Final Cartridge III flash utility

User manual

An utility to dump and flash Final Cartridge III cartridges

by Daniël Mantione

March 2026

Introduction

This manual describes the Final Cartridge III flash utility. The utility has been written to accompany my Final Cartridge III 101% design, which is a clone of the classic Final Cartridge III hardware, but with flash-ROM rather than read-only memory onboard. This allows the end-user to install and upgrade the his preferred firmware on the cartridge.

The utility can dump the ROM from any Final Cartridge III compatible cartridge, but can only write to the flash memory on a Final Cartridge III 101%. Flashing and dumping is performed from/to files on storage media attached to your Commodore computer.

The Commodore 64 and Commodore 128 will from now on be designated as C64 and C128 in this manual. Final Cartridge III will from now on designated FC3 and Final Cartridge III 101% as FC3101%.

Programming compatibility

The circuit on the FC3101% is well designed and more similar to that of the EasyFlash cartridge than to the minimal circuit on for example Gmod2 cartridges. Therefore flashing the FC3101% should work on any C64 mainboard including the C128, Ultimate64 and Mega65 with C64 core. If you can flash EasyFlash cartridges, you should be able to flash the FC3101%.

Startup and exit

The flash utility can be loaded from disk, or can flash itself into the cartridge. If the tool is flashed into the cartridge it will autostart on reset.

The flash utility does a small check on the bank switch register to check if a cartridge is present and test its essential functions.

If this check fails, this is not considered an error condition, but the tool will try to continue. You won’t be able to flash anything, but perhaps you can still dump the ROM of an FC3 cartridge.

On exit, the computer returns to BASIC or, if started from cartridge, the computer continues the normal boot process. The tool is a very well behaving application, that doesn't interfere with variables in the computer's memory used by the operating sytem. After exit, the computer will be a state where BASIC is just as functional as before start. The tool also doesn' t mess up its own memory contents: You can exit and run again, no need to reload from disk.

The flash utility uses the high level KERNAL API for the user interface and the low level KERNAL API for device operations. This means that the tool can take full advantage of custom KERNALs, JiffyDOS, SpeedDOS or DolphinDOS will be no problem.

Because a 1541 drive is slow by default, a KERNAL based fastloader is even recommended, but the tool will of course still work with a stock C64 and 1541, it will just be a bit slow... but you can use its UI- command to speed it up a bit, the tool has special code to support it and will disable the screen during TALK operations.

FC3 deactivation

If you start your computer with the FC3 inserted, it hooks itself into the BASIC and KERNAL vectors which point into the ROM memory of the cartridge. The flash utility writes to your Final Cartridge III ROM memory. Therefore it is not a good idea that the FC3 is active while the program is running; the computer could crash.

Therefore, the flash program disables the FC3 when you start it. The procedure to deactivate the FC3 used is the same the KILL command uses, with the difference is that the KILL command disables both uninstalls the FC3 firmware, as well as deacticates the FC3 hardware. The flash program uninstalls the FC3 firmware, but leaves the cartridge hardware enabled (because it is needed to do the flashing).

DOS command line

After the program has been started, you will end up in a DOS like command line interface. It will display a prompt with the current device number:

```
8>
```

... and you can type Commodore DOS commands that will be sent to the device, i.e.

```
8> n:newdisk,aa
```

... to format a disk.

To switch to a different device, type the device number:

```
8> 10
10>
```

To view a directory, type $:

```
8> $
```

The DOS command line interface has been chosen to allow you to have full control of your storage devices while working with the tool. A menu system would have always been incomplete.

In order to do the actual work, the tool supports several special commands, of which you can get a short description with the "help" command:

Reading or programming the entire flash memory

The original Final Cartridge III had 64KB of firmware. The cartridge can be easily extended to 256KB and the FC3101% therefore has 256KB of flash ROM. Depending on the size of the firmware that you want to flash, the firmware might be too large for classic floppy disks. For example, a 1541 floppy drive can store only 170KB of data on a floppy disk side. In order to be able to read or program all of the flash memory in one go, you will need a C64 storage medium that can contain files of 256KB in size, or 1033 blocks, in C64 terminology.

These storage media do exist. For example the Commodore 1571 can store 340KB on a 5.25” diskette and the Commodore 1581 can store 720KB on a 3.5" diskette. The popular SD2IEC also has no problems with 256KB files and the 1541 Ultimate and Ultimate64 support an IEC device that provides direct access to USB sticks that can contain 256KB files as well. Therefore if you have one of these storage media, you can read or program the flash memory in one go.

If you would like to dump a cartridge into the file mydump with type SEQ on device 10, you can do this with the following commands.

Switch to device 10:

```
10
```

Set the file name:

```
file:mydump
```

Dump the cartridge:

```
fdump
```

When the process is completed there will be a file called "mydump.seq" on device 10 and it will be 256KB in size. Note that Commodore storage devices will normally refuse to overwrite files, the file should not yet exist before you start.

If you have a file mycartridge with type SEQ on device 10, and would like to flash it, you can do it with the following commands.

Switch to device 10:

```
10
```

Set the file name:

```
file:mycartridge
```

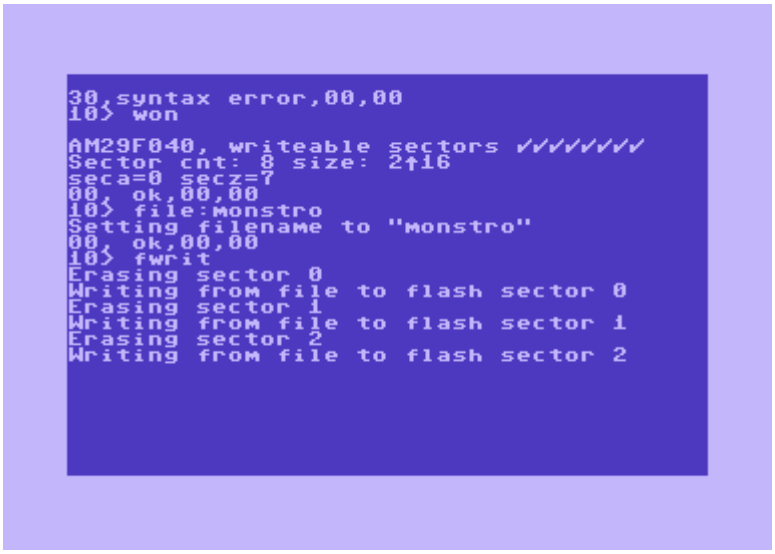
Enable the write commands:

```
won
```

Flash the cartridge:

```
fwrit
```

Programming will start:



After a while the flash memory will have been programmed with your file.

Files on Commodore storage media are sequential-only, they must be read from start to finish and you only know the exact size of the file once you have read all of it. Because of this, there is no way for the tool to know upfront if the file you are trying to flash has the correct size and therefore, the tool is not able to check for these kinds of errors. If the file isn't exactly 256KB in size, flashing will start normally, but if the tool receives an end-of-file, flashing will fail with an end-of-file eror and part of the flash memory written. If the file is larger than 256KB, flashing will complete succesfully and the first 256KB of the file will be written to flash.

Reading or programming the flash memory partially

In case you don't have a CBM storage medium available that can carry 256KB files, you can still program the flash memory from your Commodore 64/128, but you have to do it in multiple passes. The flash program allows you to flash only part of the memory of the FC3101% cartridge, you can flash invidual sectors of the flash ROM. The sector size can differ for different types of flash ROM. For example, the AM29F002B has 64KB sectors, while the SST39SF020 has only 4KB sectors. This means that if your cartridge contains an AM29F002B, you can program multiples of 64KB while with an SST39SF020, you can program multiples of 4KB.

The flash program identifies sectors with their number. So on an AM29F002B the sectors are numbered 0..3, while on an SST39F040, they are numbered 0..63.

You are completely free to decide how many sectors your would like to program in one go. For example, if you have a 1541 floppy drive, you could decide to split the 256KB ROM image into 2 files of 128KB in size. Each of these files will fit on a 1541 floppy disk side, and you can go four 2 programming operations.

(A good way to split files on a PC is the "split" command that is available on any Linux system.)

The manual will now explain the procedure how to flash using the first example, that is, you have split the 256KB ROM image into 2 128KB files stored onto 2 1541 floppy disk sides. The 1541 is connected to device 8. The files on the floppies have the filenames "part1", "part2".

Our cartridge is assumed to contain an AM29F002B.

Switch to device 8:

```
8
```

Insert the first floppy disk. Set the file name:

```
file:part1
```

Enable the write commands:

```
won
```

The first sector to flash is sector 0:

```
seca:0
```

The last sector to flash is sector 1:

```
secz:1
```

Start the flash operation.

```
fwrit
```

Insert the next floppy disk (or flip it). The first sector to flash is sector 2:

```
seca:2
```

The last sector to flash is sector 3:

```
secz:3
```

Start the flash operation again.

```
fwrit
```

After completion, your cartridge is ready.

Just like when writing the flash memory in one go, there is no checking beforehand on the file size. It is your own responsibility that the files have the correct length.

Writing the flash tool to cartridge

The flash utility can flash itself to cartridge. This can be done with the commands:

```
won
twrit
```

After rebooting the computer, it will start with the flash tool from cartridge.

Writing the flash tool to disk

If you have received a cartridge with the flash tool pre-installed and would like to write the tool to disk, you can do this as follows. Select the correct device:

```
i0
```

Set the filename:

```
file:fc3flasher
```

Write the tool to disk:

```
tdump
```

Command reference

$ - Directory

This command displays the directory on the active device. Wildcards are supported, so for example "$a" displays all files starting with an A.

file - Set filename

With this command you tell the flash tool which file you would like to read or write from/to. All dump and flash write commands use this filename. For example:

```
file:mycartridge.bin
```

... tells the tool to read/write from/to "mycartridge.bin". Wildcards are supported and immedeately expanded, for example "file:m*".

seca - Set starting sector

If you wish to program part of the flash ROM, you can set the first sector of the flash ROM to be programmed with this command. For example:

```
seca:2
```

... to start flashing at sector 2.

secz - Set end sector

If you wish to program part of the flash ROM, you can set the last sector of the flash ROM to be programmed with this command. For example:

```
secz:3
```

... to stop after flashing sector 3.

exit - Exit to BASIC

Exit the flash tool. If you did start the tool from BASIC you will return to BASIC. If you did boot the tool from cartridge, the boot process of the computer will continue.

fdump - Dump flash to file

This command dumps the flash ROM contents to the file set by the "file" command. Dumping will start at the flash ROM sector specified with the "seca" command and stop after the flash ROM sector specified with the "secz" command.

fwrit - Write flash from file

This command writes of contents of the file set by the "file" command to the flash ROM. Flashing will start at the flash ROM sector specified with the "seca" command and stop after the flash ROM sector specified with the "secz" command.

tdump - Dump flash tool to file

This command will write the flash tool itself into the file specified with the "file" command. This is useful if you have the flash tool in the cartridge ROM, are going to overwrite it and want to keep the tool.

twrit - Write flash tool to flash

This command will write the flash tool itself into the flash ROM. After completion, the computer will autostart the flash tool on reset.

won - Writing on

This command enables the commands that write to flash memory and eeprom. When you enter the command, the tool tries to detect the flash memory chip. Further, it tests whether repeated write commands arrive at the flash ROM reliably.

The tool first tries to detect 5V flash ROMs without applying 12V. It also checks for possible unsupported 5V flash ROMs. If no 5V flash ROM is detected, the tool applies 12V and checks for the presence of 12V flash ROMs.

Only if a supported chip is detected and the flash ROM receives commands reliably, the write functionality is enabled. The reason why you must explicitely enable the write commands is that this way, the tool is still usefull for dumping cartridges in case of write incompatibilities.

File formats

The flash utility works with binary ROM images, the .crt file format is not supported. You can convert between file formats on the PC with the cartconv utility. For example:

```
~> cartconv -i cartridge.crt -o cartridge.bin
~> cartconv -i cartridge.bin -t fc3 -o cartridge.crt
```

When stored on a C64 storage medium, the file type should be SEQ.

1541 UI- mode

The 1541 and many 1541 compatible devices have a UI- command that will make the device a bit faster, but it will then no longer work correctly with the screen enabled. If you type the UI- command, the tool will recognize this and disable the screen during TALK operations with the device, so if you don't have a KERNAL speedloader, you can still benefit from slightly faster disk I/O.

In order to avoid that the tool needs to keep track which device is in UI- mode and which device isn't, you will no longer be able to switch to a different device if you activate UI-. Enter UI+ before switching to a different device. Due to code simplicity, there will be no error messages, instead requests to switch to different devices are no longer recognized as commands.

UI- is still quite slow and to be used when nothing better is available: A KERNAL speedloader is the proper way to get more performance out of the CBM serial bus.

Supported flash ROMs

Your FC3 101% will contain one of the many supported flash ROMs. The FC3101% is compatible with 3 types of flash ROMs:

- 256KB 5V flash ROMs
- 512KB 5V flash ROMs
- 256KB flash ROMs

Flash memory needs to be erased before it can be written to. Flash ROMs can be divided in sectors, that can easy be individually erased. Therefore on a flash ROM with multiple sectors, you don’t need to write all the flash memory at once.

The flash tool understands the concept of sectors and allows you to write the flash memory partially.

256KB 5V flash ROMs

256KB flash ROMs are kind weird: Very few manufacturers have a “normal” 256KB flash ROM. For example AMD made a 512KB AM29F040, but not a 256KB AM29F020. Instead there exist so called “non-uniform sector” flash ROMs, there exists for example an AM29F002B. These flash ROMs have sectors with different sizes, allowing you to create a boot sector at the bottom or top of flash memory. A boot sector is completely useless for the purpose of the FC3101%.

The flash tool supports non-uniform sector flash ROMs, but makes them appear as uniform flash ROM. For example the AM29F002B appears as having 4 sectors of 64KB.

Many 256KB flash ROMs have a write protect mechanism that can be overridden when applying 12V. The flash utility assumes that if you are using the utility, you really want to write to the flash memory and applies 12V if it detects any write protected sector.

At this time the tool supports the following flash ROMs:

```
A29002T
A29002U
AM29F002T
AM29F002B
AT49F002
AT49F002T
AT49F020
EN29F002T
EN29F002B
HY29F002T
HY29F002B
M29F002T
M29F002B
MBM29F002T
MBM29F002B
MX29F002T
MX29F002B
SST39SF020
SST39VF020
```

```
W39L020
W49F002B
W49F002U
```

A few of the above flash ROMs are actually 3.3V flash ROMs rather than 5V. The flash utility supports their algorithm, but the FC3101% has no voltage conversion hardware for 3.3V flash ROMs on board.

512KB 5V flash ROMs

5V flash ROMs generally have uniform sectors and therefore less complex than 256KB flash ROMs. It can be atractive to use 512KB flash ROMs and use only 256KB. The last address line uses the pin that 256KB flash ROMs use to apply 12V. Therefore, if the tool detects a 512KB flash ROM, it will not switch on the 12V generator.

At this time the tool supports the following flash ROMs:

```
AM29F040 (including clones AS29F040 and TMS29F040)
A29040B (including clone AS29CF040)
EN29F040
HY29F040
M29F040
MBM29F040
MX29F040
SST39SF040
SST39VF040
W39L040
```

Again, a few of the above flash ROMs are actually 3.3V flash ROMs rather than 5V. The flash utility supports their algorithm, but the FC3101% has no voltage conversion hardware for 3.3V flash ROMs on board.
Flash ROMs that have sector protect functionality must be unprotected. The tool will detect protected sectors and refuse to program those, but cannot unprotect these sectors itself.

12V 256KB flash ROMs

Unlike 5V 256KB flash ROMs, their 12V counterparts are uniform and are not divided into sectors. Therefore the flash tool makes a 12V flash ROM appear as a single sector of 256KB in size.

Exception are the MX28F002B/MX28F002T, these flash ROMs have non-uniform sectors and will appear as two sectors of 128KB in size

At this time, the tool supports the following flash ROMs:

```
AM28F020
CAT28F020
i28F020
IS28F020
M28F201
MX28F002T
MX28F002B
```

Frequently encountered problems

Error message 62, file not found,00,00

If you encounter this error message, you really did specify a file that does not exists. Tips:

- Look the directory with the $ command.
- Look at file extensions, if the file is called firmware.bin, you really have to specify file:firmware.bin
- Try using a wildcard. If there is a file called firmware.bin on your storage device, you might be able to select it with file:firm*

Error message 64, file type mismatch,00,00

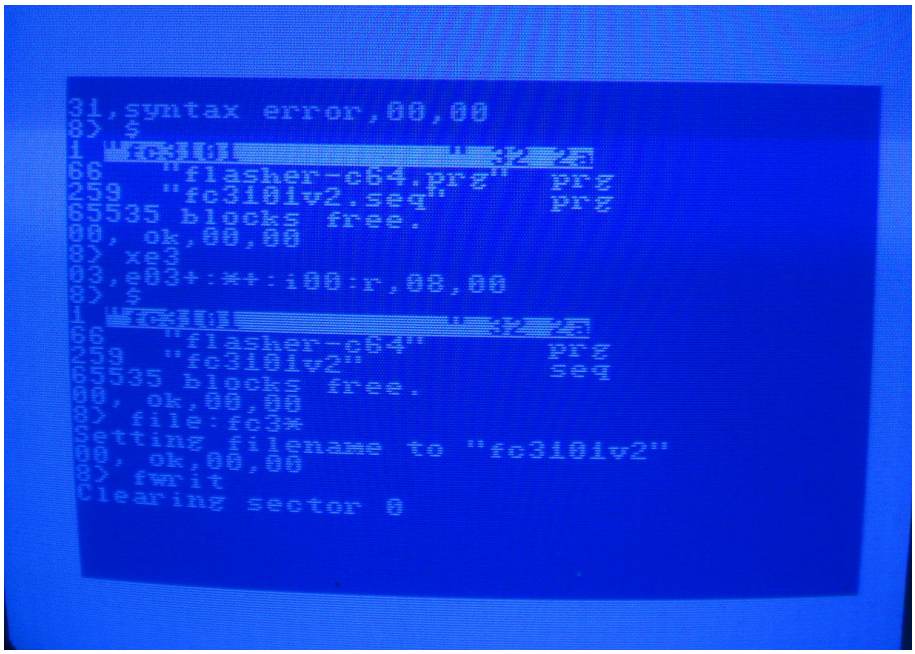
If you encounter this error message the Commodore file type of your file is not SEQ. Change the file type to SEQ to

solve the problem.

Using an SD2IEC to flash

In case you are flashing from an SD2IEC and copy the firmware directory onto an SD card (you do not use a D64, D71 or D81 image), you will run into an obstacle that the SD2IEC by default makes all files PRG. The flash utility wants to see a file of type SEQ. If you copy a file called “FC3101V2.SEQ” onto the SD card, the Commodore 64 will see a file “FC3101V2.SEQ” with type PRG and therefore it is not possible to flash the file.

The SD2IEC supports a command “XE3” to make the SD2IEC look at file extensions to determine the C64 file type. If you issue the command “XE3”, the Commodore 64 will see a file called “FC3101V2” with type SEQ.



After issuing the “XE3” command, you should have no problems flashing the firmware using an SD2IEC.

Creating an image with the firmware

If you see a need to create a D64, D71 or D81 image with the firmware and flash utility inside, you can do this with the c1541 utility from vice as follows:

```
linux> c1541
c1541 #8> format fc3firmware,f3 d71 fc3firmware.d71 8
c1541 #8> write fc3_101v2.bin fc3101v2,s
c1541 #8> write flasher-c64.prg fc3flasher
exit
```

This creates an image fc3firmware.d71 and stores the files “fc3_101v2.bin” and “flasher-c64.prg” into the image as “fc3101v2” and “fc3flasher”. Please note the use of “,s” to create a file with type SEQ.

You can then use the image fc3firmware.d71 with your favorite storage solution to access the files on a C64.

Flash chip not detected

If the utility cannot detect flash chip after typing “won”, there really is a problem. Please report the problem so will get knowledge about these issues and can take steps to make the hardware better.

Possible causes of this error:

- The 12V circuit on the cartridge is not working for some reason

- There is a bus timing issue. The FC3101% requires the address bus of the C64 tot be stable 120ns after PHI2 rise and the databus should be stable up to 35ns after PHI2 falls. This timing is quite relaxed and should be compatible with all classic C64s and C128s. Some “modern” C64s might not be compatible enough with the original C64.
- There is a bad contact, either the cartridge might not be making proper contact with the cartridge port on some pins, or the flash ROM inside the PLCC socket might not be making proper contact.

Things you can try:

- Check the 5V voltage of your C64. In case of severe under voltage, the cartridge hardware might not be able to generate 12V.
- Try a different Commodore 64.
- If you are using a Mega65: Use the C64 core instead of the native core.
- If you are using an Ultimate64: Try playing with the Cartridge Bus Mode.

Empty blue screen

If you see an empty blue screen, the C64 is trying to start the cartridge, but the cartridge startup code crashes. This may have the following causes:

- Bad firmware on cartridge. If the firmware on the cartridge is invalid, you will get this problem. Re-install the firmware on the cartridge.
- Bad contact. If the cartridge does not make proper contact inside the cartridge slot of your C64, you will get this problem. Try to resolve the bad contact.

If you have a Commodore 128, you can manually inspect the firmware on the cartridge with the built-in monitor. Set the switch on the cartridge into “PROG” mode. Then enter the monitor with Commodore 128 “MON” command, and view the ROM contents with “M88000”. Check if the output is stable and if you can see the CBM80 signature.