register int func;              /* trap function code */
char *a6;                       /* caller's static storage base */
unsigned int p1, p2, p3, p4;    /* caller's parameters */
{
   register int result;

   switch(func)
   {
      case  0 :   result = 0;        break;  /* tlink call */
      case '+':   result = p1 + p2;   break;
      case '-':   result = p1 - p2;   break;
      case '*':   result = p1 * p2;   break;
      case '/':   result = p1 / p2;   break;
      case '&':   result = p1 & p2;   break;
      case '|':   result = p1 | p2;   break;
      case '^':   result = p1 ^ p2;   break;
      case '>':   result = p1 >> p2;  break;
      case '<':   result = p1 << p2;  break;
      default :   result = -1;        break;
   }
   return (result);
}
```

The complete source for traptst.c, which calls the ctrap handler, is as follows:

```
/*********************************************
 *
 * traptst.c - Calls the "ctrap" trap handler.
 *
 */

main()
{
   int   i, n;
   int   x = 22;
   int   y = 5;
   int   trapnum = 6;
   char *operator = "+-*/&|^<>?";

   printf("tlink: %d\n", tlink(trapnum, "ctrap"));

   n = strlen (operator);
   for (i = 0; i < n; ++i)
      printf("tcall(%d %c %d) = %d\n", x, operator[i], y,
         tcall(trapnum, operator[i], x, y));
}


/* bindings for tlink, tcall */
/*********************************************************************/
/* tlink(trapnum, trapname) - link to trap handler              */
/* int trapnum;           user trap number (1-15)             */
/* char *trapname;        name of trap module (NULL to unlink)  */

#asm
tlink:    link   a5,#0
          movem.l a0-a2,-(a7)    save regs
          movea.l d1,a0          copy ptr to trap handler name
          moveq   #0,d1          no memory override
          OS9     F$TLink        link to trap handler
          bcc.s   tlink99        exit if no error
          move.l  d1,errno(a6)   save error number for caller
          moveq   #-1,d0         return error status
tlink99   movem.l (a7)+,a0-a2    restore regs
          unlk    a5
          rts
#endasm

/*********************************************************************/
/* tcall(trapnum, func, param1, param2, ...) - call trap handler   */
/* int trapnum;           user trap number (1-15)             */
/* short func;            trap function number                */
/* other parameters may be ints or pointers                   */

#asm
TRAP      equ    $4e40      user trap(0) opcode
RTS       equ    $4e75      rts opcode
```

```
        vsect
trapinst    ds.w    2
rtsinst     ds.w    1
        ends

tcall:      link    a5,#0
        tst.l   d0              valid trap number?
        beq.s   paramerr        abort if not
        cmp.l   #15,d0          valid trap number?
        bhi.s   paramerr
        add.w   #TRAP,d0
        movem.w d0-d1,trapinst(a6)  build usr trap instruction
        move.w  #RTS,rtsinst(a6)    set rts instruction
        moveq.l #0,d0           flush instruction cache
        os9     F$CCtl          ignore error
        jsr     trapinst(a6)    execute trap call
        bcc.s   tcall99         exit if no error
        move.l  d1,errno(a6)    save error number
        bra.s   tcallerr        abort

paramerr    move.l  #E$Param,errno(a6)
tcallerr    moveq   #-1,d0
tcall99
        unlk    a5
        rts
#endasm
```

## RBF Device Descriptor

```
Microware OS-9/68020 Resident Macro Assembler V2.9  90/12/07  15:29  Page     1
 ../io/d0.a
D0 Device Descriptor - Device Descriptor for Floppy disk controller
00001                    nam     D0 Device Descriptor
00002                    use     defsfile
00001
00002                    use     ../DEFS/oskdefs.d
00001                    opt     -l
00003                    use     ./systype.d
00001 * System Definitions for MVME147 System
00002 *
00003                    opt     -l
00004
00005
00003                    use     ../io/rbfdesc.a
00001
00002                    ttl     Device Descriptor for Floppy disk controller
00003
00045  0000000e Edition   equ     14          current edition number
00046
00047 * PD_DNS values
00048  00000000 Single    equ     0           FM encoded media
00049  00000001 Double    equ     1           MFM encoded media/double-track
                                                                    density
00050  00000002 Quad      equ     1<<1        Quad track density
00051  00000004 Octal     equ     1<<2        Octal track density
00052
00053 * PD_TYP values
00054 * Note: For pre-V2.4 Five/Eight defines the disk size, rotational
00055 *       speed and data transfer rate.  From V2.4 the physical size
00056 *       is defined in bits 4 - 1, and PD_Rate defines the rotational
00057 *       speed and data transfer rate.
00058
00059 * floppy disk definitions
00060  00000000 Five      equ     0<<0        drive is 5 1/4"
00061  00000001 Eight     equ     1<<0        drive is 8"
00062  00000000 SizeOld   equ             0<<1        size/speed defined by
                                                                    bit 0 value (pre-V2.4)
00063  00000002 Size8     equ     1<<1        physical size is 8"
00064  00000004 Size5     equ     2<<1        physical size is 5 1/4"
00065  00000006 Size3     equ     3<<1        physical size is 3 1/2"
00066
00067 * hard disk definitions
00068  00000040 HRemov    equ     1<<6        hard disk is removable
00069  00000080 Hard      equ     1<<7        hard disk media
00070
00071 * PD_Rate values
00072 * Note: V2.4 drivers should derive the disk data transfer rate and
00073 *       rotational speed from this field if PD_TYP, bits 4 - 1 are
00074 *       non-zero.  If not, then PD_TYP, bit 0 infers these.
00075  00000000 rpm300    equ     0           rotational speed is 300 rpm
00076  00000001 rpm360    equ     1           rotational speed is 360 rpm
```

```
00077  00000002 rpm600    equ     2          rotational speed is 600 rpm
00078  00000000 xfr125K   equ     0<<4         transfer rate is 125K bits/sec
00079  00000010 xfr250K   equ     1<<4         transfer rate is 250K bits/sec

00080  00000020 xrf300K   equ     2<<4         transfer rate is 300K bits/sec
00081  00000030 xfr500K   equ     3<<4         transfer rate is 500K bits/sec
00082  00000040 xfr1M     equ     4<<4        transfer rate is 1M bits/sec
00083  00000050 xfr2M     equ     5<<4        transfer rate is 2M bits/sec
00084  00000060 xfr5M     equ     6<<4        transfer rate is 5M bits/sec
00085
00086 * PD_VFY values
00087  00000001 ON        equ     1          "no-verify" ON
00088  00000000 OFF       equ     0           "no-verify" OFF
                                                                    (i.e. verify is ON!)
00089
00090 * macro parameter #6 definitions (drive type)
00091
00092  00000001 d877      equ     1          single density 8"
00093  00000004 dd877     equ     4           double density 8"
00094  00000002 d540      equ     2          single density 5 1/4" 40 trk
00095  00000005 dd540     equ     5           double density 5 1/4" 40 trk
00096  00000003 d580      equ     3          single density 5 1/4" 80 trk
00097  00000006 dd580     equ     6           double density 5 1/4" 80 trk
00098  00000007 ramdisk   equ     7            volatile ram disk
00099  00000008 nvramdisk equ     8              non-volatile ram disk
00100  00000009 uv580     equ     9          universal 5 1/4" 80 track
00101  0000000a autosize  equ     10          autosize device (SS_DSize tells media size)
00102  0000000b dd380     equ     11          double density 3 1/2", 80 trk
00103  0000000c uv380     equ     12         universal 3 1/2" 80 track
00104  0000000d hd580     equ     13          double density 5 1/4"
                                                                    80 track ’8" image’
00105  0000000e ed380     equ     14         double density 3 1/2"
                                                                    80 track, 4M byte unformatted
00106  0000000f hd577     equ     15         double density 5 1/4"
                                                                    77 track ’8" image’
00107  00000010 uv577     equ     16         universal 5 1/4" ’8" image’
00108  00000011 uv877     equ     17         universal 8"
00109
00110  00000003 Density   set     BitDns+(TrkDns<<1)
00111  00000024 DiskType  set     DiskKind+(DnsTrk0<<5)
00112
00113  00000f00 TypeLang  set     (Devic<<8)+0
00114  00008000 Attr_Rev  set     (ReEnt<<8)+0
00115
00116                psect   RBFDesc,TypeLang,Attr_Rev,Edition,0,0
00117
00118 0000 fffe          dc.l    Port          port address
00119 0004   45          dc.b    Vector        auto-vector trap assignment
00120 0005   04          dc.b    IRQLevel      IRQ hardware interrupt level
00121 0006   05          dc.b    Priority      irq polling priority
00122 0007   a7          dc.b    Mode          device mode capabilities
00123 0008 0048          dc.w    FileMgr       file manager name offset
00124 000a 004c          dc.w    DevDrv        device driver name offset
00125 000c 0053          dc.w    DevCon        (reserved)
```

```
00126 000e 0000          dc.w    0,0,0,0       reserved
00127 0016 0030          dc.w    OptLen
00128
00129 * Default Parameters
00130          OptTbl
00131 0018=  00          dc.b    DT_RBF        device type
00132 0019   02          dc.b    DrvNum        drive number
00133 001a   03          dc.b    StepRate      step rate
00134 001b   24          dc.b    DiskType      type of disk 8"/5 1/4"/Hard/etc
00135 001c   03          dc.b    Density       Bit Density and track density
00136 001d   00          dc.b    0             reserved
00137 001e 004f          dc.w    Cylnders-TrkOffs number of logical cylinders
00138 0020   02          dc.b    Heads         Number of Sides (Floppy)
                                                          Heads(Hard Disk)
00139 0021   00          dc.b    NoVerify      OFF = disk verify ON = no verify
00140 0022 0010          dc.w    SectTrk       default sectors/track
00141 0024 0010          dc.w    SectTrk0      default sectors/track track 0
00142 0026 0008          dc.w    SegAlloc      segment allocation size
00143 0028   04          dc.b    Intrleav      sector interleave factor
00144 0029   00          dc.b    DMAMode       DMA mode (driver dependant)
00145 002a   01          dc.b    TrkOffs       track base offset (first
                                                          accessable track)
00146 002b   01          dc.b    SectOffs      sector base offset
                                                          (starting physical sector number)
00147 002c 0100          dc.w    SectSize      # of bytes/sector
00148 002e 0002          dc.w    Control       control byte
00149 0030   07          dc.b    Trys          number of retrys
                                                          0 = no retrys/error correction
00150 0031   02          dc.b    ScsiLun       scsi logical unit number
00151 0032 0000          dc.w    WrtPrecomp    write precomp cylinder
00152 0034 0000          dc.w    RedWrtCrnt    reduce write current cylinder
00153 0036 0000          dc.w    ParkCyl       cylinder to park head
                                                          for hard disk
00154 0038 0000          dc.l    LSNOffset     logical sector offset
00155 003c 0050          dc.w    TotalCyls     total cylinders on drive
00156 003e   06          dc.b    CtrlrID       scsi controller id
00157 003f   10          dc.b    Rates         data-transfer rate &
                                                          rotational speed
00158 0040 0000          dc.l    ScsiOpts      scsi option flags
00159 0044 0000          dc.l    MaxCount-1    maximum byte count
                                                          passable to driver
00160  00000030 OptLen    equ     *-OptTbl
00161
00162 0048 5242 FileMgr   dc.b    "RBF",0       Random block file manager

00274          DiskKind  set     Size5   five inch disk
00275          Cylnders  set     80      number of (physical) tracks
00276          BitDns    set     Double MFM recording
00277          Rates     set     xfr250K+rpm300
00278          DnsTrk0   set     Double MFM track 0
00279          TrkDns    set     Double 96tpi
00280          SectTrk   set     16      sectors/track (except trk 0, side 0)
00281          SectTrk0  set     16      sectors/track, track 0, side 0
00282          SectOffs  set     1       physical sector start = 1
```

```
00283         TrkOffs   set     1     track 0 not used
00284         TotalCyls  set     Cylnders number of actual cylinders on disk
00384
00385 ****************
00386 * Descriptor Defaults
00387  000000a7 Mode      set     Dir_+ISize_+Exec_+Updat_
00388  00000000 BitDns    set     Single
00389  00000002 Heads     set     2
00390  00000002 StepRate  set     2
00391  00000003 Intrleav  set     3
00392  00000000 NoVerify  set     OFF
00393  00000000 DnsTrk0   set     Single
00394  00000000 DMAMode   set     0          non dma device
00395  00000008 SegAlloc  set     8          minimum segment allocation size
00396  00000000 TrkOffs   set     0
00397  00000000 SectOffs  set     0
00398  00000100 SectSize  set     256        default sector size 256 bytes.
00399  00000000 WrtPrecomp set     0          no write precomp
00400  00000000 RedWrtCrnt set     0          no reduced write current
00401  00000000 ParkCyl   set     0          where to park the head for
                                                                  hard disk
00402  00000000 ScsiLun   set     0          scsi logical unit number
00403  00000000 CtrlrID   set     0          controller id
00404  00000000 LSNOffset  set     0          logical sector offset for scsi
                                                                  hard disks
00405  00000000 TotalCyls  set     0          number of actual cylinders
                                                                  on disk
00406
00407 * scsi options flag definitions
00408
00409  00000001 scsi_atn   set     1<<0       assert ATN supported
00410  00000002 scsi_target set     1<<1       target mode supported
00411  00000004 scsi_synchr set     1<<2       synchronous transfers supported
00412  00000008 scsi_parity set     1<<3       enable SCSI parity
00413
00414  00000000 ScsiOpts   set     0          scsi options flags (default)
00415
00416 * device control word definitions
00417
00418  00000000 FmtEnabl   set     0<<0       enable formatting
00419  00000001 FmtDsabl   set     1<<0       disable formatting
00420  00000000 MultDsabl  set     0<<1       disable multi-sectors
00421  00000002 MultEnabl  set     1<<1       enable multi-sectors
00422  00000000 StabDsabl  set     0<<2       device doesn't have stable id
00423  00000004 StabEnabl  set     1<<2       device has stable id
00424  00000000 AutoDsabl  set     0<<3       device size from device
                                                                  descriptor
00425  00000008 AutoEnabl  set     1<<3       device tells size via SS_DSize
00426  00000000 FTrkDsabl  set     0<<4       device can't format a single track
00427  00000010 FTrkEnabl  set     1<<4       device can format a single track
00428  00000000 Control    set     0          descriptor control word (default)
00429
00430  00000007 Trys       set     7          number of Trys
00431  00010000 MaxCount   set     65536      default maximum transfer count of driver (16-bit)
```

```
00432  00000000 Rates     set     0           default transfer-rate & rotational speed
00433
00434 * end of file
00435
00004  00000000 DrvNum    set     0

00005                DiskD0
00006                RBFDesc  SCSIBase,SCSIVect,SCSILevel,5,rb5400,uv580
00011 004c 7262+DevDrv    dc.b    "rb5400",0    driver module name

00107  00000004+DiskKind  set     Size5        five inch disk
00108  00000050+Cylnders  set     80           number of (physical) tracks
00109  00000001+BitDns    set     Double       MFM recording
00110  00000010+Rates     set     xfr250K+rpm300
00111  00000001+DnsTrk0   set     Double       MFM track 0
00112  00000001+TrkDns    set     Double       96tpi
00113  00000010+SectTrk   set     16           sectors/track
                                                                    (except trk 0, side 0)
00114  00000010+SectTrk0  set     16           sectors/track, track 0, side 0
00115  00000001+SectOffs  set     1            physical sector start = 1
00116  00000001+TrkOffs   set     1            track 0 not used
00117  00000050+TotalCyls set     Cylnders     number of actual cylinders on disk

00119
00213  00000002+DrvNum    set     2            logical device number
00214  00000003+StepRate  set     3            6ms step rate
00215  00000004+Intrleav  set     4
00216  00000001+SectOffs  set     fd_base
00217  0000ff01+MaxCount  set     SectSize*255+1 practical max byte-count to pass
00218  00000002+Control   set     FmtEnabl+MultEnabl
                                                                    format enable,multi-sector i/o
00219  00000002+ScsiLun   set     OMTI_FD_LUN   Logical unit number on controller
00220  00000006+CtrlrID   set     OMTI_TargID   scsi id of controller
00221 0053 7363+DevCon    dc.b    "scsi147",0   low-level driver module
00222  0000005c          ends
```

## *SCF Device Descriptor*

```
Microware OS-9/68000 Resident Macro Assembler V1.6 86/11/04  Page 1 term.a
Term - 68000 Term device descriptor module
00001                    nam      Term
00002                    ttl      68000         Term device desc. module
00003                    use      defsfile
00004
00005
00006
00007
00008                    use      ../io/scfdesc.a
00011
00012  00000004 Edition    equ      4             current edition number
00013
00014  00000f00 TypeLang   set      (Devic<<8)+0
00015  00008000 Attr_Rev   set      (ReEnt<<8)+0
00016                    psect    ScfDesc,TypeLang,Attr_Rev,Edition,0,0
00017
00018 0000 00fe           dc.l     Port          port address
00019 0004   70           dc.b     Vector        auto-vector trap assignment
00020 0005   02           dc.b     IRQLevel      IRQ hardware interrupt lev.
00021 0006   53           dc.b     Priority      irq polling priority
00022 0007   23           dc.b     Mode          Device mode capabilities
00023 0008 0034           dc.w     FileMgr       file manager name offset
00024 000a 0038           dc.w     DevDrv        device driver name offset
00025 000c=0000           dc.w     DevCon        device constant's offset
00026 000e 0000           dc.w     0,0,0,0       reserved
00027 0016 001c           dc.w     OptSiz        option byte count
00028
00029 * Default Parameters
00030          Options
00031 *                                            default
00032 *                    name          function              value
00033 *                    --------      -----------------     -------
00034 0018   00           dc.b     DT_SCF       device type             SCF
00035 0019   00           dc.b     upclock      upcase lock             OFF
00036 001a   01           dc.b     bsb          backspace=BS,SP,BS      ON
00037 001b   00           dc.b     linedel      line del/bsp line       OFF
00038 001c   01           dc.b     autoecho     full duplex             ON
00039 001d   01           dc.b     autolf       auto line feed          ON
00040 001e   00           dc.b     eolnulls     null count              0
00041 001f   00           dc.b     pagpause     end of page pause       OFF
00042 0020   18           dc.b     pagsize      lines per page          24
00043 0021   08           dc.b     C$Bsp        backspace char          ^H
00044 0022   18           dc.b     C$Del        delete line char        ^X
00045 0023   0D           dc.b     C$CR         end of record char      <cr>
00046 0024   1B           dc.b     C$EOF        end of file char        ESC
00047 0025   04           dc.b     C$Rprt       reprint line char       ^D
00048 0026   01           dc.b     C$Rpet       dup last line char      ^A
00049 0027   17           dc.b     C$Paus       pause char              ^W
00050 0028   03           dc.b     C$Intr       Keyboard Interrupt char ^C
00051 0029   05           dc.b     C$Quit       Keyboard Quit char      ^E
00052 002a   08           dc.b     C$Bsp        backspace echo char     ^H
```

```
00053 002b   07          dc.b    C$Bell      line overflow char      ^G
00054 002c   00          dc.b    Parity      stop bits and parity    none
00055 002d   0E          dc.b    BaudRate    bits/char and baud rate  none
00056 002e=0000          dc.w    EchoNam     offset of echo device   none
00057 0030   11          dc.b    C$XOn       Transmit Enable char    ^Q
00058 0031   13          dc.b    C$XOff      Transmit Disable char   ^S
00059 0032   09          dc.b    C$Tab       tab character
00060 0033   00          dc.b    tabsize     tab column size
00061  0000001c OptSiz    equ     *-Options
00062
00063 0034 5363 FileMgr   dc.b    "Scf",0     file manager
00080
00081  00000023 Mode      set     ISize_+Updat_  default dev mode capabil.
00082
00010  00000040          ends
00011
Errors: 00000
Memory used: 31k
Elapsed time: 26 second(s)
```

## *SBF Device Descriptor*

```
Microware OS-9/68000 Resident Macro Assembler V1.9  90/12/07  15:30  Page     1
 mt0_sbviper.a
MT0 Device Descriptor - Device Descriptor for Tape controller
00001                    nam      MT0 Device Descriptor
00002
00003                    use      defsfile
00001
00002                    use      ../DEFS/oskdefs.d
00001                    opt      -l
00003                    use      ./systype.d
00001 * System Definitions for MVME147 System
00002 *
00003                    opt      -l
00004
00005
00004                    use      ../DEFS/sbfdesc.d
00001                    ttl      Device Descriptor for Tape controller
00002
00027  00000005 Edition    equ      5            current edition number
00028
00029
00030  00000f00 TypeLang   set      (Devic<<8)+0
00031  00008000 Attr_Rev   set      (ReEnt<<8)+0
00032                    psect    SBFDesc,TypeLang,Attr_Rev,Edition,0,0
00033
00034 0000 fffe           dc.l     Port          port address
00035 0004   45           dc.b     Vector        vector trap assignment
00036 0005   04           dc.b     IRQLevel      IRQ hardware interrupt level
00037 0006   05           dc.b     Priority      irq polling priority
00038 0007   67           dc.b     Mode          device mode capabilities
00039 0008 002c           dc.w     FileMgr       file manager name offset
00040 000a 0030           dc.w     DevDrv        device driver name offset
00041 000c 0038           dc.w     DevCon        device constants offset
00042 000e 0000           dc.w     0,0,0,0       reserved
00043 0016 0014           dc.w     OptLen
00044
00045 * Default Parameters
00046           OptTbl
00047 0018   03           dc.b     3             DT_SBF device type
00048 0019   00           dc.b     DrvNum        drive number
00049 001a   00           dc.b     0             reserved
00050 001b   08           dc.b     NumBlks       maximum number of block buffers
00051 001c 0000           dc.l     BlkSize       block size
00052 0020 03e8           dc.w     DrvPrior      driver process priority
00053 0022   00           dc.b     SBFFlags      file manager flags
00054 0023   00           dc.b     DrivFlag      driver flags
00055 0024 0000           dc.w     DMAMode       DMA type/usage
00056 0026   04           dc.b     ScsiID        controller ID on SCSI bus
00057 0027   00           dc.b     ScsiLUN       tape drive LUN on controller
00058 0028 0000           dc.l     ScsiOpts      scsi option flags
00059  00000014 OptLen     equ      *-OptTbl
00060
```

```
00061 002c 5342 FileMgr    dc.b     "SBF",0       Random block file manager
00062
00063
00064          SBFDesc    macro
00065
00066          Port       equ      \1 Port address
00067          Vector     equ      \2 autovector number
00068          IRQLevel   equ      \3 hardware interrupt level
00069          Priority   equ      \4 polling priority
00070          DevDrv     dc.b     "\5",0 driver module name
00071                     ifgt     \#-5 standard device setup requested?
00072
00073
00074                     endc
00075                     endm
00076
00077 *****************
00078 * Descriptor Defaults

MT0 Device Descriptor - Device Descriptor for Tape controller
00079  00000067 Mode       set      Share_+ISize_+Exec_+Updat_
00080 *DevCon   set 0
00081  00000000 Speed      set      0           driver defined
00082  00000002 NumBlks    set      2
00083  00002000 BlkSize    set      0x2000
00084  00000100 DrvPrior   set      256
00085  00000000 SBFFlags   set      0
00086  00000000 DrivFlag   set      0
00087  00000000 DMAMode    set      0           driver defined
00088  00000000 ScsiID     set      0
00089  00000000 ScsiLUN    set      0
00090
00091 * scsi options flag definitions
00092
00093  00000001 scsi_atn   set      1<<0        assert ATN supported
00094  00000002 scsi_target set     1<<1        target mode supported
00095  00000004 scsi_synchr set     1<<2        synchronous transfers supported
00096  00000008 scsi_parity set     1<<3        enable SCSI parity
00097  00000000 ScsiOpts   set      0           scsi options flags
00098
00005
00006 * user changable device descriptor defaults
00007
00008  00000000 DrvNum     set      0           drive number
00009  00000008 NumBlks    set      8           number of blocks (buffered)
00010  00008000 BlkSize    set      0x8000      LOGICAL block size (MUST be multiple of 512)
00011  000003e8 DrvPrior   set      1000        priority of "sbf" process
00012
00013  00000045 IRQVect    set      69          vector to use
00014  00000004 IRQLev     set      4           hardware interrupt level
00015  00000005 IRQPrior   set      5           polling priority (within vector)
00016
00017  00000004 ScsiID     set      4           scsi id of viper
00018  00000000 ScsiLUN    set      0           viper lun always 0.
```

```
00019
00023
00024 *********************
00025 *
00026 *  SBFDesc macro:  port, vector, IRQlevel, IRQpriority, driver name.
00027 *
00028                   SBFDesc  PortAddr,IRQVect,IRQLev,IRQPrior,"sbviper"
00036 0038 7363 DevCon     dc.b     "scsi147",0
00037
00038  00000001 ScsiOpts   set      scsi_atn       disconnect supported
00039  00000040            ends
00040
```

*End of Appendix A*

## NOTES