



Hacking n'est pas Hijacking
Le hacking est la recherche et le partage des connaissances...
en aucune façon le vol des autres !...

IDENTIFICATION SERVEUR : 178.33.103.129 - CLIENT : 203.214.157.120

[FAQ](#)
[Rechercher](#)
[Liste des Membres](#)
[Groupes d'utilisateurs](#)
[S'enregistrer](#)

[Profil](#)
[Se connecter pour vérifier ses messages privés](#)
[Connexion](#)

Lode Runner (Broderbund, 1983)

[nouveau](#)
[répondre](#)

Index du Forum -> **PROTECTION MALEFIQUE**

[Voir le sujet précédent :: Voir le sujet suivant](#)

Auteur	Message
toinet Site Admin Inscrit le: 15 Juin 2007 Messages: 1154 Localisation: Le Chesnay, France	<div style="display: flex; justify-content: space-between;"> Posté le: Mar 28 Oct 2008, 7:59 Sujet du message: Lode Runner (Broderbund, 1983) citer </div> <p>You are a miner. You must collect all the coffins, avoid foes and escape from each level.</p> <p>That game is one of my favorites: easy rules, fantastic gameplay, hours of entertainment. It is one of the best selling software of the Apple II.</p> <p>The program is heavily protected:</p> <ul style="list-style-type: none"> - boot code is coded - code executed in the zero page area - code loaded in the text screen area - all vectors redirected to the reboot code - jumps to different parts of code through the stack - on-disk protection with half-tracks (the sound of the head moving is nice) - nibbles on disk are 4*4 coded <p>But... it has passed the LoGo's exam 😊</p> <p>The following table summarizes the memory usage:</p> <ul style="list-style-type: none"> - \$0060..\$010F: boot stage / phase 1 (oh! the stack) - \$0200..\$02FF: reboot code - \$0400..\$04FF: final boot / inits (replaces data of phase 2) - \$0400..\$07FF: boot stage / phase 2 - \$0800..\$08FF: boot stage / phase 0 - \$0F00..\$1EFF: program - \$6000..\$BFFF: program <p>Final jump to \$6000 once code at \$0400 has been executed.</p> <p>Let me find my external USB CF reader and I will upload the boot source codes.</p> <p>What a grat program!!!!</p> <p>Antoine 10/2008</p> <div style="display: flex; align-items: center;"> Revenir en haut de page <div style="margin-left: 10px;"> profil mp www </div> </div>
toinet Site Admin Inscrit le: 15 Juin 2007 Messages: 1154 Localisation: Le Chesnay, France	<div style="display: flex; justify-content: space-between;"> Posté le: Jeu 30 Oct 2008, 8:09 Sujet du message: citer </div> <p>Second digging into the disk organization of Lode Runner...</p> <p>TRACK \$00</p> <ul style="list-style-type: none"> - sector 0: first boot loaded at \$0800..\$08FF - other pseudo 4*4 sectors: \$0400..\$07FF <p>TRACKS \$01-\$02 : NO DATA</p> <p>TRACKS \$03-\$0C : GAME LEVELS</p> <ul style="list-style-type: none"> - track \$0C / sector \$ 0F : score board loaded at \$1F00..\$1FFF - sectors can be copied with DEMUFFIN once \$B942:18 is set <p>TRACKS \$0D-\$1E : PROGRAM CODE</p> <ul style="list-style-type: none"> - eight 4*4 sectors per half-track (\$0D and \$0D.5) - then move to next 1.5 track - data is loaded from \$0F00..\$1EFF - program is loaded from \$6000..\$BFFF - \$B700..\$BFFF is a standard DOS 3.3 RWTS

TRACK \$21 : PROGRAM INIT
- last boot phase loaded at \$0400..\$04FF

Now that we have gathered all the necessary information about the program, there are different ways to create a standard and copyable diskette:

- rewrite the boot code at \$0400..\$07FF using the \$Cx5C routine
- use a fastboot code (the one from EA games)
- use the standard DOS 3.3 RWTS track \$00 code and use it to load the game.

As always, I would like to minimize the changes of the original code. As the \$0900..\$0EFF RAM space is free, it can be used for our routines.

My choice has not been decided yet, that fantastic game needs some good load routines...

Antoine
10/2008

[Revenir en haut de page](#) [profil](#) [tmp](#) [www](#)

toinet
Site Admin

Posté le: Lun 03 Nov 2008, 8:39 Sujet du message:

[citer](#)

Inscrit le: 15 Juin 2007
Messages: 1154
Localisation: Le Chesnay,
France

And now, introducing you to the boot 1 code of Lode Runner, the one that is loaded from track \$0 / sector \$ 0 at address \$0800:

Code:

```
*
* Lode Runner
* (c) 1983, Broderbund
*
* (k) 2008, LoGo
*
        org     $0800
        lst     off
        mx      %11

*
* Equates
*
TXTCLR    EQU    $C050
MIXCLR    EQU    $C052
TXTPAGE1  EQU    $C054
HIRES     EQU    $C057

*
* Boot code
*
L0800     DB      $01

L0801     LDY      #$00          ; clear
          LDA      #$20          ; HGR & HGR2 pages
          LDX      #$40
          STY      $00
          STA      $01
          TYA
L080C     STA      ($00),Y
          INY
          BNE      L080C
          INC      $01
          DEX
          BNE      L080C

          BIT      MIXCLR        ; HGR mode on
          BIT      HIRES
          BIT      TXTPAGE1
          BIT      TXTCLR

          LDX      $2B          ; save slot*16
          STX      $08

          NOP
          NOP
          LDY      #$00
          NOP
          NOP
L082C     LDA      L0850,Y
          NOP
          NOP
          EOR      #$A5          ; the key
          NOP
          NOP
          STA      |$0060,Y
          NOP
          NOP
          INY
          BNE      L082C

          NOP
          NOP
          LDX      #$FF          ; set stack pointer
          NOP
          NOP
          TXS
          NOP
          NOP
          RTS                  ; Return please...

          HEX      DDAFFD3F12448731

L0850     HEX      239B209F039B23E505A5009F21992098
          HEX      03AD850BA560A5755C850BA560A47550
          HEX      850BA560A77550182965B55E8F209A18
          HEX      2965B55E809A34996D7549ABA5651829
          HEX      65B55E60A67518439863E5757FC51829
          HEX      65B55EC5077123A54D23A44D23A74D23
          HEX      A60CA10FC55A8F8C145AED8FACA42FEC
          HEX      5A6CB50F4DAC0F4F5A07A5ED6F755907
```

```

HEX  AA1855A138A5A46F75523FC5BDCCA08C
HEX  A2CCA4A54F4D6F4D6F4D6F6D2D6D2D6D
HEX  16A5FAA55AA6A5A15A2E5BA25AA65AFA
    
```

Note the non standard way to jump to the next boot stage with the use of the stack. Quite difficult to follow...

[Revenir en haut de page](#) [profil](#) [mp](#) [www](#)

toinet
Site Admin

Posté le: Lun 03 Nov 2008, 8:43 Sujet du message:

[citer](#)

Inscrit le: 15 Juin 2007
Messages: 1154
Localisation: Le Chesnay,
France

The code decyphered from \$0850 to \$0060 is just there...

Code:

```

*
*  Lode Runner
*  (c) 1983, Broderbund
*
*  (k) 2008, LoGo
*
*
*          org    $0060
*          lst    off
*          mx     %11
*
*
*  * Boot code stage 2
*
KBD      EQU    $C000
* Remember $08 contains slot*16
L0060    STX     $3E      ; number of pages
          STA     $3A      ; RAM pointer
L0064    LDX     $3E
          STX     $40
          LDY     #$00
          LDA     $3A
          STY     $3C
          STA     $3D
          LDX     $08      ; slot*16
L0072    JSR     L00AE
L0075    CMP     $00      ; #$D4
          BNE     L0072
          JSR     L00AE
L007C    CMP     $01      ; #$D5
          BNE     L0075
          JSR     L00AE
          CMP     $02      ; #$D6
          BNE     L007C
L0087    LDA     $C08C,X   ; 4*4 coding ;- )
          BPL     L0087
          ROL
          STA     $3F
L008F    LDA     $C08C,X
          BPL     L008F
          AND     $3F
          STA     ($3C),Y
          INY
          BNE     L0087
L009E    ASL     KBD
          LDA     $C08C,X
          BPL     L009E
          CMP     $03      ; #$D7
          BNE     L0064
          INC     $3D      ; next high ram pointer
          DEC     $40      ; decrement number of pages
          BNE     L0087
          RTS
L00AE    LDA     $C08C,X
          BPL     L00AE
          RTS
*
*
*
L00B4    LDX     #$D4
          STX     $00      ; #$D4
          INX
          STX     $01      ; #$D5
          INX
          STX     $02      ; #$D6
          INX
          STX     $03      ; #$D7
          LDA     #$04      ; high pointer
          TAX      ; number of pages
          RTS
          DB     $FF
          ROL
          AND     #$B1
          DB     $FF
          PHA
          ROL
          ORA     #$01
          TXA
          EOR     #$FF
          CMP     #$10
          TAX
          INX
          ORA     #$AA
          NOP
          DB     $FF
*
*
*
    
```

```

L00D9    LDX    #$00
L00DB    PHA
        DEX
        BNE    L00DB

L00E1    LDX    #$0F        ; copy data
        LDA    $04F0,X    ; to stack
        STA    L0100,X
        DEX
        BNE    L00E1
        TXS
        RTS            ; and set index to $0
                        ; JUMP...

        CLC
        ADC    #$05
        AND    #$07
        ADC    #$01
        DB     $00
        NOP
        INX
        DEX
        INX
        DEX
        INX
        INY
        DEY
        INY
        DEY
        INY

*
* Jump with the stack ;- )
*

L0100    DA     L00B4-1    ; init DISK INFO
        DA     L0060-1    ; read DISK
        DA     $0400-1    ; jump RTS
        DA     $0401-1    ; jump CODE
        DA     $8C00-1
        DA     $07FF-1
        DA     $0400-1
        DA     $6000-1

        HEX    A5A5A5A5A5A5A5A5A5A5A5A5A5A5A5A5
        HEX    A5A5A5A5A5A5A5A5A5A5A5A5A5A5A5A5
        HEX    A5A5A5A5A5A5A5A5A5A5A5A5A5A5A5A5
        HEX    A5A5A5A5A5A5A5A5A5A5A5A5A5A5A5A5
        HEX    A5A5A5A5A5A5A5A5A5A5A5A5A5A5A5A5

```

I really appreciate the jump values pushed onto the stack... The entry point is \$00B4, once the RTS is performed, the next call is \$0060. Then \$0400 and the final jump is \$0401...

[Revenir en haut de page](#) [profil](#) [mp](#) [www](#)

toinet
Site Admin

Posté le: Lun 03 Nov 2008, 8:47 Sujet du message:

[citer](#)

The next boot stage on track \$00 which is loaded from \$0400 to \$07FF is included in that message.

Note that page 7 (\$0700..\$07FF) is copied from \$0200..\$02FF. The interesting load routines and arm move are located there.

Once program is loaded from \$0F00..\$1EFF and \$6000..\$BFFF, a final "sector" is loaded from track \$21 at \$0400..\$04FF then a jump is executed.

Code:

```

*
* Lode Runner
* (c) 1983, Broderbund
*
* (k) 2008, LoGo
*

        org    $0400
        lst    off
        mx     %11

*
* Equates
*

SOFTEV    EQU    $03F2
PWREDUP    EQU    $03F4
KBD        EQU    $C000
KBDSTROBE EQU    $C010
SPKR        EQU    $C030
RDBANK2    EQU    $C080
ROMIN2     EQU    $C081
LCBANK2    EQU    $C083
INIT        EQU    $FB2F
BELL1_2    EQU    $FB22
HOME        EQU    $FC58
WAIT        EQU    $FCA8
SETNORM     EQU    $FE84
OLDRST      EQU    $FF59
RESETV      EQU    $FFFC
IRQV        EQU    $FFFE

*
* Boot code stage 3
*

L0400     RTS
        NOP
        NOP
        JSR    L07E0    ; Goodbye wildcards
        NOP
        NOP
        LDA    LCBANK2    ; Copy reboot code

```

```

L0410      LDA      LCBANK2
          LDY      #$00
          LDA      L0700,Y
          STA      $0200,Y
          INY
          BNE      L0410

          LDA      #>L0200      ; reset vector
          STY      RESETV
          STA      RESETV+1
          STY      SOFTEV
          STA      SOFTEV+1
          EOR      #$A5
          STA      PWREDUP

          LDY      #<L0203      ; redirect all vectors
          LDA      #>L0203
          STY      $36
          STA      $37
          STY      $38
          STA      $39
          STY      $03F0
          STA      $03F1

*
*
*

          LDA      #$00      ; hehe
          STA      $0A

          LDX      $2B      ; slot*16
          STX      $02FF
          NOP
          NOP
          NOP
          LDA      #$1A      ; track $0D
          JSR      L0603      ; move
          NOP
          NOP
          JSR      L0550      ; read all
          NOP
          NOP

*
* Final read
*

          LDA      #$DD      ; 1st marker
          STA      $00
          LDA      #$F5      ; 2nd marker
          STA      $01
          LDA      #$D5      ; 3rd marker
          STA      $02
          LDA      #$D4      ; 4th marker
          STA      $03
          NOP
          NOP
          LDA      #$42      ; track $21
          JSR      L0603      ; move
          LDA      #$04      ; read at $0400
          LDX      #$01      ; one sector
          JSR      L0600      ; read
          JMP      $8541      ; JUMP to game...

          HEX      00000000000000000000000000000000
          HEX      00000000000000000000000000000000
          HEX      00000000000000000000000000000000

          JSR      L05C0

L04A3      LDA      KBD
          BPL      L04A3
          BIT      KBDSTROBE
          CMP      #$9B
          BEQ      L04B2
          JMP      BELL1_2

L04B2      NOP
          JMP      OLDRST

          HEX      00000000000000000000000000000000
          HEX      00000000000000000000000000000000
          HEX      00000000000000000000000000000000
          HEX      00000000000000000000000000000000
          HEX      00000000000000000000000000000000

L0500      LDA      #$FF
          JSR      L0540
L0505      CMP      #$DD
          BNE      L0500
          JSR      L0542
L050C      CMP      #$F5
          BNE      L0505
          JSR      L0542
          CMP      #$D5
          BNE      L050C
L0517      JSR      L0539
          STA      L0526+1
          JSR      L0539
          STA      L0526+2
          JSR      L0539
L0526      STA      IRQV+1
          CMP      #$EA
          BNE      L0517
          LDA      $C088,X
          RTS

          AND      $4805,Y
          JSR      L0539
          PHA
          RTS

L0539      LDA      $C08C,X
          BPL      L0539
          SEC
          ROL
L0540      STA      $FF

```

```

L0542    LDA    $C08C,X
        BPL    L0542
        AND    $FF
        RTS

        HEX    000000000000

*
* Main read loop
*
L0550    LDY    #$00        ; loop...
        STY    $05
L0554    STY    $06
        LDA    L0570,Y      ; RAM pointer
        BEQ    L0564
        JSR    L05C0        ; read data
        LDY    $06
        INY
        BNE    L0554        ; ...loop
        DB     $00
L0564    RTS

        JSR    L05C0

L0568    LDA    KBD
        BPL    L0568
        BIT    KBDSTROBE

*
* Where to load data
*
* $0F00..$1EFF
* $6000..$BFFF

L0570    HEX    0F1760687078808898A0A8B0B8000000

*
*
*
L0580    PHA
        LDA    $05        ; loop index
        AND    #$07
        TAY
        LDA    L06F8,Y      ; wonderful markers list
        STA    $00        ; first marker
        LDA    $05
        LSR
        ORA    #$AA
        STA    $01        ; second marker
        LDA    $05
        ORA    #$AA
        STA    $02        ; third marker
        PLA
        INC    $05
        LDY    #$01        ; read one sector
        JMP    L0600        ; read

        DB     $00
        DB     $00
        DB     $00
        DB     $00
        DB     $05

        LDX    #$01
        LDY    $07
        BNE    L05AE
        JMP    $1100

L05AE    PHA
        LDA    #$01
        JSR    WAIT
        PLA
        JMP    $1000

        HEX    0000000000000000

*
* Read $0800 bytes of data
*
* Two adjacent blocks in memory
* are located on half tracks...
*
L05C0    LDY    #$04        ; loop 4
        STY    $04

L05C4    PHA                ; RAM pointer
        LDX    #$01        ; unuseful
        JSR    L0580        ; read sector
        LDA    $0A        ; start with $00
        CLC
        ADC    #$01
        JSR    L0606        ; next phase
        PLA
        CLC
        ADC    #$01        ; RAM pointer++
        PHA
        JSR    L0580        ; read sector
        LDA    $0A
        SEC
        SBC    #$01
        JSR    L0606        ; previous phase
        PLA
        CLC
        ADC    #$01        ; RAM pointer++
        DEC    $04
        BNE    L05C4

        PHA
        LDA    $0A
        CLC
        ADC    #$03        ; next 1.5 track
        JSR    L0603        ; move...
        PLA
        RTS

```

```

      HEX      00000000000000000000000000000000

*
* the pseudo RWTS
*

L0600      JMP      L0609      ; Read
L0603      JMP      L065D      ; Move
L0606      JMP      L06D6      ; Change tempo

*
* Read...
*

L0609      STX      $3E      ; number of pages
          STA      $3A      ; RAM pointer
L060D      LDY      $3E      ; of boot stage 2
          STX      $40
          LDY      #$00      ; please refer to it
          LDA      $3A
          STY      $3C
          STA      $3D
          LDY      $08

L061B      JSR      L0657
L061E      CMP      $00
          BNE      L061B
          JSR      L0657
L0625      CMP      $01
          BNE      L061E
          JSR      L0657
          CMP      $02
          BNE      L0625
L0630      LDA      $C08C,X
          BPL      L0630
          ROL
          STA      $3F
L0638      LDA      $C08C,X
          BPL      L0638
          AND      $3F
          STA      ($3C),Y
          INY
          BNE      L0630
          ASL      KBD
L0647      LDA      $C08C,X
          BPL      L0647
          CMP      $03
          BNE      L060D
          INC      $3D
          DEC      $40
          BNE      L0630
          RTS

L0657      LDA      $C08C,X
          BPL      L0657
          RTS

*
* Move arm...
*

L065D      STA      $41
          CMP      $0A
          BEQ      L06B2
          LDA      #$00
          STA      $26
L0667      LDA      $0A
          STA      $27
          SEC
          SBC      $41
          BEQ      L06A1
          BCS      L0678
          EOR      #$FF
          INC      $0A
          BCC      L067C
L0678      ADC      #$FE
          DEC      $0A
L067C      CMP      $26
          BCC      L0682
          LDA      $26
L0682      CMP      #$0C
          BCS      L0687
          TAY
L0687      SEC
          JSR      L06A5
          LDA      L06BE,Y
          JSR      L06B3
          LDA      $27
          CLC
          JSR      L06A7
          LDA      L06CA,Y
          JSR      L06B3
          INC      $26
          BNE      L0667
L06A1      JSR      L06B3
          CLC
L06A5      LDA      $0A
L06A7      AND      #$03
          ROL
          ORA      $08
          TAX
          LDA      $C080,X
          LDY      $08
L06B2      RTS

L06B3      LDY      #$13
L06B5      DEX
          BNE      L06B5
          SEC
          SBC      #$01
          BNE      L06B3
          RTS

L06BE      HEX      01302824201E1D1C1C1C1C1C
L06CA      HEX      702C26221F1E1D1C1C1C1C1C

*
* Change tempo...
*

```

```

L06D6      LDX      #$0D
           STX      L06B3+1
           JSR      L065D
           LDA      #$13
           STA      L06B3+1
           RTS

           HEX      00000000000000000000000000000000
           HEX      00000000

*
* A marker table
*

L06F8      HEX      96979A9B9D9E9FCB

*
* The magnificent reboot code
*

L0700      LDA      #$D2      ; Q
           DB        $2C
L0703      LDA      #$D0      ; P
           DB        $2C
L0706      LDA      #$CC      ; L
           DB        $2C
L0709      LDA      #$A1      ; a
           PHA
           JSR      $02E0      ; =$07E0
           JSR      INIT
           JSR      HOME
           JSR      SETNORM
           PLA
           STA      L0400

L071C      LDY      #$00      ; Clear RAM
           TYA
L071F      STA      $BF00,Y
           INY
           BNE      L071F
           DEC      $0221
           LDA      $0221
           TAX
           BIT      SPKR
           NOP
           NOP
           NOP
           CMP      #$08
           BCS      L071C
           STA      SOTTEV+1
           STA      PWREDUP

           LDA      $02FF      ; Reboot
           LSR
           LSR
           LSR
           LSR
           ORA      #$C0
           SBC      #$00
           PHA
           LDA      #$FF
           PHA
           RTS

           HEX      00000000000000000000000000000000
           HEX      00000000000000000000000000000000
           HEX      00000000000000000000000000000000
           HEX      00000000000000000000000000000000
           HEX      00000000000000000000000000000000
           HEX      00000000000000000000000000000000
           HEX      00000000000000000000000000000000
           HEX      00000000000000000000000000000000
           HEX      00000000000000000000000000000000
           HEX      00000000

L07E0      LDA      ROMIN2      ; Goodbye wildcards
           LDA      ROMIN2
           LDY      #$00
           LDA      #$D0
           STY      $00
           STA      $01
L07EE      LDA      ($00),Y
           STA      ($00),Y
           INY
           BNE      L07EE
           INC      $01
           BNE      L07EE
           LDA      RDBANK2
           RTS

           DB        $00
           DB        $00
           DB        $00

```

Revenir en haut de page   

toinet
Site Admin

Posté le: Lun 03 Nov 2008, 8:48 Sujet du message:



Inscrit le: 15 Juin 2007
Messages: 1154
Localisation: Le Chesnay,
France

And now the final boot stage from track \$21 loaded from \$0400..\$04FF. It inits some game values and jumps to the program at \$6000.

Code:

```

*
* Lode Runner
* (c) 1983, Broderbund
*
* (k) 2008, LoGo
*

           org      $0400
           lst      off
           mx       %11

*

```


[illegible]

L04FE	DB	\$64
L04FF	DB	\$76

```
*
* Lode Runner
* (c) 1983, Broderbund
```

```

*
* (k) 2008, LoGo
*
        org    $2000
        lst    off
        mx      %l1

*
* Equates
*
SOFTEV    EQU    $03F2
PWREDUP    EQU    $03F4
KBD        EQU    $C000
KBDSTROBE EQU    $C010
SPKR       EQU    $C030
RDBANK2    EQU    $C080
ROMIN2     EQU    $C081
LCBANK2    EQU    $C083
INIT       EQU    $FB2F
BELL1_2    EQU    $FBE2
HOME       EQU    $FC58
WAIT       EQU    $FCA8
SETNORM    EQU    $FE84
OLDRST     EQU    $FF59
RESETV     EQU    $FFFC
IRQV       EQU    $FFFE

*
* Boot code stage 3
*
        lda    #0          ; we are on track 0
        sta    $0a

        ldx    #$60        ; slot*16
        stx    $2B
        stx    $08
        nop
        nop
        nop
        lda    #$1A        ; track $0D
        jsr    L0603        ; move
        nop
        nop
        jsr    L0550        ; read all
        nop
        nop

*
* Final read
*
        lda    #$DD        ; 1st marker
        sta    $00
        lda    #$F5        ; 2nd marker
        sta    $01
        lda    #$D5        ; 3rd marker
        sta    $02
        lda    #$D4        ; 4th marker
        sta    $03
        nop
        nop
        lda    #$42        ; track $21
        jsr    L0603        ; move

        lda    #$24        ; read at $2400 instead of $0400
        ldx    #$01        ; one sector
        jsr    L0600        ; read
        rts

*
* Main read loop
*
L0550    LDY    #$00        ; loop...
        STY    $05
L0554    STY    $06
        LDA    L0570,Y      ; RAM pointer
        BEQ    L0564
        JSR    L05C0        ; read data
        LDY    $06
        INY
        BNE    L0554        ; ...loop
L0564    RTS

*
* Where to load data
*
* $0F00..$1EFF
* $6000..$BFFF

L0570    HEX    0F1760687078808898A0A8B0B8000000

*
*
*
L0580    PHA
        LDA    $05          ; loop index
        AND    #$07
        TAY
        LDA    L06F8,Y      ; wonderful markers list
        STA    $00          ; first marker
        LDA    $05
        LSR
        ORA    #$AA
        STA    $01          ; second marker
        LDA    $05
        ORA    #$AA
        STA    $02          ; third marker
        PLA
        INC    $05
        LDX    #$01        ; read one sector
        JMP    L0600        ; read

*

```

```

* Read $0800 bytes of data
*
* Two adjacent blocks in memory
* are located on half tracks...
*

L05C0    LDY    #$04          ; loop 4
         STY    $04

L05C4    PHA
         LDX    #$01          ; RAM pointer
         JSR    L0580         ; unuseful
         LDA    $0A          ; read sector
         CLC
         ADC    #$01          ; start with $00
         JSR    L0606         ; next phase
         PLA
         CLC
         ADC    #$01          ; RAM pointer++
         PHA
         JSR    L0580         ; read sector
         LDA    $0A
         SEC
         SBC    #$01          ; previous phase
         JSR    L0606
         PLA
         CLC
         ADC    #$01          ; RAM pointer++
L0603    DEC    $04
         BNE    L05C4

         PHA
         LDA    $0A
         CLC
         ADC    #$03          ; next 1.5 track
         JSR    L0603         ; move...
         PLA
         RTS

         ds     \

*
* the pseudo RWTS
*

L0600    JMP    L0609         ; Read
L0603    JMP    L065D         ; Move
L0606    JMP    L06D6         ; Change tempo

*
* Read...
*

L0609    STX    $3E          ; number of pages
         STA    $3A          ; RAM pointer
L060D    LDX    $3E          ; of boot stage 2
         STX    $40
         LDY    #$00         ; please refer to it
         LDA    $3A
         STY    $3C
         STA    $3D
         LDX    $08

L061B    JSR    L0657
L061E    CMP    $00
         BNE    L061B
         JSR    L0657
L0625    CMP    $01
         BNE    L061E
         JSR    L0657
         CMP    $02
         BNE    L0625
L0630    LDA    $C08C,X
         BPL    L0630
         ROL
         STA    $3F
L0638    LDA    $C08C,X
         BPL    L0638
         AND    $3F
         STA    ($3C),Y
         INY
         BNE    L0630

         ASL    KBD

L0647    LDA    $C08C,X
         BPL    L0647
         CMP    $03
         nop
         nop                ; we dislike
                             ; checksums ;-)
* BNE L060D

         INC    $3D
         DEC    $40
         BNE    L0630
         RTS

L0657    LDA    $C08C,X
         BPL    L0657
         RTS

*
* Move arm...
*

L065D    STA    $41
         CMP    $0A
         BEQ    L06B2
         LDA    #$00
         STA    $26
L0667    LDA    $0A
         STA    $27
         SEC
         SBC    $41
         BEQ    L06A1
         BCS    L0678
         EOR    #$FF
         INC    $0A
         BCC    L067C

```

```

L0678    ADC    #$FE
          DEC    $0A
L067C    CMP    $26
          BCC    L0682
          LDA    $26
L0682    CMP    #$0C
          BCS    L0687
          TAY
L0687    SEC
          JSR    L06A5
          LDA    L06BE,Y
          JSR    L06B3
          LDA    $27
          CLC
          JSR    L06A7
          LDA    L06CA,Y
          JSR    L06B3
          INC    $26
          BNE    L0667
L06A1    JSR    L06B3
          CLC
L06A5    LDA    $0A
L06A7    AND    #$03
          ROL
          ORA    $08
          TAX
          LDA    $C080,X
          LDX    $08
L06B2    RTS

L06B3    LDX    #$13
L06B5    DEX
          BNE    L06B5
          SEC
          SBC    #$01
          BNE    L06B3
          RTS

L06BE    HEX    01302824201E1D1C1C1C1C1C
L06CA    HEX    702C26221F1E1D1C1C1C1C1C

*
* Change tempo...
*

L06D6    LDX    #$0D
          STX    L06B3+1
          JSR    L065D
          LDA    #$13
          STA    L06B3+1
          RTS

*
* A marker table
*

L06F8    HEX    96979A9B9D9E9FCB

ds      \

```

Revenir en haut de page   

toinet
Site Admin

Posté le: Lun 03 Nov 2008, 8:54 Sujet du message:



Inscrit le: 15 Juin 2007
Messages: 1154
Localisation: Le Chesnay,
France

And the final code which should have been put before the previous message is the rewritten code to get the original \$0400..\$07FF code from track \$00:

Code:

```

*
* Lode Runner
* (c) 1983, Broderbund
*
* (k) 2008, LoGo
*

        org    $0800
        lst    off
        mx     %11

*
* Boot code stage 2 modified
*

        ldx    #$60
        stx    $08
        lda    $c089,x

        LDY    #$D4
        STX    $00        ; #$D4
        INX
        STX    $01        ; #$D5
        INX
        STX    $02        ; #$D6
        INX
        STX    $03        ; #$D7

        LDX    #$04        ; high pointer
        LDA    #$14        ; number of pages
        jsr    L0060

        ldx    $08
        lda    $c088,x
        rts

* Remember $08 contains slot*16

L0060    STX    $3E        ; number of pages
        STA    $3A        ; RAM pointer

L0064    LDX    $3E
        STX    $40
        LDY    #$00
        LDA    $3A
        STY    $3C

```

```

                STA    $3D
                LDX    $08      ; slot*16
L0072          JSR    L00AE
L0075          CMP    $00      ; #$D4
                BNE    L0072
                JSR    L00AE
L007C          CMP    $01      ; #$D5
                BNE    L0075
                JSR    L00AE
                CMP    $02      ; #$D6
                BNE    L007C

L0087          LDA    $C08C,X   ; 4*4 coding ;- )
                BPL    L0087
                ROL
                STA    $3F
L008F          LDA    $C08C,X
                BPL    L008F
                AND    $3F
                STA    ($3C),Y
                INY
                BNE    L0087

L009E          LDA    $C08C,X
                BPL    L009E
                CMP    $03      ; #$D7
                BNE    L0064
                INC    $3D      ; next high ram pointer
                DEC    $40      ; decrement number of pages
                BNE    L0087
                RTS

L00AE          LDA    $C08C,X
                BPL    L00AE
                RTS

```

Revenir en haut de page   

toinet
Site Admin

Posté le: Lun 03 Nov 2008, 10:46 Sujet du message:

 citer

Inscrit le: 15 Juin 2007
Messages: 1154
Localisation: Le Chesnay,
France

The pre-final stage is to reconstruct the disk with the data and program recorded onto different tracks.

The following table lists how it is done:

- T00/S00: boot 1 code (\$0800..\$08FF)
- T00/S01..S04: new boot 2 code (\$0400..\$07FF)
- T03-T0C: level data
- T0D: program data (\$0F00..\$1EFF)
- T0E..T13: program code (\$6000..\$BFFF)

For vintage and historical digging, please find the original boot codes there:

- T01/S01..S04: original boot 2 code (\$0400..\$07FF)
- T21/S00: original boot 3 code (\$0400..\$04FF)

Please use our own program to copy the program and its data at the right places.

Antoine
11/2008

Revenir en haut de page   

vladitx

Posté le: Mar 04 Nov 2008, 21:00 Sujet du message:

 citer

Inscrit le: 19 Déc 2007
Messages: 22

Toto, you rule. 😊

Revenir en haut de page  

toinet
Site Admin

Posté le: Mar 04 Nov 2008, 21:49 Sujet du message:

 citer

Inscrit le: 15 Juin 2007
Messages: 1154
Localisation: Le Chesnay,
France

And now... the rewritten Boot 1 routine as well as my new "fast" track loading, similar to the Electronic Arts' one.

As previously written, I always want to make minimal changes to the programs, therefore I load five sectors directly with the ROM routine, copy data to the right places and load the program. I apply the same vector routines change and other inits that are not really useful there but that is similar to the original program 😊

And for curious people, download the disk at the following place: <http://www.brutal-deluxe.fr/crack/LODERUNNER.DSK>

Code:

```

*
* Lode Runner
* (c) 1983, Broderbund
*
* (k) 2008, LoGo
*
                org    $0800
                lst    off
                mx     %11

*
* Equates
*
dpSLOT    =    $08
dpCPHASE  =    $0A
dpLOADED  =    $F9
dpRAM     =    $FA
dpTRACK   =    $FB
dpNBTRACKS = $FC

```

```

dpNIBBLE = $FD
dpBUFFER = $FE

L0200 = $0200
L0203 = $0203
L0300 = $0300
L0356 = $0356
L0400 = $0400
L0600 = $0600
moveARM = $0603
L6000 = $6000 ; The program

SOFTEV EQU $03F2
PWREDUP EQU $03F4

TXTCLR EQU $C050
MIXCLR EQU $C052
TXTPAGE1 EQU $C054
HIRES EQU $C057

*
* Boot code
*

L0800 DB $05

L0801 LDY #$00 ; clear
      LDA #$20 ; HGR & HGR2 pages
      LDX #$40
      STY $00
      STA $01
      TYA
jlp STA ($00),Y
   INY
   BNE jlp
   INC $01
   DEX
   BNE jlp

      BIT MIXCLR ; HGR mode on
      BIT HIRES
      BIT TXTPAGE1
      BIT TXTCLR

*-----

jlp LDY #$00 ; copy all data
   LDA M0600,Y
   STA L0600,Y
   LDA M0200,Y
   STA L0200,Y
   LDA M0400,Y
   STA L0400,Y
   INY
   BNE jlp

*-----

      LDA #$00 ; current track
      STA dpCPHASE

      LDX $2B ; slot*16
      STX dpSLOT
      STX $02FF

*-----

      lda #$0D ; Track
      ldx #$0F ; RAM
      ldy #$01 ; Nb tracks to read
      jsr readALL

      lda #$0E ; Track
      ldx #$60 ; RAM
      ldy #$06 ; Nb tracks to read
      jsr readALL

*-----

      LDA #>L0200 ; reset vector
      STY SOFTEV
      STA SOFTEV+1
      EOR #$A5
      STA PWREDUP

      LDY #<L0203 ; redirect all vectors
      LDA #>L0203
      STY $36
      STA $37
      STY $38
      STA $39
      STY $03F0
      STA $03F1

*-----

      LDX $02FF ; slot*16
      STX $B7E9
      STX $B7F7

      TXA
      LSR
      LSR
      LSR
      LSR
      TAX
      LDA #$18 ; Phase (Track = #$0C)
      STA $0478,X
      jsr $0603 ; move arm

      LDA #$06
      STA $8C
      LDA #$FF
      STA $99
      LDA #$CA
      STA $95
      LDA #$4C
      STA $23

```

```

        LDA    #$50      ; $8E50
        STA    $36
        LDA    #$8E
        STA    $37
        LDA    #$B5      ; $B7B5
        STA    $38
        LDA    #$B7
        STA    $39

        lda    #$64      ; The final
        sta    $00
        lda    #$76      ; checksums
        sta    $01

        ldx    $02FF
        LDA    $C088,X
        JMP    L6000

        ds     \

*-----
* A= Track
* X= RAM
* Y= Nb of tracks to read
*

readALL  sta    dpTRACK
        txa
        sta    dpRAM
        sty    dpNBTRACKS

        lda    #0
        sta    dpBUFFER

*

readMAIN lda    dpTRACK    ; move arm
        asl
        jsr    moveARM

        ldx    #$0F      ; number of sectors
        stx    dpLOADED  ; to read per track

        ldy    dpRAM      ; prepare table
jlp      tya              ; of RAM pointers
        clc
        adc    tblINTER,x
        sta    tblMEMORY,x ; where to load
        dex
        bpl    jlp

*
* Read header
*

readHEADER CLC
readDATA  PHP
        LDY    dpSLOT
read1     LDA    $C08C,X
        BPL    *-3
read2     EOR    #$D5
        BNE    read1
        LDA    $C08C,X
        BPL    *-3
        CMP    #$AA
        BNE    read2
        LDA    $C08C,X
        BPL    *-3
        CMP    #$96
        BEQ    doHEADER
        PLP
        BCC    readHEADER
        EOR    #$AD
        BEQ    doDATA
        BNE    readHEADER

* Read header

doHEADER LDY    #4          ; volume/volume
jlp      LDA    $C08C,X      ; track/track
        BPL    *-3
        DEY
        BNE    jlp          ; we skip them ;- )

        LDA    $C08C,X      ; we want sector
        BPL    *-3
        ROL
        STA    dpNIBBLE
        LDA    $C08C,X
        BPL    *-3
        AND    dpNIBBLE
        AND    #$0f         ; $00..$0F

        PLP                  ; restore status

        TAX
        LDA    tblMEMORY,X
        BEQ    readHEADER ; already been read
        STA    dpBUFFER+1
        LDA    #0           ; sector has been read
        STA    tblMEMORY,X
        SEC
        BCS    readDATA    ; read data

* Read data

doDATA   LDY    #$56
jlp      STY    dpNIBBLE
        LDY    $C08C,X
        BPL    *-3
        EOR    L0356-$80,Y
        LDY    dpNIBBLE
        STA    L0300,Y
        BNE    jlp

jlp      STY    dpNIBBLE
        LDY    $C08C,X

```


[illegible]

That game is fantastic...

Antoine
Nov. 2008

Revenir en haut de page

profil tmp www

Montrer les messages depuis: Tous les messages Le plus ancien en premier Aller

nouveau répondre

Index du Forum -> PROTECTION MALEFIQUE

Toutes les heures sont au format GMT

Page 1 sur 1

Sauter vers: PROTECTION MALEFIQUE Aller

Vous **ne pouvez pas** poster de nouveaux sujets dans ce forum
 Vous **ne pouvez pas** répondre aux sujets dans ce forum
 Vous **ne pouvez pas** éditer vos messages dans ce forum
 Vous **ne pouvez pas** supprimer vos messages dans ce forum
 Vous **ne pouvez pas** voter dans les sondages de ce forum

Powered by phpBB © 2001, 2005 phpBB Group
 Traduction par : phpBB-fr.com