

# AARD Source Code Text

Brett Glass

```
. *****
;
; AARD.ASM                Created 9/28/1998 by Brett Glass
. *****
;
; This TSR triggers the "AARD" code, which detects non-Microsoft
; versions of DOS, in Windows 3.1. The purpose is to coax Windows
; into displaying the controversial message. This code was made
; possible by the research of Geoff Chappell and Andrew Schulman.
;
; Copyright (C) 1998 by L. Brett Glass. All rights reserved.
;
; Redistribution and use in source and binary forms, with or without
; modification, are permitted provided that the following conditions
; are met:
;
; 1. Redistributions of source code must retain the above copyright
; notice, this list of conditions and the following disclaimer.
; 2. Redistributions in binary form must reproduce the above copyright
; notice, this list of conditions and the following disclaimer in the
; documentation and/or other materials provided with the distribution.
; 3. The name(s) of the author(s) may not be used to endorse or promote
; products derived from this software without specific prior written
; permission.
;
; THIS SOFTWARE IS PROVIDED BY THE AUTHOR(S) "AS IS" AND ANY EXPRESS OR
; IMPLIED WARRANTIES, INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES
; OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE ARE DISCLAIMED.
; IN NO EVENT SHALL THE AUTHOR(S) BE LIABLE FOR ANY DIRECT, INDIRECT,
; INCIDENTAL, SPECIAL, EXEMPLARY, OR CONSEQUENTIAL DAMAGES (INCLUDING, BUT
; NOT LIMITED TO, PROCUREMENT OF SUBSTITUTE GOODS OR SERVICES; LOSS OF USE,
; DATA, OR PROFITS; OR BUSINESS INTERRUPTION) HOWEVER CAUSED AND ON ANY
; THEORY OF LIABILITY, WHETHER IN CONTRACT, STRICT LIABILITY, OR TORT
; (INCLUDING NEGLIGENCE OR OTHERWISE) ARISING IN ANY WAY OUT OF THE USE OF
; THIS SOFTWARE, EVEN IF ADVISED OF THE POSSIBILITY OF SUCH DAMAGE.
;
. *****
;
; Building AARD.COM from AARD.ASM
. *****
;
; To build this program with Borland's TASM (Turbo Assembler) and
; TLINK (Turbo Linker), issue the following commands from the DOS
; prompt or in a batch file:
;
; tasm /zn aard,aard,aard,aard
; tlink /Tdc aard
;
; Note that the Borland linker can create .COM files directly
; and does not require you to use the EXE2BIN utility.
```

**AARD Source Code Text**  
**Brett Glass**

```
.
;
;
; *****
;
; Interrupts
; *****

DOS    equ 21h ; Interrupt number for: DOS function
TSR    equ 27h ; Terminate/stay resident function
MPX    equ 2Fh ; Multiplex interrupt for IO.SYS, TSRs

; *****
;
; DOS function codes
; *****

PRINTMSG equ 9    ; Print message
SETVEC   equ 25h  ; Set interrupt vector
GETDOSVER equ 30h ; Get DOS version (possibly from SETVER)
GETVEC   equ 35h  ; Get interrupt vector
CTRYINFO equ 38h  ; Get/Set Country Info
OPENFILE equ 3Dh  ; Open file
CLOSEFILE equ 3Eh ; Close file
READFILE equ 3Fh  ; Read from file
WRITEFILE equ 40h ; Write to file
SEEK     equ 42h  ; Move file pointer
TERMINATE equ 4Ch ; Terminate program
GETSYSVARS equ 52h ; Find list of lists
          ; (Undocumented API)

; *****
;
; DOS file access constants
; *****

READWRITE equ 2    ; Open for reading and writing
DENYALL   equ 10h  ; Exclusive access
ABSOLUTE  equ 0     ; Code for an absolute (not relative) seek

; *****
;
; Offsets
; *****

FCBSFT   equ 1Ah ; Offset of FCB-SFT pointer in list of lists
CASEMAP  equ 18  ; Offset of case mapping routine address in
              ; buffer returned by CTRYINFO
AARDTAG  equ 3DD5h ; Offset of AARD tag in WIN.COM
AARDFLAG equ 15D4h ; Offset of AARD enable flag in WIN.COM
```

**AARD Source Code Text**  
**Brett Glass**

```
. *****
;
; Exit codes
. *****
;

WIN    equ 1 ; Windows running
ALREADY equ 2 ; Already installed
BADDOS  equ 3 ; Bad DOS version
NOTFOUND equ 4 ; WIN.COM File not found or cannot be opened
PATCHERR equ 5 ; Some other error occurred patching WIN.COM

. *****
;
; Set the stage for a .COM program with ORG 100h (required) and
; "assume" directives
. *****
;

Code    segment para public 'Code'
        assume cs:Code,ds:nothing,ss:nothing,es:nothing
        org 100h
Entry:  jmp near ptr Init ; Jump over resident portion of code
        ; to transient portion

. *****
;
; Data storage for resident code
. *****
;
OldDosInt label dword ; Make a label for the whole vector
OldDosOfs dw 0 ; Now make labels for segment and offset
OldDosSeg dw 0

OldMpxInt label dword ; Make a label for the whole vector
OldMpxOfs dw 0 ; Now make labels for segment and offset
OldMpxSeg dw 0

CaseMapVec label dword ; Make a label for the whole vector
CaseMapOfs dw 0 ; Now make labels for segment and offset
CaseMapSeg dw 0

. *****
;
; Multiplex Interrupt Handler
. *****
;
; The following multiplex interrupt handler (for Int 2Fh)
; signals that we're resident, so that we can detect an
; existing copy and not load another. We look for the TSR ID
; "AARD" in registers AX and BX (with the high bit of AX set
; to avoid collisions with Microsoft device drivers). We do
```

## AARD Source Code Text

Brett Glass

; byte swaps on AX and BX if we see our ID.

```
MpxISR proc far
    pushf          ; Save the flags
    cmp  ax,"AA" or 8000h ; Is interrupt for me?
    jne  MpxNotMe  ; Code is "AARD" in AX and BX
    cmp  bx,"RD"   ; with high bit of AX set per
    jne  MpxNotMe  ; Microsoft convention.
    xchg ah,al     ; We signal we're present
    xchg bh,bl     ; by doing two byte swaps
    popf          ; Break the Mpx chain and return
    iret
MpxNotMe:
    popf          ; Not for me? Pass the buck to
    jmp  cs:[OldMpxInt] ; the next Int 2Fh handler
MpxISR endp
```

```
. *****
;
; DOS Function Call Interceptor
. *****
;
; This handler intercepts DOS function 38h (Get/Set Country Info)
; for the default country code (00). It alters the pointer that's
; returned for the default case mapping function so that its
; segment is not the same as the DOS data segment. We could do this
; by denormalizing the pointer, but it might mess up CS-relative
; addressing within the mapping routine. So, instead, we vector
; the routine through a small stub of code just after this handler.
```

```
DosISR proc far
    cmp  ax,(CTRYINFO shl 8) + 00h ; Function of interest?
    jne  DosNotMe  ; If not, begone
    cmp  dx,0FFFFh ; Setting country info?
    jne  DosFixResults ; If not, must be getting it.
DosNotMe:
    jmp  cs:[OldDosInt] ; Otherwise, begone
DosFixResults:
    ; Before the call, save ds:dx just to be sure
    push ds        ; Save segment of result buffer
    push dx        ; Save offset of result buffer
    pushf         ; Push flags to simulate INT
    call cs:[OldDosInt] ; Call DOS. It'll return to us.
    pop  dx        ; Bring back ds:dx. They should
    pop  ds        ; not have changed, but who knows
                ; what evil lurks in the heart of
                ; DOS?
```

## AARD Source Code Text

Brett Glass

```
    jnc  DosOK      ; If carry, there was an error;
                    ; let the caller deal with it.
    retf 2         ; Return to caller, removing
                    ; caller's saved flags from stack
DosOK:
    ; Change the results. Alter no registers.
    pushf         ; Save DOS's flags
    push  ax      ; Save registers we'll use
    push  si
    mov   si,dx
    add  si,CASEMAP
    mov  ax,ds:[si]
    mov  cs:[CaseMapOfs],ax ; Save routine offset
    mov  [si],offset CaseMapJump ; Point to vector
    add  si,2
    mov  ax,ds:[si]
    mov  cs:[CaseMapSeg],ax ; Save routine segment
    mov  [si],cs      ; And vector to it
    pop  si           ; Restore si to what it was
    pop  ax           ; Same for ax
    popf             ; And DOS's returned flags
    retf 2           ; Return from interrupt
                    ; but do not restore caller's
                    ; flags. This lets DOS signal
                    ; errors, etc. in the flags.

DosISR endp

. *****
;
; Case mapping function stub
. *****
;
CaseMapJump:
    jmp  cs:[CaseMapVec] ; Vector to the case mapping routine

. *****
;
; Beginning of transient data and code
. *****
;

DoneMsg  db 'Installation complete! Enter "WIN" to see the AARD error message.'
         db 0Dh,0Ah,'$'

Copyright:
db 'AARD Code Activation Utility Version 1.02a'
db 0Dh,0Ah,'Copyright (C) 1998 by L. Brett Glass.'
db 0Dh,0Ah
db 'All rights reserved.'
```

## AARD Source Code Text

Brett Glass

```
db 0Dh,0Ah,0Dh,0Ah
db 'Description:'
db 0Dh,0Ah,0Dh,0Ah
db 'When this program is run on a computer with Windows 3.1 and MS-DOS, it'
db 0Dh,0Ah
db 'disguises the fact that the machine is using MS-DOS instead of an'
db 0Dh,0Ah
db 'alternative DOS (e.g. DR-DOS). It also changes a flag byte in the file'
db 0Dh,0Ah
db 'C:\WINDOWS\WIN.COM to re-enable the famous message which Microsoft'
db 0Dh,0Ah
db 'appears to have added to the Windows 3.1 beta to arouse fears about'
db 0Dh,0Ah
db 'DR-DOS compatibility. This program refutes Microsoft''s claim that'
db 0Dh,0Ah
db 'the message was an "urban legend."'
db 0Dh,0Ah
db 0Dh,0Ah
db 'This program is based on the research of Andrew Schulman and Geoff'
db 0Dh,0Ah
db 'Chappell.'
db 0Dh,0Ah,0Dh,0Ah,'$'

BadDosMsg db 'Error: This DOS is either too old or too new to work'
          db 0Dh,0Ah,'with the AARD activation utility or is an emulation.'
          db 0Dh,0Ah,'$'

WinMsg    db 'Error: Windows is running. Please quit'
          db 0Dh,0Ah,'Windows, then run the AARD utility again.'
          db 0Dh,0Ah,'$'

AlreadyMsg db 'Warning: The AARD utility is already installed.'
          db 0Dh,0Ah,'To remove it, reboot the computer.'
          db 0Dh,0Ah,'$'

WinComMsg db 'Looking for C:\WINDOWS\WIN.COM....'
          db 0Dh,0Ah,'$'

NotFoundMsg db 'Error: Could not find and open C:\WINDOWS\WIN.COM.'
          db 0Dh,0Ah,'$'

PatchErrMsg db 'Error: Could not patch C:\WINDOWS\WIN.COM to resurrect'
            db 0Dh,0Ah,'the AARD code. The file might be in an unusual place,'
            db 0Dh,0Ah,'or you might not have Windows 3.1.'
            db 0Dh,0Ah,'$'
```

## AARD Source Code Text

Brett Glass

```
PatchedMsg db 'Patched your WIN.COM file to enable the error message.'  
            db 0Dh,0Ah,'Intercepting DOS calls to emulate DR-DOS....'  
            db 0Dh,0Ah,'$'
```

```
FCBMsg     db 'Changing the FCB-SFT pointer to emulate DR-DOS....'  
            db 0Dh,0Ah,'$'
```

```
WinComPath db 'C:\WINDOWS\WIN.COM',0h ; ASCIIZ path to WIN.COM
```

```
One        db 1 ; One byte to patch WIN.COM
```

```
SmallBuf   dd 0 ; Four bytes to hold file data
```

```
Init       proc near ; Transient initialization routine  
            assume ds:Code,ss:Code ; Only during setup
```

```
; Say hello
```

```
    mov     dx,offset Copyright ; Prepare to print  
    mov     ah,PRINTMSG  
    int     DOS
```

```
; Check DOS version number here. If it's greater than 9,  
; this isn't really DOS. If it's less than 3, who knows  
; what functions are implemented?
```

```
    mov     ah,GETDOSVER  
    int     DOS  
    cmp     al,3 ; Older than DOS 3? Report error.  
    jb     BadDosVer  
    cmp     al,10  
    jb     DosVerOK
```

```
BadDosVer:
```

```
    mov     dx,offset BadDosMsg  
    mov     ax,PRINTMSG  
    int     DOS  
    mov     ax,(TERMINATE shl 8 + BADDOS) ; Exit with code  
    int     DOS
```

```
DosVerOK:
```

```
; Next, detect if Windows is running.
```

```
    mov     ax,1600h ; Windows multiplex presence check  
    int     MPX ; Issue multiplex interrupt 2Fh  
    and     al,7Fh ; Windows not running if al=0 or 80h
```

## AARD Source Code Text

Brett Glass

```
    jz    NoWindows
WinError:
    mov   dx,offset WinMsg ; Complain
    mov   ah,PRINTMSG
    int   DOS
    mov   ax,(TERMINATE shl 8 + WIN) ; Exit with code
    int   DOS
```

; Now check to see if we've already been loaded.

```
NoWindows:
    mov   ax,("AA" or 8000h) ; Our secret code
    mov   bx,"RD"
    int   MPX                ; Scan the multiplex chain
    cmp   ax,("AA" or 0080h) ; Were the bytes swapped?
    jne   EnableAARD
    cmp   bx,"DR"
    jne   EnableAARD
    mov   dx,offset AlreadyMsg ; Warn
    mov   ah,PRINTMSG
    int   DOS
    mov   ax,(TERMINATE shl 8 + ALREADY) ; Exit with code
    int   DOS
```

; It doesn't pay to install if WIN.COM can't be patched to  
; re-enable the AARD code, so do that next.

```
EnableAARD:
    mov   dx,offset WinComMsg ; Tell user we're opening file
    mov   ah,PRINTMSG
    int   DOS
    mov   dx,offset WinComPath ; Open WIN.COM
    mov   ax,(OPENFILE shl 8 + READWRITE + DENYALL)
    int   DOS                ; If we're successful, we get a handle in AX
    jnc   OpenOK
    mov   dx,offset NotFoundMsg
    mov   ah,PRINTMSG
    int   DOS
    mov   ax,(TERMINATE shl 8 + NOTFOUND) ; Exit with code
    int   DOS
```

```
OpenOK:
    mov   bx,ax                ; Prepare for a seek
    xor   cx,cx
```

## AARD Source Code Text

Brett Glass

```
mov  dx,AARDTAG ; To offset 3DD5h
mov  ax,(SEEK shl 8 + ABSOLUTE) ; Seek to absolute offset
int  DOS
jc   CloseWithError ; If seek fails, try to clean up

mov  cx,4 ; Read just four bytes
mov  dx,offset SmallBuf ; Into buffer
mov  ah,READFILE
int  DOS
jc   CloseWithError ; If error, try to clean up
cmp  ax,4 ; Must have gotten four bytes
jne  CloseWithError
cmp  word ptr [SmallBuf],"AA" ; Windows 3.1 WIN.COM has
jne  CloseWithError ; the programmer's "tag" here
cmp  word ptr [SmallBuf+2],"DR"
jne  CloseWithError

xor  cx,cx ; Now seek again
mov  dx,AARDFLAG ; To offset of flag
mov  ax,(SEEK shl 8 + ABSOLUTE) ; Seek to absolute offset
int  DOS
jc   CloseWithError ; If seek fails, try to clean up

mov  cx,1 ; Patch the one-byte flag
mov  dx,offset One ; Point to the data to overwrite it
mov  ah,WRITEFILE
int  DOS
jnc  CloseOK

CloseWithError:
mov  ah,CLOSEFILE ; Try to close; ignore further errors
int  DOS

CloseProblem:
mov  dx,offset PatchErrMsg
mov  ah,PRINTMSG ; Complain
int  DOS
mov  ax,(TERMINATE shl 8 + PATCHERR)
int  DOS

CloseOK:
mov  ah,CLOSEFILE ; Close and do NOT ignore errors
int  DOS
jc   CloseProblem
mov  dx,offset PatchedMsg ; If whole process went OK, say so
mov  ah,PRINTMSG
int  DOS
```

## AARD Source Code Text

Brett Glass

; OK, now we need to install ourselves. Hook DOS and multiplex  
; interrupts.

NewInstall:

```
mov ax,(GETVEC shl 8) + DOS ; Get interrupt vector for DOS
int DOS
mov [OldDosOfs],bx ; Save the old vector
mov [OldDosSeg],es
mov dx,offset DosISR ; Get ready to add our own handler
mov ah,SETVEC ; Set the interrupt vector
int DOS ; DOS call
```

```
mov ax,(GETVEC shl 8) + MPX ; Get MPX interrupt vector
int DOS
mov [OldMpxOfs],bx ; Save the old vector
mov [OldMpxSeg],es
mov dx,offset MpxISR ; Get ready to add our own handler
mov ah,SETVEC ; Set the interrupt vector
int DOS ; DOS call
```

; Now, we must tweak the FCB-SFT pointer, as per Chappell and  
; Schulman. When we're done, the pointer still points to the  
; same place, but its segment is no longer that of DOS's data  
; area.

```
mov dx,offset FCBMsg ; Say what we're up to
mov ah,PRINTMSG
int DOS
```

```
mov ax,(GETSYSVARS shl 8) ; Undocumented DOS call returns
; pointer to "list of lists" in
; ES:BX (See Schulman et al)
int DOS
add word ptr es:[bx+FCBSFT],40h ; Add 40h to FCB-SFT offset
sub word ptr es:[bx+FCBSFT+2],4h ; And subtract 4h from segment
```

; Finally, we can go resident. Microsoft's MS-DOS reference says  
; that Int 27h is deprecated as a way of going resident, but on the  
; other hand, ALL of DOS is deprecated nowadays and Int 27h still  
; works just fine. So, let's go for it.

```
mov dx,offset DoneMsg ; Prepare to print message
mov ah,PRINTMSG ; Print message through DOS
int DOS ; DOS call
```

## AARD Source Code Text

Brett Glass

```
int TSR      ; Terminate and stay resident. Final
              ; message was at top of transient portion,
              ; so dx need not be reloaded.
Init  endp
Code  ends
end   Entry
```