```
; File: Rom4EQU

TRUE        .EQU        $FFFF
FALSE       .EQU        $0000

ScrnSize    .EQU        <ScrnLSize>*4           ;size of screen in bytes
sizeCmds    .EQU        42*4                    ; # commands (this is max for
fullsize/noterm/withdis)
kbdRoll     .EQU        70                      ; loop value of hitting keyboard
kbdWait     .EQU        60                      ; loop value for waiting for keyboard

            .MACRO      MAKEMARK
            MOVE.L      A0,-(SP)                ;save A0
            MOVE.L      ScrnBase,A0             ;get base of screen
            .IF         '&2' = 'DARK'
            MOVE.L      minusOne,&1(A0)         ;mark 32 pixels black
            .ELSE
            CLR.L       &1(A0)                  ;clear a stripe
            .ENDC
            MOVE.L      (SP)+,A0                ;restore A0
            .ENDM

;The following set of equates are if this is the MacsBug for running under
;MacWorks on a Lisa.  This implies that the equate files used to assemble
;the debugger are also the MacWorks equate files.

            .IF         onLisaTrue               ;kwk - twm
dbgHead     .EQU        $410900                 ;start of debugger globals
SYSTACK     .EQU        $411000                 ;grow down above driver globals,
                                                ;below the dispatch table
dSpace      .EQU        dLines*8*<LineLen>+90   ;number of bytes for display
                                                ;add some slop so eight bytes
                                                ;immediately past end of screen
                                                ;memory don't get cleared
dOffset     .EQU        <ScrnSize>+2-dSpace     ;offset bytes for display

;The following equates are used to communicate with the Lisa hardware interface.
;The technique is to use the TRAPTO macro with one of the following equates,
;after setting up the appropriate registers.  All registers without explicit
;return values are preserved.
;NOTE -- Other TRAPTO equates are already defined in GrafEqu/SysEqu

_DriverInit     .EQU      0
_KeyMap         .EQU     54
_KeybdEvent     .EQU     58
_MMISync        .EQU    140
_ScreenKeybd    .EQU    142
_Poll           .EQU    158
_COPSSync       .EQU    170

;These are the standard set of equates for the family of MacsBug debuggers
;running on a Mac.

            .ENDC                               ;(kwk)
            .IF         onMacTrue               ;twm
dSpace      .EQU        dLines*8*<LineLen>      ;number of bytes for display
dOffset     .EQU        <ScrnSize>+2-dSpace     ;offset bytes for display
screenA     .EQU        $FA700                  ;base address of screen (for 512K, wraps
for 128K)
dbgHead     .EQU        screenA-$400            ;hardwired global area
SYSTACK     .EQU        screenA                 ;stack grows down
            .ENDC                               ;(kwk) - twm
```

```
; These are the standard Mac equates modified for the Yacc.

            .IF         onYaccTrue                  ;twm
dSpace      .EQU        dLines*8*<LineLen>          ;number of bytes for display
dOffset     .EQU        <ScrnSize>+2-dSpace         ;offset bytes for display
screenA     .EQU        ScreenLow                   ;base address of screen -
psuedo-absolute for Yacc
dbgHead     .EQU        DispatchTab-$800            ;hardwired global area
SYSTACK     .EQU        DispatchTab-$50             ;stack grows down - skip DeepShit Reg
area
            .ENDC                                   ;twm

.IF         on68000=0                               ; twm - on 68010
ATrapFmt    .equ        $0028                       ; 68010's format/vector constant for an
'1010' trap
.ENDC

rSYSCmds    .EQU        dbgHead                     ; list of commands

REGPC       .EQU        rSYSCmds+sizeCmds           ; saved value of user's PC
REGSR       .EQU        REGPC+4                     ; saved value of user's Status Register

.IF         on68000
REGS        .EQU        REGSR+4                     ; saved values of user's data registers
.ELSE
REGFMT      .EQU        REGSR+4                     ; exception format value
REGS        .EQU        REGFMT+2                    ; saved values of user's data registers
.ENDC

AREGS       .EQU        REGS+32                     ; saved values of user's address
registers (A0-A4)
USERA5      .EQU        AREGS+20                    ; saved value of user's A5 reg
REGA6       .EQU        AREGS+24                    ; saved value of user's A6 reg
REGA7       .EQU        AREGS+28                    ; saved value of user's A7 reg
REGUS       .EQU        REGA7+4                     ;

            .IF         DDBG
DDBG1       .EQU        REGUS+4                     ;space for one long temp
DDBG2       .EQU        DDBG1+4                     ;another debugging the debugger temp
keyWait     .EQU        DDBG2+4                     ; keyboard constant (waiting for token)
keyRoll     .EQU        keyWait+2                   ; keyboard constant (delay or rollover
time)

lastCmd     .EQU        keyRoll+2                   ; last two letter command
            .ELSE
lastCmd     .EQU        REGUS+4                     ; last two letter command
            .ENDC

BPADD       .EQU        lastCmd+2
BPTILL      .EQU        BPADD+32                    ; SPACE FOR 8 BREAKS
BPCNT       .EQU        BPTILL+4
BPDATA      .EQU        BPCNT+36
SAVETRAP    .EQU        BPDATA+18

; The following A(whatever) equates must all be together (all flags for A-trap debugger
cmds)
; Must also be exactly 6 bytes long

ABreak      .EQU        SAVETRAP+4                  ; break A traps
AHeap       .EQU        ABREAK+1                    ; check heap on A traps
ATrace      .EQU        AHeap+1                     ; trace A traps
```

```
ASpy            .EQU        ATRACE+1            ; data spy on A traps
AScramble       .EQU        ASpy+1              ; scramble heap on a traps
ARecord         .EQU        AScramble+1         ; record trap call info

GotAarg         .EQU        ARecord+1           ; GetAArgs was passed value
ShowPC          .EQU        GotAArg+1           ; ST => print out PC location during
disassembly
ATrapSave       .EQU        ShowPC+1            ; space for one ATrap record

showSyms        .EQU        ATrapSave+50        ; <> 0 => show pascal symbols
AbortPrint      .EQU        showSyms+1          ; abort printing flag
LastRoutine     .EQU        AbortPrint+1        ; ST if LookupPC called w/PC in last
routine
SymFound        .EQU        LastRoutine+1       ; ST if LookupPC found a name

UseSysHeap      .EQU        SymFound+1          ; ST if system heap
BugHeap         .EQU        UseSysHeap+2        ; heap address to use
heapMask        .EQU        BugHeap+4           ; mask value for heap
printEntry      .EQU        heapMask+4          ; 0 => print heap entry, otherwise skip
it
maskTotal       .EQU        printEntry+2        ; total # of bytes of objects in heap
selected by mask
maskCount       .EQU        maskTotal+4         ; total # of objects in heap selected by
mask

ALOW            .EQU        maskCount+2         ; low limit for A trap
AHIGH           .EQU        ALOW+2              ; high limit for A trap
ALowPC          .EQU        AHIGH+2             ; low PC limit (or data spy)
AHighPC         .EQU        ALowPC+4            ; high PC limit
ALowD0          .EQU        AHighPC+4           ; low D0 limit (or data spy)
AHighD0         .EQU        ALowD0+4            ; high D0 limit
ASAVEPC         .EQU        AHighD0+4           ; Saved PC for breaks
sumPlace        .EQU        ASAVEPC+4           ; saved checksum from AS/TS commands

CSlow           .EQU        sumPlace+4          ; low address for checksum calc
CShigh          .EQU        CSlow+4             ; high address for checksum
CSsum           .EQU        CShigh+4            ; saved checksum from CS command

findMask        .EQU        CSsum+4             ; what to mask memory value with before
find compare
findWidth       .EQU        findMask+4          ; width of data to find (byte/word/long)
findData        .EQU        findWidth+4         ; data to find
findLength      .EQU        findData+4          ; how long to search
findStart       .EQU        findLength+4        ; starting address

TRACECNT        .EQU        findStart+4         ; current # of traces done if limit set
traceSpy        .EQU        TRACECNT+4          ; ST if trace spy
traceTill       .EQU        traceSpy+1          ; ST if trace till
tracePC         .EQU        traceTill+1         ; pc for trace till
traceGo         .EQU        tracePC+4           ; ST if go-trace

RUN             .EQU        traceGo+1           ; ST if user is running
BPSTATUS        .EQU        RUN+1               ;
BASE            .EQU        BPSTATUS+2          ; for parsing input value in ReadToken
SIGN            .EQU        BASE+2              ; sign of input value in ReadToken

TEMP            .EQU        Sign+2              ; save location for address of cmd
routines, etc.
WORK1           .EQU        TEMP+4              ; another temp register
LOCSAVE         .EQU        WORK1+4             ; flag and a long to hold location
OPCOD           .EQU        LocSave+12          ; string for opcode
OPERAND         .EQU        OPCOD+12            ; string buffer for operand
```

```
smallMode   .EQU        operand+64          ; no reg dump on trace
noRegs      .EQU        smallMode+1         ; used with above
dotAddress  .EQU        noRegs+1            ; last address used
magicPC     .EQU        dotAddress+4        ; saved PC from MR command
magicSR     .EQU        magicPC+4           ; saved SR from MR command
trapNum     .EQU        magicSR+2           ; trap number saved from parsing cmd
line
outAddress  .EQU        trapNum+2           ; scc to write out
postEvent   .EQU        outAddress+4        ; address of PostEvent trap routine

keyHit      .EQU        postEvent+4         ; key hit
keyEvt      .EQU        keyHit+4            ; last event number

swapped     .EQU        keyEvt+2            ; swapped flag
saveKeybd   .EQU        swapped+2           ; old keyboard handler routine (Lisa)
offscreen   .EQU        saveKeybd+4         ; pointer to offscreen

DMcmdPtr    .EQU        offScreen+4         ; ptr to next memory display command
byte
DMmemPtr    .EQU        DMcmdPtr+4          ; ptr to current memory to display
DMmemEnd    .EQU        DMmemPtr+4          ; ptr to last memory to display

BUFFER      .EQU        DMmemEnd+4          ; space for 80 charactres

ReEntrFlg   .EQU        BUFFER+80           ; twm - count of times NMI switch
bounced & bounced & ...

debugEnd    .EQU        ReEntrFlg+4         ; end of vars
dbgWrdCnt   .EQu        <debugEnd-REGPC>/2  ; number of words used by the debugger

; EOFile: Rom4EQU
```