Commodore 1541 ROM genealogy

Author: Audioniek

Revision: July 15, 2025

Over time quite a lot of 1541 ROM revisions have taken place. This text analyzes them in a chronological order. As the 1541 was in fact a development of the V(I)C-1540[1] and that drive in turn was in effect a 2031 with a serial bus interface instead of an IEEE-488 connection, the sequence starts with the 2031, the father of all single drive Commodore floppy drives with one CPU.

Checksum bytes

At the time the various ROMs were produced, Commodore used a mechanism based on a checksum to verify the code/ROM integrity. In all 1541 variants, these checksums are checked upon power up, and the drive will not start and show a flashing LED in case a checksum is found to be wrong.

The algorithm to determine the checksum is simple. All the bytes in the ROM (including the checksum byte) are added together, with a possible carry resulting from the previous addition ignored. The resulting checksum should be equal to the high byte address of the first byte in the ROM; for example an 8k byte ROM intended for the addresses $C000-$DFFF should have a checksum of $C0.

The checksum byte for the ROM at $C000-$DFFF is located at $C000; that for the ROM at $E000-$FFFF at $FEE6.

Although the checksum bytes differ for each ROM revision, they are not listed as a difference in the sections below.

Unused ROM space

Commodore was in the habit of filling unused areas in ROMs with $AA bytes (disassembles as TAX). Filling unused space with $00 or $FF as is often done, is less advisable with the 6502 CPU, as $00 constitutes a BRK instruction, and $FF is an illegal opcode. $AA at least does not cause a crash or activates stack operations. A single $AA byte by itself surrounded by seemingly random other bytes is most likely valid code.

Code changes

Commodore implemented changes in code by adding so called patches instead of inserting them inline and reassembling the (source) code. This technique is often done by using hand assembly, although the changes were incorporated in the assembler source code. The most probable reason is compatibility; this way the calling addresses of all routines remained unchanged between ROM revisions.

In the 1541, two areas of 254 bytes each have been reserved and used as patch areas; each 8k byte ROM having one area. The areas are $C002-$C0FF and $FEE6-$FFE5. The area $C002-$C0FF remained unused until the second ROM revision for the 1541B[2] drive appeared; by then 16 kbyte ROMs were in use. The area $FEE6-$FFE5 was used for patches in all ROM revisions, even in the 2031.

[1] In Germany the VIC-1540 was called the VC-1540 to avoid an unfortunate pun. Hence the spelling V(I)C-1540. The same goes for the V(I)C-20 computer.

[2] Commodore's source code and technical documentation consistently calls this model the 1541B because for Commodore's engineering department, the model was the successor of the 1541A(2) with the short board. However the user manual refers to this model as the 1541C, probably because the matching computer is called 64C. Most sources use the designation 1541C. The type plate on the device simply states 1541 without a letter.

Hardware compatibility

The various revisions of the 1541 hardware are compatible with each other but for one feature, that is unique to the 1541B model, the last version with a built-in power supply. The very small main board in this model supports a track zero sensor that can be defeated by closing a solder blob. The track zero sensor is connected to a VIA pin, that is grounded in the later model 1541-II but unconnected in the models prior to the 1541B. The net result of all this that all ROM revisions can be used in a 1541B (this will render the track zero sensor inoperative if present) and a 1541-II, but the ROMs developed for the 1541B (251968-01 and 251968-02) cannot be used in the models prior to it (unless the VIA pin involved is connected to ground). The ROMs for the 1541B model can be identified by the fact that they bump the drive mechanism on power up.

The 2031 drive

The 2031 is in effect a single drive 4040. Commodore simplified the 4040 by omitting one drive mechanism, halving the amount of RAM and leaving out the twin CPU arrangement (one CPU as disk/mechanism controller and one for the DOS and bus interface) to reduce cost. This led to a new piece of code called the Low Cost Controller or LCC, and a new DOS version number: V2.6. The former DOS/interface CPU now does all the work and is in effect running a multitasking environment through the clever use of interrupts and the overflow flag. The 2031 writes the exact same disk format as the 4040 and has the same DOS bugs (the save-with-replace bug among them). By the time the 1541 appeared, Commodore created the 2031 LP that uses the same cabinet style as the 1541 including the colour and the low profile drive mechanism. For the 2031(LP) only one ROM revision exists, consisting of two 8 kbyte ROMs for a total of 16 kbyte (ROMs 901484-03 and 901484-05).

The V(I)C-1540 drive

The V(I)C-20 computer introduced the serial bus, replacing the IEEE-488 interface of the computer models before it. The bus was cheaper to implement but also was quite a bit slower. Because of a bug in the 6522 VIA's shift register discovered at the time of development of the serial bus, the serial protocol had to be implemented completely in software, making it even slower. The V(I)C-1540 writes the exact same disk format as the 4040 and has the same bugs (the save-with-replace bug among them). For the V(IC)-1540 only one ROM revision exists, consisting of two 8 kbyte ROMs for a total of 16 kbyte (ROMs 325302-01 and 325303-01). The ROM 325302-01 would be used throughout a larger number of ROM revisions for the 1541 as well, as the code in it was not changed until the 1541B appeared and the two 8 kbyte ROMs were replaced by one 16 kbyte device.

Compared to the 2031 most code is different in that it is located on different locations. For the DOS part however, the code is functionally the same; likely the reason Commodore did not change the DOS version number. Part of the code differences are of course caused by the difference in interface. For these reasons, a direct comparison on the byte level is not useful or instructive. If you want more info, compare the source code for the two models that can be found on the internet.

1541, 901229-01 ROM

This is the first ROM revision for the 1541. Commodore's plan was to use the 1540 for the 64 computer as well, but the internal timing of 64 made its serial bus slower than that of the V(I)C-20; the 64 could not keep up with the floppy drive during read operations. For this reason the serial bus of the drive was slowed down, giving rise to a replacement of the ROM 325303-01 by ROM 901229-01. To keep the two versions apart the model number of the drive was changed to 1541.

For reasons unknown to the author, the ROM had additional changes introducing a bug/feature. In the format routine of the LCC, a superfluous INX instruction was added (or not replaced with a NOP), causing the data part of a newly formatted sector to contain one $4D byte and 255 $01 bytes, as opposed to 256 zero bytes previously. Apart from this difference, the 1541 with ROM 901229-01 writes the same format as the previous drive models 4040, 2031(LP) and V(I)C-1540 and the disks are therefore read and write compatible between these models.

All changes in ROM code were limited to the ROM for the area $E000-$FFFF (ROMs 325303-01 and 901229-01 respectively), the ROM 325302-01 remained unchanged (that ROM remained unchanged until Commodore switched to a single 23128 ROM for the 1541B; all upcoming changes for the original 1541 model were limited to the ROM 901229-0X).

Code differences compared to the V(I)C-1540

1540 ROM: 325303-01
1541 ROM: 901229-01

Difference #1: Power up string at $E5B6.

Note that the V(I)C-1540 identifies itself as V170, not 1540.

```
E5B6:  73       .BY    $73          ;error number
E5B7:  C3       .BY    ‘C’|$80
E5B8:  42       .BY    ‘B’
E5B9:  4D       .BY    ‘M’
E5BA:  20       .BY    ‘ ’
E5BB:  44       .BY    ‘D’
E5BC:  4F       .BY    ‘O’
E5BD:  53       .BY    ‘S’
E5BE:  20       .BY    ‘ ’
E5BF:  56       .BY    ‘V’
E5C0:  32       .BY    ‘2’
E5C1:  2E       .BY    ‘.’
E5C2:  36       .BY    ‘6’
E5C3:  20       .BY    ‘ ’
E5C4:  56       .BY    ‘V’
E5C5:  31       .BY    ‘1’
E5C6:  37       .BY    ‘7’
E5C7:  B0       .BY    ‘0’|$80
```

```
E5B6:  73       .BY    $73          ;error number
E5B7:  C3       .BY    ‘C’|$80
E5B8:  42       .BY    ‘B’
E5B9:  4D       .BY    ‘M’
E5BA:  20       .BY    ‘ ’
E5BB:  44       .BY    ‘D’
E5BC:  4F       .BY    ‘O’
E5BD:  53       .BY    ‘S’
E5BE:  20       .BY    ‘ ’
E5BF:  56       .BY    ‘V’
E5C0:  32       .BY    ‘2’
E5C1:  2E       .BY    ‘.’
E5C2:  36       .BY    ‘6’
E5C3:  20       .BY    ‘ ’
E5C4:  31       .BY    ‘1’
E5C5:  35       .BY    ‘5’
E5C6:  34       .BY    ‘4’
E5C7:  B1       .BY    ‘1’|$80
```

Difference #2: Slowing the serial bus for the Commodore 64.

The Commodore 64 was too slow to keep up with the V(I)C-1540. Commodore solved this by making the serial bus go slower. The slowing down has been made switchable through the commands UI- and UI+ and defaults to the slow Commodore 64 speed.

Note the four NOPs in the V(I)C-1540 ROM; what was there originally?

```
E973   209CE9   ISRHI  JSR DATHI
E976:  20B7E9   ISRCLK JSR CLKHI
E979:  EA              NOP
E97A:  EA              NOP
E97B:  EA              NOP
E97C:  EA              NOP
E97D:  20AEE9          JSR CLKLOW
E980:  209CE9          JSR DATHI
E983:  C698            DEC CONT    ;count bits
```

```
FEF3:  AA              TAX
FEF4:  AA              TAX
FEF5:  AA              TAX
FEF6:  AA              TAX
FEF7:  AA              TAX
FEF8:  AA              TAX
FEF9:  AA              TAX
FEFA:  AA              TAX
```

```
FEFB:  AA              TAX
FEFC:  AA              TAX
FEFD:  AA              TAX
FEFE:  AA              TAX
FEFF:  AA              TAX
FF00:  AA              TAX
```

```
E973:  209CE9   ISRHI  JSR DATHI
E976:  20B7E9   ISRCLK JSR CLKHI
E979:  A523            LDA DRVTRK+1 ;! added: get flag
E97B:  D003            BNE ISR03    ;! added: if zero


E97D:  20F3FE          JSR SLOWD    ;! added: wait a bit
E980:  20FBFE   ISR03  JSR CLKDAT   ;! changed: CLK lo/DATA hi
E983:  C698            DEC COUNT    ;count bits
```

```
                ;****************************************
                ;* Patch to slow down serial bus        *
                ;****************************************
FEF3:  8A       SLOWD  TXA          ;save X
FEF4:  A205            LDX #5       ;get loop count

FEF6:  CA       SLOWE  DEX          ;wait a bit
FEF7:  D0 FD           BNE SLOWE    ;if not done, loop

FEF9:  AA              TAX          ;else restore X
FEFA:  60              RTS          ;and exit
```

```
                ;****************************************
                ;* Patch: set data hi when setting      *
                ;* clock low                            *
                ;****************************************
FEFB:  20AEE9   CLKDAT JSR CLKLOW   ;set CLK lo


FEFE:  4C9CE9          JMP DATHI    ;and DATA hi
```

Difference #3: The data block contents of a freshly formatted sector.

Why this phenomenon was introduced or somebody forgot to incorporate an already known fix is unclear. What is also unclear is why the bug was never fixed throughout the lifespan of the 1541. The bug provides an easy way to determine whether a disk was formatted in a 1541 or other drive though.

```
FC7A:  EE2806          INC SECT    ;increment sector number
FC7D:  AD2806          LDA SECT    ;get it
FC80:  C543            CMP SECTR   ;if whole track done,
FC82:  90BB            BCC MAK10   ;proceed
FC84:  98              TYA         ;else get block size in A
FC85:  48              PHA         ;save it
FC86:  EA              NOP         ;X=0 (probably a fix)

FC87:  8A              TXA         ;A=0
FC88:  9D0005   CRTDAT STA BUFF2,X ;store A in data block
FC8B:  E8              INX         ;point to next
FC8C:  D0FA            BNE CNTDAT  ;loop until block filled
```

```
FC7A:  EE2806          INC SECT    ;increment sector number
FC7D:  AD2806          LDA SECT    ;get it
FC80:  C543            CMP SECTR   ;if at maximum sector#,
FC82:  90BB            BCC MAK10   ;proceed
FC84:  98              TYA         ;else get block size in A
FC85:  48              PHA         ;save it
FC86:  E8              INX         ;! BUG: X should be 0!
                                   ;(is one now)
                ;****************************************
                ;* BUG: because X holds drive# (zero at *
                ;* this point) it is incremented to     *
                ;* one.                                 *
                ;*                                      *
                ;* This has the following effect:       *
                ;*                                      *
                ;* - The first byte of the data block   *
                ;*   is skipped (and left at $4D);      *
                ;* - Because X is 1, the remaining data *
                ;*   bytes in the block are set to $01. *
                ;*                                      *
                ;* The 2031/4040/1540 fill all 256 data *
                ;* bytes with the value zero.           *
                ;*                                      *
                ;****************************************
FC87:  8A              TXA         ;move X to A (A=1)
FC88:  9D0005   CNTDAT STA BUFF2,X ;fill data part0
FC8B:  E8              INX         ;point to next
FC8C:  D0FA            BNE CNTDAT  ;loop until block filled
```

Difference #4: Make the serial bus speed switchable.

The disk command UI (or U9) unleashes the same actions as an actual NMI and its associated processing. The NMI vector was changed and points to a new routine that tests for plus and minus characters as third command character. If not found, the original NMI code is executed.

```
FF01:  AA              TAX
FF02:  AA              TAX
FF03:  AA              TAX
FF04:  AA              TAX
FF05:  AA              TAX
FF06:  AA              TAX
FF07:  AA              TAX
FF08:  AA              TAX
FF09:  AA              TAX
FF0A:  AA              TAX
FF0B:  AA              TAX
FF0C:  AA              TAX
FF0D:  AA              TAX
FF0E:  AA              TAX
FF0F:  AA              TAX

               ;****************************************
               ;* 6502 vectors                         *
               ;****************************************
FFFA:  E7FE            .WO NMI     ;NMI
FFFC:  A0EA            .WO DSKINT  ;Reset
FFFE:  67FE            .WO IRQ     ;IRQ
```

```
               ;****************************************
               ;* NMI entry.                           *
               ;*                                      *
               ;* Check for UI+ and UI- to set serial  *
               ;* bus speed.                           *
               ;*                                      *
               ;****************************************
FF01:  AD0202  NNMI   LDA CMDBUF+2  ;get 3rd command char.


FF04:  C92D           CMP #'-'      ;if a minus

FF06:  F005           BEQ NNMI10    ;set $23 to non zero

FF08:  38             SEC           ;else prepare for subtract
FF09:  E92B           SBC #'+'      ;if original was a plus

FF0B:  D0DA           BNE NMI       ;zero $23

FF0D:  8523    NNMI10 STA DRVTRK+1

FF0F:  60             RTS           ;and exit

               ;****************************************
               ;* 6502 vectors                         *
               ;****************************************
FFFA:  01FF           .WO NNMI      ;! changed
FFFC:  A0EA           .WO DSKINT    ;Reset
FFFE:  67FE           .WO IRQ       ;IRQ
```

This concludes the differences between the V(I)C-1540 and the 1541 901229-01 ROMs.

1541, 901229-02 ROM

The second ROM revision for the 1541 only brought one serial protocol change.

All changes in ROM code were limited to the ROM for the area $E000-$FFFF (901229-02). The changes that occurred between ROMs 325303-01 and 901229-01 were retained unchanged.

Code differences compared to the ROM 901229-01

ROMs involved: 901229-01 and 901229-02

Difference #1: Change reading a data bit.

The change swaps the order of events while reading the next bit of a byte off the serial bus: the reading of the bit and waiting for the CLK line to go low have been reversed. The NOPs are there because one read of VIA port B is now superfluous. The comments in the source code refer to a removal of a speed up for the European V(I)C-20, which seems odd, as the new code is faster (measured in clock cycles).

```
E9F2:  20A5E9   ACP00B JSR DATLOW  ;set DATA low
E9F5:  A20A            LDX #10     ;wait
E9F7:  CA       ACP02  DEX         ;a
E9F8:  D0FD            BNE ACP02   ;bit
E9FA:  209CE9          JSR DATHI   ;set DATA high
E9FD:  2059EA   ACP02A JSR TSTATN  ;test for ATN
EA00:  20C0E9          JSR DEBNC   ;wait for CLK low
EA03:  2904            AND #$04    ;get CLK in
EA05:  F0F6            BEQ ACP02A  ;if hi, wait
EA07:  A900            LDA #$00    ;else
EA09:  85F8            STA EOIFLG  ;clear EOI flag
EA0B:  AD0018   ACP01  LDA PB      ;get bus lines
EA0E:  2904            AND #$04    ;get CLK in
EA10:  D0F9            BNE ACP01   ;if lo, wait
EA12:  AD0018          LDA PB      ;get bus lines
EA15:  4901            EOR #$01    ;invert DATA in
EA17:  4A              LSR A       ;get it in C


EA18:  6685            ROR DATA    ;rotate received bit in
EA1A:  2059EA   ACP03A JSR TSTATN
```

```
E9F2:  20A5E9   ACP00B JSR DATLOW  ;set DATA low
E9F5:  A20A            LDX #10     ;wait
E9F7:  CA       ACP02  DEX         ;a
E9F8:  D0FD            BNE ACP02   ;bit
E9FA:  209CE9          JSR DATHI   ;set DATA high
E9FD:  2059EA   ACP02A JSR TSTATN  ;test for ATN
EA00:  20C0E9          JSR DEBNC   ;wait for CLK low
EA03:  2904            AND #$04    ;get CLK in
EA05:  F0F6            BEQ ACP02A  ;if hi, wait
EA07:  A900            LDA #$00    ;else
EA09:  85F8            STA EOIFLG  ;clear EOI flag
EA0B:  AD0018   ACP01  LDA PB      ;get bus lines
EA0E:  4901            EOR #$01    ;! changed: toggle DATA in
EA10:  4A              LSR A       ;! changed: get bit in C
EA11:  2902            AND #$02    ;! changed: wait for CLK
EA13:  D0F6            BNE ACP01   ;! changed: hi
EA15:  EA              NOP         ;! changed: remove PAL
EA16:  EA              NOP         ;! changed: V(I)C-20
EA17:  EA              NOP         ;! changed: speedup
EA18:  6685            ROR DATA    ;rotate received bit in
EA1A:  2059EA   ACP03A JSR TSTATN
```

Difference #2: Initials of the programmer added.

At the end of the ROM the letters RSR were added, probably standing for the initials of the author of the difference mentioned above.

```
FFE1:  AA              TAX
FFE2:  AA              TAX
FFE3:  AA              TAX
FFE4:  AA              TAX
FFE15  AA              TAX
```

```
FFE1:  AA              TAX
FFE2:  52              .BY 'R'    ;! changed: patcher's
FFE3:  53              .BY 'S'    ;! changed: initials
FFE4:  52              .BY 'R'    ;! changed
FFE5:  AA              TAX
```

This concludes the differences between the 1541 901229-01 and 901229-02 ROMs.

1541, 901229-03 ROM

The third ROM revision for the 1541 only changed the value of gap1 in the sector headers.

All changes in ROM code were limited to the ROM for the area $E000-$FFFF (901229-03). The changes that occurred between ROMs 901229-01 and 901229-02 were retained unchanged.

Code differences compared to the ROM 901229-02

ROMs involved: 901229-02 and 901229-03

Difference #1: Change gap1.

The change makes gap1 (the number of disk bytes between two parts of a sector header) one disk byte (10 bits) longer. The change is required in two places: the code that (re)writes a sector starting at $F586 and the formatting code starting at $FCCD. In both cases only one constant is changed.

This change causes disks formatted with all previous ROMs (as well as the 2031(LP) and the 4040) to be no longer write compatible. Because the read routines simply wait during the passing of the gap for the SYNC marker to pass, reading is not affected: the new value for gap1 simply causes the remaining sector parts to pass the head later in time and within the time slot allowed.

```
F586:  208FF7   WRT10  JSR BINGCR  ;convert the data to GCR
F589:  2010F5          JSR SRCH    ;position the head
F58C:  A209            LDX #8      ;get gap1 length
F58E:  50FE     WRT20  BVC WRT20
F590:  B8              CLV
F591:  CA              DEX         ;count gap bytes
F592:  D0FA            BNE WRT20   ;and loop

FCCD:  D0F3            BNE WRTS20
FCCF:  A209            LDX #8      ;get gap1 length
FCD1:  50FE     WRTS30 BVC WRTS30
FCD3:  B8              CLV
FCD4:  A955            LDA #$55    ;get gap data
FCD6:  8D011C          STA DATA2   ;set it
FCD9:  CA              DEX         ;count gap bytes
FCDA:  D0F5            BNE WRTS30  ;and loop
                       ; write out data block
```

```
F586:  208FF7   WRT10  JSR BINGCR
F589:  2010F5          JSR SRCH
F58C:  A209            LDX #9      ;! changed: get gap1
F58E:  50FE     WRT20  BVC WRT20
F590:  B8              CLV
F591:  CA              DEX         ;count gap bytes
F592:  D0FA            BNE WRT20   ;and loop

FCCD:  D0F3            BNE WRTS20
FCCF:  A209            LDX #9      ;! changed: get gap1 count
FCD1:  50FE     WRTS30 BVC WRTS30
FCD3:  B8              CLV
FCD4:  A955            LDA #$55    ;get gap data
FCD6:  8D011C          STA DATA2   ;set it
FCD9:  CA              DEX         ;count gap bytes
FCDA:  D0F5            BNE WRTS30  ;and loop
                       ; write out data block
```

This concludes the differences between the 1541 901229-01 and 901229-02 ROMs.

1541, 901229-04 ROM

The fourth ROM revision for the 1541 was apparently only used for the internal drive mounted in the SX-64 portable computer. It improved power on behaviour, and improved handling an EOI condition.

All changes in ROM code were limited to the ROM for the area $E000-$FFFF (901229-03 and 901229-04). The changes that occurred between ROMs 901229-02 and 901229-03 were retained unchanged apart from difference #4 below.

Code differences compared to the ROM 901229-03

ROMs involved: 901229-03 and 901229-04

<u>Difference #1: Wait for all devices on the serial bus to idle before starting the EOI timer.</u>

```
E9CD:  2059EA   ACP00A JSR TSTATN
E9D0:  20C0E9          JSR DEBNC
E9D3:  2904            AND #$04    ;get state of CLK
E9D5:  D0F6            BNE ACP00A  ;wait for high
E9D7:  209CE9          JSR DATHI   ;then set DATA high
E9DA:  A901            LDA #$01    ;wait
E9DC:  8D0518          STA T1HC1   ;255 us


E9DF:  2059EA   ACP00  JSR TSTATN
```

```
FF20:  AA              TAX
FF21:  AA              TAX
FF22:  AA              TAX
FF23:  AA              TAX
FF24:  AA              TAX
FF25:  AA              TAX
FF26:  AA              TAX
FF27:  AA              TAX
FF28:  AA              TAX
FF29:  AA              TAX
FF2A:  AA              TAX
FF2B:  AA              TAX
FF2C:  AA              TAX
```

```
E9CD:  2059EA   ACP00A JSR TSTATN
E9D0:  20C0E9          JSR DEBNC
E9D3:  2904            AND #$04    ;get state of CLK
E9D5:  D0F6            BNE ACP00A  ;wait for high
E9D7:  209CE9          JSR DATHI   ;then set DATA high
E9DA:  A901            LDA #$01    ;wait
E9DC:  4C20FF          JMP PATCH6  ;! changed: wait for all
                                   ;devices to idle, start
                                   ;timer for 255 us
E9DF:  2059EA   ACP00  JSR TSTATN

Added:
               ;****************************************
               ;* PATCH6:                              *
               ;*                                      *
               ;* Fix EOI behaviour to wait for all    *
               ;* devices                              *
               ;*                                      *
               ;****************************************
FF20:  AD0018  PATCH6 LDA PB       ;get bus lines


FF23:  2901            AND #$01    ;get DATA in

FF25:  D0F9            BNE PATCH6  ;wait for it to go high
                                   ;(all devices idle)
FF27:  A901            LDA #$01    ;now

FF29:  8D0518          STA T1HC1   ;set timer to


FF2C:  4CDFE9          JMP ACP00   ;wait 255 us and return
```

Difference #2: On power on, immediately set CLK line hi.

This change immediately sets the bus CLK line to high on power up, to avoid blocking the serial bus until the whole drive initialization is completed.

```
                ;****************************************
                ;* Reset entry                          *
                ;****************************************
EAA0:  78       DSKINT SEI          ;no interrupts
EAA1:  D8              CLD          ;clear decimal mode
EAA2:  A2FF            LDX #$FF     ;direction is all outputs
EAA4:  8E0318          STX DDRA1    ;set it
EAA7:  E8              INX          ;X=0
```

```
FF10:  AA              TAX
FF11:  AA              TAX
FF12:  AA              TAX
FF13:  AA              TAX
FF14:  AA              TAX
FF15:  AA              TAX
FF16:  AA              TAX
FF17:  AA              TAX
FF18:  AA              TAX
FF19:  AA              TAX
FF1A:  AA              TAX
FF1B:  AA              TAX
FF1C:  AA              TAX
FF1D:  AA              TAX
FF1E:  AA              TAX
FF1F:  AA              TAX
```

```
                ;****************************************
                ;* Reset entry                          *
                ;****************************************
EAA0:  78       DSKINT SEI          ;no interrupts
EAA1:  D8              CLD          ;clear decimal mode
EAA2:  A2FF            LDX #$FF     ;direction is all outputs
EAA4:  4C10FF          JMP PATCH5   ;! changed: set CLK out hi
EAA7:  E8       DKIT10 INX          ;X=0

Added:
                ;****************************************
                ;* PATCH5:                              *
                ;*                                      *
                ;* Set CLK hi                           *
                ;*                                      *
                ;****************************************
FF10:  8E0318   PATCH5 STX DDRA1    ;set direction


FF13:  A902            LDA #$02     ;set CLK out high

FF15:  8D0018          STA PB       ;on bus


FF18:  A91A            LDA #$1A     ;set DDR port B

FF1A:  8D0218          STA DDRB1


FF1D:  4CA7EA          JMP DKIT10   ;and return
```

Difference #3: Improve the initialization of the VIA port controlling the bus lines on power on.

This change simply reverses the order of setting the port data and the associated direction register. The new order is setting the data first, then setting the line directions. This has the advantage that the port lines do not temporarily assume an undetermined value in the new situation; they immediately assume the desired state upon setting the direction.

```
EBCD:  A90A           LDA #10     ;set up
EBCF:  8569           STA SECINC  ;vector offset
EBD1:  A905           LDA #55     ;set up
EBD3:  856A           STA REVCNT  ;recovery count
EBD5:  A973           LDA #$73    ;get power on error code
EBD7:  20C1E6         JSR ERRTS0  ;set it
EBDA:  A91A           LDA #$1A    ;get direction bits

EBDC:  8D0218         STA DDRB1   set direction
EBDF:  A900           LDA #$00    ;set
EBE1:  8D0018         STA PB      ;bus lines
EBE4:  2080E7         JSR BOOT    ;check for diagnostic,
                                  ;if not:
               ;Idle loop
EBE7:  58      IDLE   CLI         ;no interrupts
```

```
EBCD:  A90A           LDA #10     ;set up
EBCF:  8569           STA SECINC  ;vector offset
EBD1:  A905           LDA #5      ;set up
EBD3:  856A           STA REVCNT  ;recovery count
EBD5:  A973           LDA #$73    ;get power on error code
EBD7:  20C1E6         JSR ERRTS0  ;set it
EBDA:  A900           LDA #$00    ;! changed: first set
                                  ;lines all hi
EBDC:  8D0018         STA PB      ;! changed: then
EBDF:  A91A           LDA #$1A    ;! changed: set
EBE1:  8D0218         STA DDRB1   ;! changed: direction
EBE4:  2080E7         JSR BOOT    ;check for diagnostic,
                                  ;if not:
              ; Idle loop
EBE7:  58      IDLE   CLI         ;no interrupts
```

Difference #4: Removal of the initials of the programmer.

The letters RSR at the end of the ROM (introduced with ROM 901229-02) were removed again.

```
FFE1:  AA             TAX
FFE2:  52             .BY 'R'     ;patcher's
FFE3:  53             .BY 'S'     ;initials
FFE4:  52             .BY 'R'
FFE5:  AA             TAX
```

```
FFE1:  AA             TAX
FFE2:  AA             TAX
FFE3:  AA             TAX
FFE4:  AA             TAX
FFE15  AA             TAX
```

This concludes the differences between the 1541 901229-03 and 901229-04 ROMs.

1541, 901229-05 ROM

The fifth ROM revision for the 1541 removed some remnants of the IEEE-488 interface of the 2031. In addition the power-on diagnostic sensing was removed, as its functionality was already largely covered by the UTLODR code.

All changes in ROM code were limited to the ROM for the area $E000-$FFFF (901229-05). The changes that occurred between ROMs 901229-03 and 901229-04 were retained unchanged.

Of the ROM revisions for the 1541 with built-in power supply, this is probably the most common.

Code differences compared to the ROM 901229-04

ROMs involved: 901229-04 and 901229-05

Difference #1: Remove IEEE-488 remnants.

```
                ; Talker error recovery
                ;
                ; If command channel, release DAV (n.a.)
                ; If data channel, force not ready
                ; (n.a.) and release channel
E680:  20EBD0   TLKERR JSR FNDRCH   ;find read channel
E683:  204EEA          JSR ITERR    ;release bus lines
                                    ;then go to idle

E686:  D006            BNE TLERR    ;(never executed)
                ; Listener error recovery
                ;
                ; If command channel, release RFD (n.a.)
                ; If data channel, force not ready (n.a.) and
                ; release channel
E688:  2007D1   LSNERR JSR FNDWCH   ;find write channel
E68B:  204EEA          JSR ILERR    ;release bus lines
                                    ;then go to idle

E68E:  2025D1   TLERR  JSR TYPFIL   ;(never executed)
E691:  C904            CMP #$04     ;if relative
E693:  B003            BCS ERR10
```

```
                ; Talker error recovery
                ;
                ; If command channel, release DAV (n.a.)
                ; If data channel, force not ready
                ; (n.a.) and release channel
E680:  20EBD0   TLKERR JSR FNDRCH   ;find read channel
E683:  EA              NOP          ;! changed: no JSR ITERR
E684:  EA              NOP
E685:  EA              NOP
E686:  D006            BNE TLERR
                ; Listener error recovery
                ;
                ; If command channel, release RFD (n.a.)
                ; If data channel, force not ready (n.a.) and
                ; release channel
E688:  2007D1   LSNERR JSR FNDWCH   ;find write channel
E68B:  EA              NOP          ;! changed: no JSR ILERR
E68C:  EA              NOP
E68D:  EA              NOP
E68E:  2025D1   TLERR  JSR TYPFIL   ;get file type
E691:  C904            CMP #$04     ;if relative
E693:  B003            BCS ERR10
```

Notes:

Although they have different labels, the routines ITERR and ILERR are one and the same. The routine itself was not removed from the ROM, as it is called in other parts of the code as well.

Difference #2: Remove diagnostic sensing code.

The 1540 and 1541 can be put in a diagnostic mode by externally grounding both the CLK and DATA serial bus lines and switching the drive on. After removal of the connections the UTLODR code is used to load a file over the bus. Upon completion of the load and a successful checksum check the loaded file is executed. The diagnostic sense part was removed with this ROM revision, the UTLODR was retained however. Note that the routine was replaced by an RTS instruction and NOPs but for the last byte of the routine: this was also replaced by an RTS instruction.

```
                ; Power on diagnostic sensing
E77F:  60       BOOT2  RTS          ;exit
E780:  AD0018   BOOT   LDA PB       ;get bus lines


E783:  AA              TAX          ;in X
E784:  2904            AND #$04     ;get CLK in

E786:  F0F7            BEQ BOOT2    ;if hi, exit

E788:  8A              TXA          ;get bus lines back
E789:  2901            AND #$01     ;get DATA in

E78B:  F0F2            BEQ BOOT2    ;if hi, exit

E78D:  58              CLI          ;if both CLK and DATA low,
                                    ;diag is set,
                                    ;allow interrupts
E78E:  AD0018   BOOT3  LDA PB       ;get bus lines
E791:  2905            AND #$05     ;wait for both to go hi

E793:  D0F9            BNE BOOT3

E795:  EE7802          INC F2CNT    ;


E798:  EE7402          INC CMDSIZ   ;file name length is 1


E79B:  A92A            LDA #'*'     ;any file name will do

E79D:  8D0002          STA CMDBUF


E7A0:  4CA8E7          JMP BOOT4    ;and go load test file
```

```
                ; Power on diagnostic sensing
E77F:  60       BOOT2  RTS          ;exit
E780:  60       BOOT   RTS          ;!: immediately return
E781:  EA              NOP          ;! changed
E782:  EA              NOP          ;! changed
E783:  EA              NOP          ;! changed
E784:  EA              NOP          ;! changed
E785:  EA              NOP          ;! changed
E786:  EA              NOP          ;! changed
E787:  EA              NOP          ;! changed
E788:  EA              NOP          ;! changed
E789:  EA              NOP          ;! changed
E78A:  EA              NOP          ;! changed
E78B:  EA              NOP          ;! changed
E78C:  EA              NOP          ;! changed
E78D:  EA              NOP          ;! changed
E78E:  EA              NOP          ;! changed
E78F:  EA              NOP          ;! changed
E790:  EA              NOP          ;! changed
E791:  EA              NOP          ;! changed
E792:  EA              NOP          ;! changed
E793:  EA              NOP          ;! changed
E794:  EA              NOP          ;! changed
E795:  EA              NOP          ;! changed
E796:  EA              NOP          ;! changed
E797:  EA              NOP          ;! changed
E798:  EA              NOP          ;! changed
E799:  EA              NOP          ;! changed
E79A:  EA              NOP          ;! changed
E79B:  EA              NOP          ;! changed
E79C:  EA              NOP          ;! changed
E79D:  EA              NOP          ;! changed
E79E:  EA              NOP          ;! changed
E79F:  EA              NOP          ;! changed
E7A0:  EA              NOP          ;! changed
E7A1:  EA              NOP          ;! changed
E7A2:  60              RTS          ;! changed
```

This concludes the differences between the 1541 901229-04 and 901229-05 ROMs.

1541, 901229-06 ROM

The sixth ROM revision for the 1541 improved formatting by clearing the format flag before starting the actual formatting of a disk.

All changes in ROM code were limited to the ROM for the area $E000-$FFFF (901229-06).The changes that occurred between ROMs 901229-04 and 901229-05 were retained unchanged.

This was the last revision to appear for the 1541 models. The next revision was for the 1541B model and used a 16 kbyte ROM instead of two 8 kbyte devices.

Code differences compared to the ROM 901229-05

ROMs involved: 901229-05 and 901229-06

Difference #1: Clear format flags before formatting.

```
EE39:  A901      LDA #1      ;start
EE3B:  8580      STA TRACK   ;with track 1
EE3D:  20C6C8    JSR FORMAT  ;transfer format to RAM
EE40:  2005F0    JSR CLRBAM  ;create an empty BAM
EE43:  4C56EE    JMP N110    ;and go write empty BAM
```

```
FF2F:  AA        TAX
FF30:  AA        TAX
FF31:  AA        TAX
FF32:  AA        TAX
FF33:  AA        TAX
FF34:  AA        TAX
```

```
EE39:  A901      LDA #1      ;start
EE3B:  8580      STA TRACK   ;with track 1
EE3D:  202FFF    JSR PATCH7  ;! changed: clear format
                             ;flags before formatting
EE40:  2005F0    JSR CLRBAM  ;create an empty BAM
EE43:  4C56EE    JMP N110    ;and go write empty BAM

                 ;***************************************
                 ;* PATCH7:                             *
                 ;*                                     *
                 ;* Clear format flags before formatting *
                 ;*                                     *
                 ;***************************************
FF2F:  A9FF    PATCH7 LDA #$FF   ;clear

FF31:  8551           STA FTNUM  ;format flags

FF33:  4CC6C8         JMP FORMAT ;and do old routine
                                 ;(transfer format to RAM)
```

This concludes the differences between the 1541 901229-05 and 901229-06 ROMs.

1541B, 251968-01 ROM

The first ROM revision for the 1541B introduced three changes. The first is the switch to a single 16 kbyte ROM instead of two 8 kbyte devices. This enabled the use of the second patch area in the range $C002-$C0FF without the need for two new ROMs. The area was not used with this ROM revision.

The second change is support for a track 0[3] sensor to get rid of the bumping while recalibrating. To initialize the track position, the 1541B recalibrates on power up; when the track 0 sensor is present this is almost silent and does not mechanically stress the drive mechanics.

On the main board, a solder blob is present to defeat the sensor. When closed, the drive will do the classic rattling bump (also on power up). The 1541B was produced with both Newtronics and ALPS mechanics; in practice only a part of the Newtronics drives actually had a track zero sensor.

The third change is an attempt to speed up the drive a bit by servicing the command(s) to the LCC or occurrences of ATNs not every 15 milliseconds, but every 8 milliseconds.

The changes in ROM code were still only present in the ROM area covering $E000-$FFFF (second half of ROM 251968-01).The changes that occurred between ROMs 901229-05 and 901229-06 were retained unchanged.

Code differences compared to the ROM 901229-06

ROMs involved: 901229-06 and 251968-01

[3] In Commodore DOS, there is no track zero. The outermost track of a drive has the number one. The common terminology with drive mechanisms uses the term track 0 sensor for the device that signals the head is at the outermost track. For reasons of clarity, the term track 0 was retained here and should be read as "outermost track" in the context of this document.

Difference #1: Track 0 sensor support, power on initialization.

```
                ;****************************************
                ;* RESET entry point                    *
                ;****************************************
EAA0:  78       DSKINT SEI          ;no interrupts
EAA1:  D8              CLD          ;clear decimal mode
EAA2:  A2FE            LDX #$FF     ;port A: all outputs

EAA4:  4C10FF          JMP PATCH5
EAA7:  C8              INX          ;X=0

                ;****************************************
                ;* PATCH5:                              *
                ;*                                      *
                ;* Set CLK hi                           *
                ;*                                      *
                ;****************************************
FF10:  8E0318   PATCH5 STX DDRA1    ;set direction
FF13:  A902            LDA #$02     ;set CLK out high
FF15:  8D0018          STA PB       ;on bus
FF18:  A91A            LDA #$1A     ;set DDR port B
FF1A:  8D0218          STA DDRB1
FF1D:  4CA7EA          JMP DKIT10   ;and return
```

```
                ;****************************************
                ;* RESET entry point                    *
                ;****************************************
EAA0:  78       DSKINT SEI          ;no interrupts
EAA1:  D8              CLD          ;clear decimal mode
EAA2:  A2FE            LDX #$FE     ;! changed: PA0 is an
                                    ;input (TRK0)
EAA4:  4C10FF          JMP PATCH5
EAA7:  C8       DKIT10 INX          ;X=0

                ;****************************************
                ;* PATCH5:                              *
                ;*                                      *
                ;* Set CLK hi                           *
                ;*                                      *
                ;****************************************
FF10:  8E0318   PATCH5 STX DDRA1    ;set direction
FF13:  A902            LDA #$02     ;set CLK out high
FF15:  8D0018          STA PB       ;on bus
FF18:  A91A            LDA #$1A     ;set DDR port B
FF1A:  8D0218          STA DDRB1
FF1D:  4CA7EA          JMP DKIT10   ;and return
```

Note: PATCH5 was already present since ROM 901229-04 and is unchanged.

Difference #2: Track 0 sensor support, perform the power on bump.

```
EBBA:  851D          STA WPSW+1  ;clear WP status drive 1
EBBC:  2063CB        JSR USRINT  ;initialize USER JMP
EBBF:  20FACE        JSR LRUINT  ;initialize LRU
EBC2:  2059F2        JSR CNTINT  ;controller initialization
EBC5:  A922          LDA #<DIAGOK
EBC7:  8565          STA VNMI
```

```
FF6F:  AA            TAX
FF70:  AA            TAX
FF71:  AA            TAX
FF72:  AA            TAX
FF73:  AA            TAX
FF74:  AA            TAX
FF75:  AA            TAX
FF76:  AA            TAX
FF77:  AA            TAX
FF78:  AA            TAX
FF79:  AA            TAX
FF7A:  AA            TAX
```

```
EBBA:  851D          STA WPSW+1  ;clear WP status drive 1
EBBC:  2063CB        JSR USRINT  ;initialize USER JMP
EBBF:  20FACE        JSR LRUINT  ;initialize LRU
EBC2:  206FFF        JSR PTCH10  ;! changed: perform a bump
EBC5:  A922          LDA #<DIAGOK
EBC7:  8565          STA VNMI

                ;****************************************
                ;* PTCH10:                              *
                ;*                                      *
                ;* Move head to track zero at end of    *
                ;* device initialization                *
                ;*                                      *
                ;****************************************
FF6F:  2059F2   PTCH10 JSR CNTINT  ;controller initialization


FF72:  A901          LDA #$01    ;command length

FF74:  8506          STA HDRS    ;is one byte

FF76:  A9C0          LDA #$C0    ;get bump command

FF78:  8500          STA JOBS    ;set it

FF7A:  60            RTS         ;and exit
```

Difference #3: Improve operating speed slightly by setting periodic timer interrupt to every 8 milliseconds.

```
F26C:  8D0C1C          STA IFR2-1
F26F:  A941            LDA #$41    ;cont IRQ, latch mode
F271:  8D0B1C          STA PCR2
F274:  A900            LDA #$00    ;get timer1 lo byte
F276:  8D061C          STA T1LL2   ;set it
F279:  A93A            LDA #$3A    ;15ms IRQ
F27B:  8D071C          STA T1HL2   ;set timer 2 hi
F27E:  8D 051C         STA T1HC2   ;get 6522's attention
F281:  A97F            LDA #$7F    ;clear all IRQ sources
```

```
F26C:  8D0C1C          STA IFR2-1
F26F:  A941            LDA #$41    ;cont IRQ, latch mode
F271:  8D0B1C          STA PCR2
F274:  A900            LDA #$00    ;get timer1 lo byte
F276:  8D061C          STA T1LL2   ;set it
F279:  A920            LDA #$20    ;! changed: 8ms IRQ
F27B:  8D071C          STA T1HL2   ;set timer 2 hi
F27E:  8D 051C         STA T1HC2   ;get 6522's attention
F281:  A97F            LDA #$7F    ;clear all IRQ sources
```

Difference #4: Track 0 sensor support, check for track 0 while stepping out.

```
FA2E:  A54A     DOSTEP LDA STEPS   ;get steps to do/direction
FA30:  1031            BPL STPIN   ;if stepping in, do that
FA32:  E64A            INC STEPS   ;update steps to do
FA34:  AE001C          LDX DSKCNT  ;get phase
FA37:  CA              DEX         ;update it

FA38:  4C69FA          JMP STP     ;and drive step motor
FA3B:  A54A     SHORT  LDA STEPS   ;get # of steps to do
```

```
FF36:  AA              TAX
```

up
to

```
FF6E:  AA              TAX
```

```
FA2E:  A54A     DOSTEP LDA STEPS   ;get steps to do/direction
FA30:  1031            BPL STPIN   ;if stepping in, do that
FA32:  4C36FF   STPOUT JMP PATCH9  ;!: check for track 0
FA35:  EA              NOP         ;! changed
FA36:  EA              NOP         ;! changed
FA37:  EA              NOP         ;! changed
FA38:  4C69FA   PPPPPP JMP STP     ;and drive step motor
FA3B:  A54A     SHORT  LDA STEPS   ;get # of steps to do

                ;****************************************
                ;* PATCH9:                              *
                ;*                                      *
                ;* Check if at track 0                  *
                ;*                                      *
                ;****************************************
FF36:  8A       PATCH9 TXA         ;save
FF37:  48              PHA         ;X
FF38:  98              TYA         ;and
FF39:  48              PHA         ;Y
FF3A:  A201            LDX #$01    ;get outer loop count
FF3C:  A064     PTCH91 LDY #100    ;get inner loop count
FF3E:  AD0F18   PTCH92 LDA POTA1   ;get port A
FF41:  CD0F18          CMP POTA1   ;if no change
FF44:  D01C            BNE PTCH93
FF46:  88              DEY         ;count down (inner loop)
FF47:  D0F5            BNE PTCH92  ;if inner not done, loop
FF49:  CA              DEX         ;else adapt outer loop
FF4A:  D0F0            BNE PTCH91  ;if not finished, loop
FF4C:  2901            AND #$01    ;else get port data
FF4E:  F012            BEQ PTCH93  ;if not at track 0
FF50:  AD001C          LDA DSKCNT  ;get controller phase
FF53:  2903            AND #$03    ;bits
FF55:  D00B            BNE PTCH93  ;if at phase A
FF57:  68              PLA         ;exit
FF58:  A8              TAY         ;restoring
FF59:  68              PLA         ;Y
FF5A:  AA              TAX         ;and X
FF5B:  A900            LDA #0      ;clear
FF5D:  854A            STA STEPS   ;number of steps to do
FF5F:  4CBEFA          JMP END33   ;and return to step code
FF62:  68       PTCH93 PLA         ;not at phase A, restore
FF63:  A8              TAY         ;Y
FF64:  68              PLA         ;and
FF65:  AA              TAX         ;X
FF66:  E64A            INC STEPS   ;count steps to do
FF68:  AE001C          LDX DSKCNT  ;get motor phase
FF6B:  CA              DEX         ;update it
FF6C:  4C38FA          JMP PPPPPP  ;and return
```

This concludes the differences between the 1541 901229-06 and 251968-01 ROMs.

1541B, 251968-02 ROM

The second ROM revision for the 1541B introduced quite a lot of changes and can be regarded as a major effort to settle all known issues. The second patch area in the range $C002-$C0FF was used for the first time with this ROM revision.

Some bugs were fixed that were probably discovered while developing the never released dual drive 1541D. The bugs would occur only when accessing drive one, which does not exist on a 1541.

The changes in ROM code were present in the entire ROM area covering $C000-$FFFF (251968-02). The changes that occurred between ROMs 901229-06 and 251968-01 were retained unchanged, apart from the 8 ms periodic interrupt that reverted back to 15 ms.

Code differences compared to the ROM 251968-01

ROMs involved: 251968-01 and 251968-02

<u>Difference #1: Addition of a copyright statement.</u>

```
                                                ;****************************************
                                                ;* Add copyright for legal types and    *
                                                ;* thieves                              *
                                                ;****************************************
C057:  AA              TAX                      C057:  434F50+  CRIGHT .BY 'COPYRIGHT (C)1985 COMMODORE'
up                                                                    .BY ' ELECTRONICS, LTD.',$0D
to                                                                    .BY 'ALL RIGHTS RESERVED",$0D
C098:  AA              TAX
```

Difference #2: Fix NODRV crossing a page.

The variable NODRV is used five times in the entire code, this change therefore yields five patches in total.

```
C1AD:  A57F     SCREN1 LDA DRVNUM   ;get current drive#
C1AF:  8D8E02          STA LSTDRV   ;as list drieve
C1B2:  AA              TAX          ;and as index
C1B3:  A900            LDA #0       ;set current drive status
C1B5:  95FF            STA NODRV,X  ;!BUG: fails with drive 1
C1B7:  20BDC1          JSR CLRCB
```

```
C1AD:  A57F     SCREN1 LDA DRVNUM   ;get current drive#
C1AF:  8D8E02          STA LSTDRV   ;get drive#
C1B2:  AA              TAX          ;as index
C1B3:  4CB8FF          JMP PTCH43   ;! changed: NODRV is abs.
C1B6:  EA              NOP          ;needed because of patch
C1B7:  20BDC1   RTCH43 JSR CLRCB

                       ;****************************************
                       ;* PTCH43:                              *
                       ;*                                      *
                       ;* Change NODRV to absolute address     *
                       ;*                                      *
                       ;* (because of X indexing)              *
                       ;*                                      *
                       ;****************************************
FFB8:  A900     PTCH43 LDA #0       ;clear

FFBA:  9DFF00          .BY $9D,$FF,$00      ;write NODRV,X
                                            ;ABSOLUTE

FFBD:  4CB7C1          JMP RTCH43   ;and return
```

```
FFB8:  AA              TAX
FFB9:  AA              TAX
FFBA:  AA              TAX
FFBB:  AA              TAX
FFBC:  AA              TAX
FFBD:  AA              TAX
FFBE:  AA              TAX
FFBF:  AA              TAX
```

```
C65D:  A000            LDY #0
C65F:  A67F     AUTO1  LDX DRVNUM   ;get drive number as index
C661:  98              TYA          ;get status
C662:  95FF            STA NODRV,X  ;set status
C664:  D003     RTCH20 BNE LC669
C666:  2042D0          JSR INITDR
C669:  A67F     AUTO2  LDX DRVNUM   ;get drive number as index
C66B:  B5FF            LDA NODRV,X  ;!BUG drive1: get status
C66D:  60              RTS          ;and exit
```

```
C65D:  A000            LDY #0
C65F:  A67F     AUTO1  LDX DRVNUM   ;get drive number
C661:  4CC0FF          JMP PTCH44   ;! changed

C664:  D003     RTCH44 BNE LC669
C666:  2042D0          JSR INITDR
C669:  A67F     AUTO2  LDX DRVNUM   ;get drive number as index
C66B:  4CACFF          JMP PTCH50   ;! changed:
C66D:  60              RTS          ;and exit (never executed)

                       ;****************************************
                       ;* PTCH50:                              *
                       ;*                                      *
                       ;* Change NODRV to absolute address     *
                       ;*                                      *
                       ;* (because of X indexing)              *
                       ;*                                      *
                       ;****************************************
FFAC:  BDFF00   PTCH50 .BY $BD,$FF,$00      ;read NODRV,X
                                            ;ABSOLUTE

FFAF:  60       RTS
                       ;****************************************
                       ;* PTCH44:                              *
                       ;*                                      *
                       ;* Change NODRV to absolute address     *
                       ;*                                      *
                       ;* (because of X indexing)              *
                       ;*                                      *
                       ;****************************************
FFC0:  98       PTCH44 TYA
FFC1:  9DFF00          .BY $9D,$FF,$00      ;write NODRV,X
                                            ;ABSOLUTE

FFC4:  4C64C6          JMP RTCH44
```

```
FFAC:  AA              TAX
FFAD:  AA              TAX
FFAE:  AA              TAX
FFAF:  AA              TAX
FFC0:  AA              TAX
FFC1:  AA              TAX
FFC2:  AA              TAX
FFC3:  AA              TAX
FFC4:  AA              TAX
FFC5:  AA              TAX
FFC6:  AA              TAX
```

```
D06A:  A67F           LDX DRVNUM  ;get drive# as index
D06C:  9D0101         STA DSKVER,X
D06F:  A900           LDA #0
D071:  951C           STA WPSW,X  ;clear WP switch
D073:  95FF           STA NODRV,X ;!BUG: fails with drive 1
D075:  203AEF         JSR SETBPT
```

```
FFC7:  AA             TAX
FFC8:  AA             TAX
FFC9:  AA             TAX
FFCA:  AA             TAX
FFCB:  AA             TAX
FFCC:  AA             TAX
FFCD:  AA             TAX
FFCE:  AA             TAX
```

```
D06A:  A67F           LDX DRVNUM  ;get drive# as index
D06C:  9D0101         STA DSKVER,X
D06F:  A900           LDA #0
D071:  4CC7FF         JMP PTCH51  ;! changed: NODRV is abs.
D074:  EA             NOP         ;needed because of patch
D075:  203AEF   RTCH51 JSR SETBPT

                ;***************************************
                ;* PTCH51:                             *
                ;*                                     *
                ;* Change NODRV to absolute address    *
                ;*                                     *
                ;* (because of X indexing)             *
                ;*                                     *
                ;***************************************
FFC7:  951C     PTCH51 STA WPSW,X  ;clear WP switch

FFC9:  9DFF00          .BY $9D,$FF,$00      ;write NODRV,X
                                            ;ABSOLUTE

FFCC:  4C75D0          JMP RTCH51  ;and return
```

```
F011:  A56F     LF011  LDA TEMP
F013:  48              PHA
F014:  A570            LDA T0+1
F016:  48              PHA
F017:  A67F     PTCH52 LDX DRVNUM  ;get drive# as  index
F019:  B5FF            LDA NODRV,X ;!BUG drive1: get status
F01B:  F005     RTCH52 BEQ LF022
```

```
FFB0:  AA             TAX
FFB1:  AA             TAX
FFB2:  AA             TAX
FFB3:  AA             TAX
FFB4:  AA             TAX
FFB5:  AA             TAX
FFB6:  AA             TAX
FFB7:  AA             TAX
```

```
F011:  A56F     LF011  LDA TEMP
F013:  48              PHA
F014:  A570            LDA T0+1
F016:  48              PHA
F017:  4CB0FF          JMP PTCH52  ;! changed: NODRV is abs.
F01A:  EA              NOP         ;needed because of patch
F01B:  F005     RTCH52 BEQ LF022

                ;***************************************
                ;* PTCH52:                             *
                ;*                                     *
                ;* Change NODRV to absolute address    *
                ;*                                     *
                ;* (because of X indexing)             *
                ;*                                     *
                ;***************************************
FFB0:  A67F     PTCH52 LDX DRVNUM  ;get drive# as index

FFB2:  BDFF00   .      BY $BD,$FF,$00       ;read NODRV,X
                                            ;ABSOLUTE

FFB5:  4C1BF0          JMP RTCH52  ;and return
```

Clarification: The IEEE-488 dual floppy drives of the past used two zero page RAM locations to flag whether or not a drive was in use. The two RAM locations were defined as variable NODRV with a length of two bytes. Getting the active flag involved loading register X with the drive number and getting the status with an LDA NODRV,X instruction. Over time some variables were moved around and some were added, with the result that in the V(I)C-1540 and 1541 NODRV ended up at locations $FF and $100, apparently without anybody noticing. When the aforementioned instruction LDA NODRV,X is assembled with NODRV defined at $FF, the assembler will use zero page addressing.

In this particular instance this introduces a bug that is partly responsible for the infamous save-with-replace bug. When using zero page addressing the 6502 CPU will not cross a page when indexing (whether this is a bug or a feature is debatable; in later (CMOS) processors the feature was retained). In this case, when X=1 the processor would load the contents of location $00 and not that of location $100 which would be correct. The bug was fixed by changing the LDA NODRV,X instruction to use absolute indexed addressing. In practice however X=1 would never occur, as the 1541 only has one drive mechanism. This leads to the conclusion that the problem was probably discovered during the development of the dual drive 1541D which was never released. Anyway, the code is now bomb proof regarding this aspect...

Difference #3: Fix for block read (B-R).

```
CD8C:  2064D4   BW20   JSR DRTWRT   ;write block
CD8F:  68              PLA
CD90:  AA              TAX
CD91:  20CDED3         JSR RNDGET
CD94:  4C94C1          JMP ENDCMD




C024:  AA              TAX
C025:  AA              TAX
C026:  AA              TAX
C027:  AA              TAX
C028:  AA              TAX
```

```
CD8C:  2064D4   BW20   JSR DRTWRT   ;write block
CD8F:  68              PLA
CD90:  AA              TAX
CD91:  2024C0          JSR PTCH15   ;!: load Y with LINDX
                                    ;(fix for block read)
CD94:  4C94C1          JMP ENDCMD

                ;****************************************
                ;* PTCH15:                              *
                ;*                                      *
                ;* Load Y with LINDX                    *
                ;*                                      *
                ;****************************************
C024:  A482     PTCH15 LDY LINDX    ;! changed: get LINDX

C026:  4CDED3          JMP RNDGET
```

Difference #4: Check for reportable errors after stealing a buffer.

```
D363:  B500     STL40  LDA JOBS,X  ;get job status
D365:  30FC            BMI STL40   ;if not finished, wait
D367:  C902            CMP #2      ;error# >2?
D369:  9007            BCC STL50   ;if not, proceed
D36B:  A66F     RTCH54 LDX T0
```

```
FF91:  AA              TAX

up
to

FF9E:  AA              TAX
```

```
D363:  B500     STL40  LDA JOBS,X  ;get job status
D365:  30FC            BMI STL40   ;if not finished, wait
D367:  4C91FF          JMP PTCH54  ;!: check for errors
D36A:  EA              NOP         ;needed because of patch
D36B:  A66F     RTCH54 LDX T0

                       ;****************************************
                       ;* PTCH54:                              *
                       ;*                                      *
                       ;* Check if reportable errors present   *
                       ;*                                      *
                       ;****************************************
FF91:  C902     PTCH54 CMP #2      ;error# >2?
FF93:  9007            BCC PTH541  ;if not, proceed
FF95:  C90F            CMP #$0F    ;! added: else if error 15
FF97:  F003            BEQ PTH541  ;proceed
FF99:  4C6BD3          JMP RTCH54  ;else return reporting
                                   ;error
FF9C:  4C73D3   PTH541 JMP STL50
```

Difference #5: Set and clear micro stepping flag when recovering from read problems.

```
D60D   B9DBFE          LDA OFFSET,Y
D610:  2076D6          JSR HEDOFF   ;move head
D613:  EE9902          INC EPTR
D616:  20A6D6          JSR DOREC
D619:  C902            CMP #2       ;error code < 2?
D61B:  9008            BCC LD625
D61D:  AC9902          LDY EPTR
D620:  B9DBFE          LDA OFFSET,Y
D623:  D0DB            BNE LD600
D625:  AD9A02   LD625  LDA TOFF
D628:  2076D6          JSR HEDOFF   ;move head
```

```
FF84:  AA              TAX
```

up
to

```
FF90:  AA              TAX
```

```
D60D   B9DBFE          LDA OFFSET,Y
D610:  2084FF          JSR PTCH12   ;!: set micro stepping flg
D613:  EE9902          INC EPTR
D616:  20A6D6          JSR DOREC
D619:  C902            CMP #2       ;error code < 2?
D61B:  9008            BCC LD625
D61D:  AC9902          LDY EPTR
D620:  B9DBFE          LDA OFFSET,Y
D623:  D0DB            BNE LD600
D625:  AD9A02   LD625  LDA TOFF
D628:  2089 FF         JSR PTCH13   ;!: clr micro stepping flg

                ;****************************************
                ;* PTCH12:                              *
                ;*                                      *
                ;* Set micro stepping flag              *
                ;*                                      *
                ;****************************************
FF84:  857B     PTCH12 STA ADRSED   ;set micro stepping flag
FF86:  4C76D6          JMP HEDOFF   ;and move head
                ;****************************************
                ;* PTCH13:                              *
                ;*                                      *
                ;* Clear micro stepping flag            *
                ;*                                      *
                ;****************************************
FF89:  2076D6   PTCH13 JSR HEDOFF   ;move head
FF8C:  A900            LDA #0       ;and
FF8E:  857B            STA ADRSED   ;clear micro stepping flag
FF90:  60              RTS          ;then exit
```

Difference #6: Improve initializing variables when opening a file.

```
DCB6:  A682     INITP  LDX LINDX
DCB8:  B5A7            LDA BUF0,X
DCBA:  0A              ASL A

DCBB:  A8              TAY          ;
DCBC:  A902            LDA #$02     ;
DCBE:  999900          STA BUFTAB,Y ;
DCC1:  B5AE            LDA BUF1,X   ;
DCC3:  0980            ORA #$80     ;
DCC5:  95AE            STA BUF1,X   ;
DCC7:  0A              ASL A        ;
DCC8:  A8              TAY          ;

DCC9:  A902            LDA #$02     ;
DCCB:  999900          STA BUFTAB,Y ;
DCCE:  A900            LDA #0       ;clear
DCD0:  95B5            STA NBKL,X   ;
DCD2:  95BB            STA NBKH,X   ;
DCD4:  A900            LDA #0       ;
DCD6:  9D4402          STA LSTCHR,X ;
DCD9:  60              RTS          ;and exit
```

```
C029:  AA              TAX
up
to

C032:  AA              TAX
```

```
DCB6:  A682     INITP  LDX LINDX
DCB8:  B5A7            LDA BUF0,X
DCBA:  0A              ASL A        ;get next bit
DCBB:  3006            BMI INITP1   ;! added: if clear
DCBD:  A8              TAY          ;save current byte in Y
DCBE:  A902            LDA #$02     ;
DCC0:  999900          STA BUFTAB,Y ;
DCC3:  B5AE     INITP1 LDA BUF1,X   ;
DCC5:  0980            ORA #$80     ;set MSbit
DCC7:  95AE            STA BUF1,X   ;
DCC9:  0A              ASL A        ;get next bit
DCCA:  3006            BMI INITP2   ;! added:
DCCC:  A8              TAY          ;save A in Y as index
DCCD:  A902            LDA #$02     ;set
DCCF:  999900          STA BUFTAB,Y ;
DCD2:  A900     INITP2 LDA #0       ;clear
DCD4:  95B5            STA NBKL,X   ;
DCD6:  4C29C0          JMP PTCH41   ;! changed:
DCD9:  EA              NOP          ;(never executed)

                ;*************************************
                ;* PTCH41:                           *
                ;*                                   *
                ;* Complete INITP routine            *
                ;*                                   *
                ;*************************************
C029:  95B5     PTCH41 STA NBKL,X   ;! done already
C02B:  95BB            STA NBKH,X
C02D:  A900            LDA #$00     ;clear
C02F:  9D4402          STA LSTCHR,X
C032:  60              RTS          ;and exit
```

Difference #7: Fix HEXDEC routine: no interrupts and restore state of D flag.

```
E69B:  AA       HEXDEC TAX          ;get number to convert
E69C:  A900            LDA #$00     ;clear result register
E69E:  F8              SED          ;set decimal mode
E69F:  E000     HEX0   CPX #0       ;if result zero, now
E6A1:  F007            BEQ HEX5     ;we're done
                       ;! BUG: decimal mode remains set
```

```
C033:  AA              TAX
```

up
to

```
C046:  AA              TAX
```

```
E69B:  AA       HEXDEC TAX
E69C:  4C33C0          JMP PTCH67   ;! changed: retain D flag
E69F:  E000     HEX0   CPX #0       ;if done (never ex.)
E6A1:  F007            BEQ HEX5     ;proceed (never executed)

                       ;****************************************
                       ;* PTCH67:                              *
                       ;*                                      *
                       ;* Complete INITP routine               *
                       ;*                                      *
                       ;****************************************
C033:  08       PTCH67 PHP          ;save flags
C034:  78              SEI          ;no interrupts
C035:  A900            LDA #0       ;clear result register
C037:  F8              SED          ;set decimal mode
C038:  E000     PTH671 CPX #0       ;if input is 0 now,
C03A:  F007            BEQ PTH672   ;we're done
C03C:  18              CLC          ;prepare for add
C03D:  6901            ADC #1       ;increment A
C03F:  CA              DEX          ;count tens
C040:  4C38C0          JMP PTH671   ;and loop
C043:  28       PTH672 PLP          ;restore flags (incl. D,I)
C044:  4CAAE6          JMP HEX5     ;and continue
```

Difference #8: Test for ATN at end of reset code.

```
E780:  60       BOOT   RTS          ;return immediately
E781:  EA              NOP
E781:  EA              NOP
```

```
E780: 4C59EA    BOOT   JMP TSTATN   ;! changed: test for ATN
```

Difference #9: Clear IRQ flag in VIA when servicing a new ATN while in ATN mode.

```
EA59:  A57D     TSTATN LDA ATNMOD   ;test if in ATN mode
EA5B:  F006            BEQ TSTR50   ;if not, check if new ATN
EA5D:  AD0018          LDA PB       ;else
EA60:  1009            BPL TATN20   ;if ATN gone, do what we
                                    ;have to do
EA62:  60       TSTRTN RTS          ;else exit
EA63:  AD0018   TSTR50 LDA PB       ;still in ATN mode
EA66:  10FA            BPL TSTRTN   ;if no ATN, exit
EA68:  4C5BE8          JMP ATNSRV   ;else service ATN
                                    ;!BUG: IRQ flag in 6522
                                    ;is not cleared
EA6B:  4C D7 E8 TATN20 JMP ATNS20
```

```
FFA6:  AA              TAX
FFA7:  AA              TAX
FFA8:  AA              TAX
FFA9:  AA              TAX
FFAA:  AA              TAX
FFAB:  AA              TAX
```

```
EA59:  A57D     TSTATN LDA ATNMOD   ;test if in ATN mode
EA5B:  F006            BEQ TSTR50   ;if not, check if new ATN
EA5D:  AD0018          LDA PB       ;else
EA60:  1009            BPL TATN20   ;if ATN gone, do what we
                                    ;have to do
EA62:  60       TSTRTN RTS          ;else exit
EA63:  AD0018   TSTR50 LDA PB       ;still in ATN mode
EA66:  10FA            BPL TSTRTN   ;if no ATN, exit
EA68:  4CA6FF          JMP PTCH30   ;! changed: else clear
                                    ;IRQ flag and service ATN
EA6B:  4C D7 E8 TATN20 JMP ATNS20

                ;****************************************
                ;* PTCH30:                              *
                ;*                                      *
                ;* Clear IRQ flags in VIA before        *
                ;* servicing new ATN                    *
                ****************************************
FFA6:  2C0118   PTCH30 BIT PA1      ;clear IRQ flag


FFA9:  4C5BE8          JMP ATNSRV   ;do ATN command
```

Difference #10: Execute reset routine with interrupts blocked.

```
EB22:  A245     DIAGOK LDX #$45    ;get top of stack
EB24:  9A              TXS         ;set stack pointer
EB25:  AD001C          LDA LEDPRT  ;get LED port
```

```
FF9F:  AA              TAX
FFA0:  AA              TAX
FFA1:  AA              TAX
FFA2:  AA              TAX
FFA3:  AA              TAX
FFA4:  AA              TAX
FFA5:  AA              TAX
```

```
EB22:  4C9FFF   DIAGOK JMP PTCH31  ;! changed: add SEI,
                                   ;init SP
EB25:  AD001C   RTCH31 LDA LEDPRT  ;get LED port

                ;****************************************
                ;* PTCH31:                              *
                ;*                                      *
                ;* block interrupts before initializing *
                ;* stack pointer                        *
                ;*                                      *
                ;****************************************
FF9F:  78       PTCH31 SEI         ;added: no interrupts
FFA0:  A245            LDX #$45    ;initialize

FFA2:  9A              TXS         ;stack pointer
FFA3:  4C25EB          JMP RTCH31  ;and proceed
```

Difference #11: At the end of the selftest, recalibrate head to outermost track.

```
EBBF:  20FACE          JSR LRUINT   ;initialize the LRU table
EBC2:  206FFF          JSR PTCH10   ;perform a bump
```

```
FF73:  AA              TAX
FF74:  AA              TAX
FF75:  AA              TAX
FF76:  AA              TAX
FF77:  AA              TAX
FF78:  AA              TAX
FF79:  AA              TAX
FF7A:  AA              TAX
FF7B:  AA              TAX
FF7C:  AA              TAX
FF7D:  AA              TAX
FF7E:  AA              TAX
```

```
EBBF:  20FACE          JSR LRUINT   ;initialize the LRU table
EBC2:  2073FF          JSR PTCH10   ;PATCH: perform a bump

                    ****************************************
                    * PTCH10:                              *
                    *                                      *
                    * Move head to track zero at end of    *
                    * device initialization                *
                    *                                      *
                    ****************************************
FF73:  2059F2   PTCH10 JSR CNTINT   ;controller init


FF76:  A901            LDA #$01     ;command is 1 byte long

FF78:  8506            STA HDRS

FF7A:  A9C0            LDA #$C0     ;get bump command

FF7C:  8500            STA JOBS     ;set it

FF7E:  60              RTS          ;and exit
```

This difference is listed because PTCH10 was moved. The code itself is unchanged.

Difference #12: Set drive active before starting format.

```
EE19:  2901     N101   AND #$01     ;get drive#
EE1B:  857F            STA DRVNUM   ;set it
EE1D:  2000C1          JSR SETLDS   ;update drive LEDs

EE20:  A57F            LDA DRVNUM
```

```
FF7F:  AA              TAX
FF80:  AA              TAX
FF81:  AA              TAX
FF82:  AA              TAX
FF83:  AA              TAX
```

```
EE19:  2901     N101   AND #$01
EE1B:  857F            STA DRVNUM
EE1D:  207FFF          JSR PTCH11   ;! changed: add clear
                                    ;NODRV (with bug)
EE20:  A57F            LDA DRVNUM

                ;****************************************
                ;* PTCH11:                              *
                ;*                                      *
                ;* clear NODRV                          *
                ;****************************************
FF7F:  85FF     PTCH11 STA NODRV    ;clear NODRV
                                    ;!BUG: drive#
FF81:  4C00C1          JMP SETLDS   ;and set LEDs
```

This patch has serious problems in the context of this ROM revision. Difference #2 is a major effort to fix the potential page cross problem with NODRV. Yet the patch:

1. repeats the same error by using zero page addressing;
2. forgets to index for drive 1 support;
3. fails when drive 1 would be actually used, because NODRV for drive 0 (!) would be set to the wrong value of 1.

Amazingly, the rest of the NEW routine handles drive 1 correctly. When will we ever learn...

A correct patch would be:

```
FF7F:  AA       PTCH11 TAX          ;get drive# as index
FF80:  A900            LDA #0       ;set current drive inactive
FF82:  9DFF00          .BY $9D,$FF,$00      ;write NODRV,X
                                    ;ABSOLUTE
FF85:  4C00C1          JMP SETLDS   ;and set LEDs
```

Difference #13: If less than 3 blocks free, report 1 block free.

The purpose of this change is to fix the disk full bug. When three or less free blocks remain, the DOS starts reporting one block free. The idea behind this to have sufficient blocks free (if possible) in order to be able to close a potentially open write file successfully. A side effect is that under these circumstances, a directory listing will show an incorrect number of free blocks, a condition that can be rectified by validating the disk.

```
EFBD:  BDFC02   USE20  LDA NDBH,X   ; get blocks free hi
EFC0:  D00C            BNE USERTS   ;if not zero, exit
EFC2:  BDFA02          LDA NDBL,X   ;get blocks free lo
EFC5:  C903            CMP #$03     ;if less than 3
EFC7:  B005            BCS USERTS

EFC9:  A972            LDA #$72     ;get error number (Disk
                                    ;full)
EFCB:  20C7E6          JSR ERRMSG   ;and report it
EFCE:  60       USERTS RTS          ;then exit

C047:  AA              TAX

up
to

C052:  AA              TAX
```

```
EFBD:  BDFC02   USE20  LDA NDBH,X   ;get blocks free hi
EFC0:  D00C            BNE USERTS   ;if not zero, exit
EFC2:  BDFA02          LDA NDBL,X   ;get blocks free lo
EFC5:  4C47C0          JMP PCTH66   ;! changed: keep at least
                                    ;2 blocks available
EFC8:  EA              NOP          ;needed because of patch
EFC9:  A972            LDA #$72     ;get error number (Disk
                                    ;full) (never executed)
EFCB:  20C7E6          JSR ERRMSG   ;and report it(never ex.)
EFCE:  60       USERTS RTS          ;then exit

                ;****************************************
                ;* PTCH66:                              *
                ;*                                      *
                ;* If less than 3 blocks free, report   *
                ;* only one block free to keep 2        *
                ;* sectors available for a successful   *
                ;* file close.                          *
                ;*                                      *
                ;****************************************
C047:  C903     PCTH66 CMP #3       ;if 3 or less blocks free
C049:  B005            BCS PTH661   ;report 1 block free
C04B:  A972            LDA #$72     ;else
C04D:  20C7E6          JSR ERRMSG   ;report disk full
C050:  A901     PTH661 LDA #1       ;get remaining blocks
C052:  60              RTS          ;and exit
```

Difference #14: Reverse change of periodic interrupt every 8 ms.

The reason given in the release documentation for this reversal is compatibility; some software in the field apparently uses the 15 ms period between IRQs for timing purposes. (This is bad programming practice: one should not rely on internals of an operating environment).

```
F26C:  8D0C1C          STA IFR2-1
F26F:  A941            LDA #$41    ;cont IRQ, latch mode
F271:  8D0B1C          STA PCR2
F274:  A900            LDA #$00    ;get timer1 lo byte
F276:  8D061C          STA T1LL2   ;set it
F279:  A920            LDA #$20    ;8ms IRQ
F27B:  8D071C          STA T1HL2   ;set timer 2 hi
F27E:  8D 051C         STA T1HC2   ;get 6522's attention
F281:  A97F            LDA #$7F    ;clear all IRQ sources
```

```
F26C:  8D0C1C          STA IFR2-1
F26F:  A941            LDA #$41    ;cont IRQ, latch mode
F271:  8D0B1C          STA PCR2
F274:  A900            LDA #$00    ;get timer1 lo byte
F276:  8D061C          STA T1LL2   ;set it
F279:  A93A            LDA #$3A    ;! changed: 15ms IRQ
F27B:  8D071C          STA T1HL2   ;set timer 2 hi
F27E:  8D 051C         STA T1HC2   ;get 6522's attention
F281:  A97F            LDA #$7F    ;clear all IRQ sources
```

This change reverses Difference #3 in ROM 251968-01.

Difference #15: Expand track zone table to 40 tracks.

The drive mechanism in the 1541 is actually 40 tracks, not 35 as in the 4040. Some software accessed tracks 36-40 (or even up to 42). The problem is that the zone table stops at track 35, so the bit density would not be set correctly for tracks over 35.

```
F33C:  A204     GOTU   LDX #4      ;set track, sector
F33E:  B132            LDA (HDRPNT),Y       ;get track#
F340:  8540            STA TRACC   ;save it
F342:  DDD6FE   LF342  CMP TRKNUM-1,X       ;compare
F345:  CA              DEX         ;get zone number
```

```
C053:  AA              TAX
C054:  AA              TAX
C055:  AA              TAX
C056:  AA              TAX
```

```
F33C:  A204     GOTU   LDX #4      ;set track, sector
F33E:  B132            LDA (HDRPNT),Y
F340:  8540            STA TRACC
F342:  DD52C0   LF342  CMP TRACKN-1,X       ;! changed: new
zone table
F345:  CA              DEX

               ;****************************************
               ;* New track zone table                 *
               ;****************************************
C053:  29      TRACKN .BY 41       ;! changed: track 36->41
C054:  1F             .BY 31
C055:  19             .BY 25
C056:  12             .BY 18
```

Note that the old unmodified table is left in place rather than changing one byte in it. This approach ensures greater compatibility in cases the original table would be used in a user program.

Strictly speaking this is in effect a bug in the LCC: it allows the use of track numbers that are officially not there, but in the 1541 physically do exist.

Commodore did not use these facts to use all 40 physical tracks in the 1541 giving it more storage space while at the same time allowing 35 track formatted diskettes to be used for compatibility with older drives; a gap that was successfully exploited by other software.

Difference #16: Change test for minimum gap2 from 4 to 2.

```
FC17:  A200           LDX #$00
FC19:  38       DS20  SEC
FC1A:  E543           SBC SECTR
FC1C:  B003           BCS DS22
FC1E:  88             DEY
FC1F:  3003           BMI DS30
FC21:  E8       DS22  INX
FC22:  D0F5           BNE DS20
FC24:  8E2606   DS30  STX DTRCK
FC27:  E002           CPX #4      ;gap2
FC29:  B005           BCS DS32
```

```
FC17:  A200           LDX #$00
FC19:  38       DS20  SEC
FC1A:  E543           SBC SECTR
FC1C:  B003           BCS DS22
FC1E:  88             DEY
FC1F:  3003           BMI DS30
FC21:  E8       DS22  INX
FC22:  D0F5           BNE DS20
FC24:  8E2606   DS30  STX DTRCK
FC27:  E002           CPX #2      ;! changed: gap2 was 4
FC29:  B005           BCS DS32
```

Difference #17: Clear disk before formatting with less bytes.

```
FCAA:  A900            LDA #$00
FCAC:  8532            STA HDRPNT
FCAE:  200EFE          JSR CLEAR
FCB1:  A9FF     LFCB1  LDA #$FF

FE0E:  AD0C1C   CLEAR  LDA PCR2    ;enable write
FE11:  291F            AND #$1F
FE13:  09C0            ORA #$C0
FE15:  8D0C1C          STA PRC2
FE18:  A9FF            LDA #$FF
FE1A:  8D031C          STA DDR2
FE1D:  A955            LDA #$55
FE1F:  8D011C          STA DATA2
FE22:  A228            LDX 40      ;get outer loop count
FE24:  A000            LDY #0      get inner loop cnt (256)
FE26:  50FE     CLER10 BVC CLER10  ;wait for controller
FE28:  B8              CLV         ;clear controller ready
FE29:  88              DEY         ;count inner loop
FE2A:  D0FA            BNE CLER10  ;if not done, loop
FE2C:  CA              DEX         count outer loop
FE2D:  D0F7            BNE CLER10  ;if not done, loop
FE2F:  60              RTS         ;else exit
```

```
FCAA:  A900            LDA #$00    ;init counter
FCAC:  8532            STA HDRPNT
FCAE:  2002C0          JSR CLEAR   ;! new CLEAR routine
FCB1:  A9FF     WRTSYN LDA #$FF

C002:  AD0C1C   CLEAR  LDA PCR2    ;enable write
C005:  291F            AND #$1F    ;clear bit 5-7
C007:  09C0            ORA #$C0    ;set bit 6 & 7
C009:  8D0C1C          STA PCR2    ;
C00C:  A9FF            LDA #$FF    ;make port an output
C00E:  8D031C          STA DDRA2
C011:  A955            LDA #$55    ;write a 1F pattern
C013:  8D011C          STA DATA2
C016:  A203            LDX #3      ;!: get outer loop count
C018:  A000            LDY #0      ;get inner loop cnt (256)
C01A:  50FE     CLER10 BVC CLER10  ;wait for controller
C01C:  B8              CLV         ;clear controller ready
C01D:  88              DEY         ;count inner loop
C01E:  D0FA            BNE CLER10  ;if not done, loop
C020:  CA              DEX         ;count outer loop
C021:  D0F7            BNE CLER10  ;if not done, loop
C023:  60              RTS         ;else exit
```

Note: the routine CLEAR on the left is still present in the ROM unchanged, and has been listed for comparison reasons. The only difference is the number of bytes the disk is cleared with: it used to be 256 x 40 = 10240 bytes (more than two tracks) the best part of an entire track), the new value is 256 x 3 = 768 bytes (about two disk sectors).

This concludes the differences between the 1541 251968-01 and 251968-02 ROMs.

1541-II, 251968-03 ROM

This is the first and effectively only ROM version for the 1541-II. It is generally speaking the last ROM revision for the 1541B (ROM 251968-02) with the support for the track 0 sensor removed. The changed copyright statement and some patches have been moved to different locations, but their code has not changed; only changes in code have been listed below.

The changes in ROM code were present in the entire ROM area covering $C000-$FFFF (251968-03).The changes that occurred between ROMs 251958-01 and 251968-02 were retained largely unchanged, apart from the removal of the support for the track 0 sensor, return to the old zone table and control of the micro stepping flag. The test using gap2 now uses the old value for gap2 (4) again.

The ROM 251968-02 will also work in the 1541-II, because its main board has the VIA input PA0 used for the track 0 sensor on the 1541B connected to ground, defeating the sensor. Commodore probably grounded the input to be able to use the 251968-02 ROM for the 1541B in the 1541-II. The most probable cause for this is that by the time the production of the 1541-II started, the 251968-03 ROM code was not ready yet. A 1541-II equipped with a 252968-02 ROM will bump on power up.

Code differences compared to the ROM 251968-02

ROMs involved: 251968-02 and 251968-03

Difference #1: Copyright statement changed and moved.

Space previously occupied by now unused or moved patches.

```
C002:  434F+          .BY 'COPYRIGHT (C)1982,1985,1987'
                      .BY ' COMMODORE ELECTRONICS, LTD.'
C039:  0D             .BY $0D
C03A:  414C+          .BY 'ALL RIGHTS RESERVED'
C04D:  0D             .BY $0D
```

Difference #2: Set/clear micro stepping flag removed.

```
D60D   B9DBFE          LDA OFFSET,Y
D610:  2084FF          JSR PTCH12  ;!: set micro stepping flg
D613:  EE9902          INC EPTR
D616:  20A6D6          JSR DOREC
D619:  C902            CMP #2      ;error code < 2?
D61B:  9008            BCC LD625
D61D:  AC9902          LDY EPTR
D620:  B9DBFE          LDA OFFSET,Y
D623:  D0DB            BNE LD600
D625:  AD9A02   LD625  LDA TOFF
D628:  2089 FF         JSR PTCH13  ;!: clr micro stepping flg

               ;****************************************
               ;* PTCH12:                              *
               ;*                                      *
               ;* Set micro stepping flag              *
               ;*                                      *
               ;****************************************
FF84:  857B    PTCH12 STA ADRSED  ;set micro stepping flag
FF86:  4C76D6         JMP HEDOFF  ;and move head
               ;****************************************
               ;* PTCH13:                              *
               ;*                                      *
               ;* Clear micro stepping flag            *
               ;*                                      *
               ;****************************************
FF89:  2076D6  PTCH13 JSR HEDOFF  ;move head
FF8C:  A900           LDA #0      ;and
FF8E:  857B           STA ADRSED  ;clear micro stepping flag
FF90:  60             RTS         ;then exit
```

```
D60D   B9DBFE          LDA OFFSET,Y
D610:  2076D6          JSR HEDOFF  ;move head
D613:  EE9902          INC EPTR
D616:  20A6D6          JSR DOREC
D619:  C902            CMP #2      ;error code < 2?
D61B:  9008            BCC LD625
D61D:  AC9902          LDY EPTR
D620:  B9DBFE          LDA OFFSET,Y
D623:  D0DB            BNE LD600
D625:  AD9A02   LD625  LDA TOFF
D628:  2076D6          JSR HEDOFF  ;move head

FF84:  AA              TAX

up
to

FF90:  AA              TAX
```

This change reverses Difference #5 introduced with the previous revision 251968-02.

Difference #3: Remove test for ATN at end of selftest.

```
E780: 4C59EA    BOOT   JMP TSTATN  ;test for ATN
```

```
E780:  60       BOOT   NOP         ;! changed
E781:  EA              NOP         ;! changed
E781:  EA              NOP         ;! changed
```

This change reverses Difference #8 introduced with the previous ROM revision 251968-02. Note that the instruction at location $E780 is now a NOP (it used to be an RTS). The RTS at $E7A2 now has this function after needlessly executing a number of NOPs.

Difference #4: VIA pin PA0 is an output again.

```
               ;****************************************
               ;* RESET entry point                    *
               ;****************************************
EAA0:  78      DSKINT SEI         ;no interrupts
EAA1:  D8             CLD         ;clear decimal mode
EAA2:  A2FE           LDX #$FE    ;PA0 is an input,
                                  ;rest output
```

```
               ;****************************************
               ;* RESET entry point                    *
               ;****************************************
EAA0:  78      DSKINT SEI         ;no interrupts
EAA1:  D8             CLD         ;clear decimal mode
EAA2:  A2FF           LDX #$FF    ;! changed: port A is
                                  ;all outputs
```

Note: this simply reverses the setting of PA0 to an input, but does not take into account that PA0 is connected to ground on the 1541-II main board. As port A is not initialized properly (only the direction is set) the PA0 pin could be driven high, constituting in effect a short circuit and potentially destroying the internal VIA circuitry for PA0. This change should not have been made.

This change reverses Difference #1 introduced with ROM revision 251968-01.

Difference #5: Remove head recalibrate at the end of the selftest.

```
EBBA:  851D            STA WPSW+1  ;clear WP status drive 1
EBBC:  2063CB          JSR USRINT  ;initialize USER JMP
EBBF:  20FACE          JSR LRUINT  ;initialize LRU
EBC2:  206FFF          JSR PTCH10  ;perform a bump
EBC5:  A922            LDA #<DIAGOK
EBC7:  8565            STA VNMI

               ;****************************************
               ;* PTCH10:                              *
               ;*                                      *
               ;* Move head to track zero at end of    *
               ;* device initialization                *
               ;*                                      *
               ;****************************************
FF6F:  2059F2  PTCH10 JSR CNTINT  ;controller initialization
FF72:  A901           LDA #$01    ;command length
FF74:  8506           STA HDRS    ;is one byte
FF76:  A9C0           LDA #$C0    ;get bump command
FF78:  8500           STA JOBS    ;set it
FF7A:  60             RTS         ;and exit
```

```
EBBA:  851D            STA WPSW+1  ;clear WP status drive 1
EBBC:  2063CB          JSR USRINT  ;initialize USER JMP
EBBF:  20FACE          JSR LRUINT  ;initialize LRU
EBC2:  2059F2          JSR CNTINT  ;! changed: contrllr init
EBC5:  A922            LDA #<DIAGOK
EBC7:  8565            STA VNMI

FF6F:  AA              TAX
up
to

FF7A:  AA              TAX
```

This change reverses Difference #2 introduced with the previous ROM revision 251968-01.

Difference #6: Revert to the previous track zone table.

```
F33C:  A204     GOTU   LDX #4       ;set track, sector
F33E:  B132            LDA (HDRPNT),Y
F340:  8540            STA TRACC
F342:  DD52C0   LF342  CMP TRACKN-1,X       ;new zone table
F345:  CA              DEX

               ;***************************************
               ;* New track zone table                *
               ;***************************************
C053:  29       TRACKN .BY 41
C054:  1F              .BY 31
C055:  19              .BY 25
C056:  12              .BY 18
```

```
F33C:  A204     GOTU   LDX #4       ;set track, sector
F33E:  B132            LDA (HDRPNT),Y
F340:  8540            STA TRACC
F342:  DDD6FE   LF342  CMP TRKNUM-1,X       ;! changed
F345:  CA              DEX



C053:  AA              TAX
C054:  AA              TAX
C055:  AA              TAX
C056:  AA              TAX
```

This changes reverses Difference #15 introduced with ROM revision 251968-02.

Difference #7: Remove check for track 0 when stepping out.

```
FA2E:  A54A     DOSTEP LDA STEPS   ;get steps to do/direction
FA30:  1031            BPL STPIN   ;if stepping in, do that
FA32:  4C36FF   STPOUT JMP PATCH9  ;check for track 0
FA35:  EA              NOP         ;needed because
FA36:  EA              NOP         ;of
FA37:  EA              NOP         ;patch
FA38:  4C69FA   PPPPPP JMP STP     ;
FA3B:  A54A     SHORT  LDA STEPS   ;get # of steps to do

               ;****************************************
               ;* PATCH9:                              *
               ;*                                      *
               ;* Check if at track 0                  *
               ;*                                      *
               ;****************************************
FF36:  8A       PATCH9 TXA         ;save
FF37:  48              PHA         ;X
FF38:  98              TYA         ;and
FF39:  48              PHA         ;Y
FF3A:  A201            LDX #$01    ;get outer loop count
FF3C:  A064     PTCH91 LDY #100    ;get inner loop count
FF3E:  AD0F18   PTCH92 LDA POTA1   ;get port A
FF41:  CD0F18          CMP POTA1   ;if no change
FF44:  D01C            BNE PTCH93
FF46:  88              DEY         ;count down (inner loop)
FF47:  D0F5            BNE PTCH92  ;if inner not done, loop
FF49:  CA              DEX         ;else adapt outer loop
FF4A:  D0F0            BNE PTCH91  ;if not finished, loop
FF4C:  2901            AND #$01    ;else get port data
FF4E:  F012            BEQ PTCH93  ;if not at track 0
FF50:  AD001C          LDA DSKCNT  ;get controller phase
FF53:  2903            AND #$03    ;bits
FF55:  D00B            BNE PTCH93  ;if at phase A
FF57:  68              PLA         ;exit
FF58:  A8              TAY         ;restoring
FF59:  68              PLA         ;Y
FF5A:  AA              TAX         ;and X
FF5B:  A900            LDA #0      ;clear
FF5D:  854A            STA STEPS   ;number of steps to do
FF5F:  4CBEFA          JMP END33   ;and return to step code
FF62:  68       PTCH93 PLA         ;not at phase A, restore
FF63:  A8              TAY         ;Y
FF64:  68              PLA         ;and
FF65:  AA              TAX         ;X
FF66:  E64A            INC STEPS   ;count steps to do
FF68:  AE001C          LDX DSKCNT  ;get motor phase
FF6B:  CA              DEX         ;update it
FF6C:  4C38FA          JMP PPPPPP  ;and return
```

```
FA2E:  A54A     DOSTEP LDA STEPS   ;get steps to do/direction
FA30:  1031            BPL STPIN   ;if stepping in, do that
FA32:  E64A     STPOUT INC STEPS   ;! changed: steps to do
FA34:  AE001C          LDX DSKCNT  ;get phase
FA37:  CA              DEX         ;update it
FA38:  4C69FA          JMP STP     ;
FA3B:  A54A     SHORT  LDA STEPS   ;get # of steps to do

FF36:  AA               TAX

up
to

FF6E:  AA               TAX
```

This change reverses Difference #4 introduced with ROM revision 251968-01.

Difference #8: Remove test for minimum gap2 from 4 to 2.

```
FC17:  A200           LDX #$00
FC19:  38      DS20   SEC
FC1A:  E543           SBC SECTR
FC1C:  B003           BCS DS22
FC1E:  88             DEY
FC1F:  3003           BMI DS30
FC21:  E8      DS22   INX
FC22:  D0F5           BNE DS20
FC24:  8E2606  DS30   STX DTRCK
FC27:  E002           CPX #2      ;gap 2
FC29:  B005           BCS DS32
```

```
FC17:  A200           LDX #$00
FC19:  38      DS20   SEC
FC1A:  E543           SBC SECTR
FC1C:  B003           BCS DS22
FC1E:  88             DEY
FC1F:  3003           BMI DS30
FC21:  E8      DS22   INX
FC22:  D0F5           BNE DS20
FC24:  8E2606  DS30   STX DTRCK
FC27:  E002           CPX #4      ;! changed: gap2 is 4
FC29:  B005           BCS DS32
```

This change reverses Difference #16 introduced with ROM revision 251968-02.

This concludes the differences between the 1541 901229-06 and 251968-01 ROMs.

1541-II, 355649-01 ROM

This 1541-II ROM version is reported by some sources as being present in some 1541-II drives with Newtronics mechanisms. Analysis of the contents reveals the following:

The part $C000-$DFFF is equal to the ROM 325302-01.

The part $E000-$FFFF has the following features compared to ROM 251968-03:

1. No copyright statement;
2. NODRV problems are fixed;
3. CLEAR routine is the first version;
4. Check for reportable errors after stealing a buffer is absent;
5. Fix to improve initializing variables when opening a file is absent;
6. HEXDEC routine has not been fixed;
7. TSTATN does not test for a 2nd ATN while processing current ATN;
8. Reset routine executed with interrupts enabled;
9. Format flag is not set before formatting a disk;
10. Disk full bug still present;
11. Gap1 has the new value of 9.

These features are a match with ROM 901229-05.

The ROM is therefore simply the contents of the older ROMs 325302-01 and 901229-05 combined in one 16 kbyte device, and brings nothing new (in fact it is a step back compared to ROM 251968-03).

Code differences compared to the ROMs 325302-01 & 901229-05

ROMs involved: 325302-01 + 901229-05 and 335640-01

There are no differences; all bytes are the same.

Remark: in all 1541 ROMs there is no code specific for a particular mechanism manufacturer; these differences were solved in the hardware. This ROM revision is therefore not specifically geared towards Newtronics drives.
