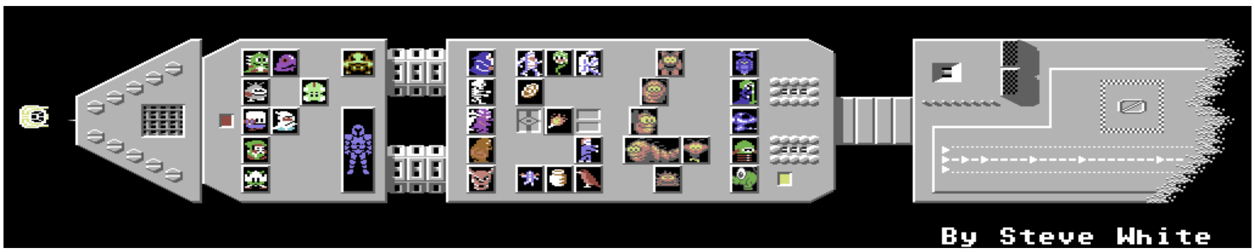


Pi1541 User Manual

Revised July 2021 for version 1.24

Table of Contents

Introduction

Pi1541 is an emulator package for the Commodore 1541 and 1581 drives that runs on various models of Raspberry Pi. Pi1541 provides real-time, cycle exact emulation of drive's 6502 CPU and 6522 CIA chips. It provides a SD-card solution for Commodore 8-bit computers such as the 64, 128, Vic20, 16, and Plus/4.

Images supported include D64, G64, NIB, NBZ (read only), D81, T64 (read only), and individual PRG files (read only).

Raspberry devices supported include 3A, 3B, 3B+, 0/1 (limited features). The Raspberry Pi 4 is not supported.

How does Pi1541 differ from SD2IEC? Unlike SD2IEC, Pi1541 emulates a 6502 and the two 6522s. Any code it is asked to run is run in a cycle exact way. SD2IEC supports a limited set of fast loaders by attempting to guess the fast loader from the code sent to it. SD2IEC will not, and cannot, execute the code, it just simulates the communication protocols. As a consequence only a small amount of popular fast loaders are supported. As Pi1541 can execute code on its emulated 6502 core it supports a vast range of fast loaders (games and demo scene) even copy protected originals.

NOTE: Always use copies of your images or backup your images in case they get accidentally modified.

NOTE: Some Demos can crash some C64s that are susceptible to the VSP bug. This is not caused by Pi1541.

Background from Steven White

Like most people I was a little disappointed in the SD2IEC offerings (being very hit and miss with their compatibility) and the hard to order, FPGA solutions were out of my budget. Inspired by projects such as Peter Edwards' Tapuino and David Banks' PiTubeDirect I set about implementing a 1541 on a Raspberry Pi. My goal was to make a highly compatible, inexpensive SD card solution for all Commodore 8 bit machines.

The Commodore 1541 disk drive is a computer itself. It consists of a CPU, ROM, RAM, IO devices and the drive mechanics. Due to the popularity of Commodore machines and the subsequent proliferation of software created for them, all kinds of exotic fast loaders and copy protection schemes were developed. As a consequence an inexpensive, cycle exact, SD card solution has taken longer than other systems to come to fruition.

When I started out, I had no way of selecting disk images using the Pi's screen and keyboard. I used NBLA000's excellent CBMFileBrowser that all SD2IEC users would be familiar with. CBMFileBrowser runs on the target Commodore computer and allows you browse and select diskimages using that target computer. In order to do this I had to implement minimal SD2IEC commands. This way, folders can be navigated and disk images selected as, at this level the Pi behaves like a SD2IEC device. But once a disk image has been selected the Pi drops down into full cycle exact emulation and compatibility is near 100%. Subsequently I have implemented file browsing and selection using the Pi's screen and keyboard. I was going to remove SD2IEC support but others have convinced me to leave it in. Be warned though; the bare minimum functionality was implemented and was initially only used for testing. Going forwards I hope others will implement the full functionality.

If you would like to support me, any donation will be greatly appreciated. It keeps me going and makes the project possible. I will endeavour to keep enhancing the project. Others are profiting from my work.
All I ask is that if you are not going to build it yourself and are willing to pay others then please consider the value of my efforts and show me some support, thanks.

Paypal: pi1541contact@gmail.com

Patreon: https://patreon.com/user?u=11191060

Hardware

You can choose to use a Pi Hat, or a cable. For cables, you have two options: A for a single cable, or B for a full serial implementation. Pi Hat’s are circuit boards which make it easy to connect all of the proper components.

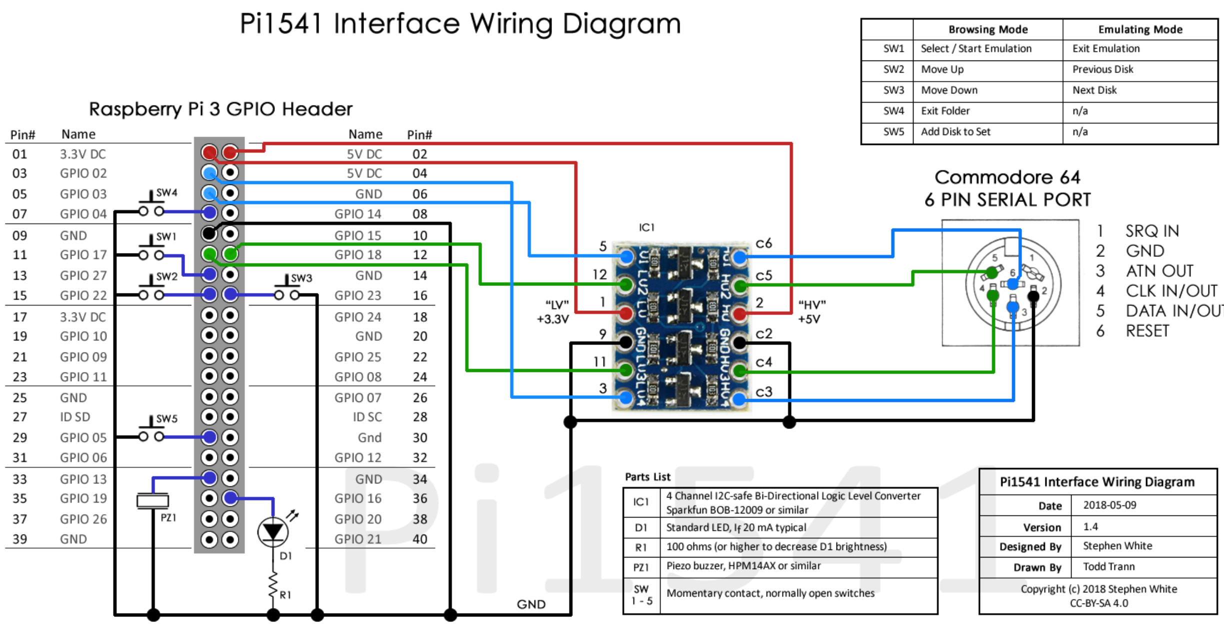
Commodore computer's can produce 5V on their serial ports. Raspberry Pi's can only tolerate 3.3V on their GPIO pins. Unfortunately this complicates the cable design. A I2C Bi-Directional Logic Level Converter is required to convert the voltages so each device is only exposed to the voltages it requires.

The switches are entirely optional. If you are going to connect a keyboard to your Pi then you don't need them. The Piezo buzzer is also optional (in fact to enable it you need to configure it in the options.txt file explained below). At the moment only Piezo buzzers without generators are supported. If you don't connect and configure a Piezo you can still get the head stepping sounds via the Pi's headphone socket. Again the activity LED is optional. It has been provided for those who want an external LED for their custom Pi cases.

For display you can use HDMI or composite video if you have a Raspberry Pi 3, or you can use a OLED screen on a Pi Hat. HDMI/composite video is not supported yet on Raspberry Pi 0/1. If you use composite video, you will need to purchase a composite video cable for Raspberry Pi.

Option A (single device on serial bus)

This is the simplest version of the hardware. It will work if the Pi1541 is the only device on your serial bus (i.e., you don't have other drives, printers, etc.)



Power the Pi with a suitable external power supply. The I2C Bi-Directional Logic Level Converter gets its 5V from Pi pin 2 and 3.3V from PI pin 1. Both side GNDs and the C64's GND serial port pin 2 are connected to the PI pin 9.

C64's ATN serial port pin 3 is connected to a free 5v side pin of the level converter and the 3.3v side is then connected to the Pi's pin 3 (GPIO02).

C64's CLOCK serial port pin 4 is connected to a free 5v side pin of the level converter and the 3.3v side is then connected to the Pi's pin 11 (GPIO17).

C64's DATA serial port pin 5 is connected to a free 5v side pin of the level converter and the 3.3v side is then connected to the Pi's pin 12 (GPIO18).

C64's RESET serial port pin 6 is connected to a free 5v side pin of the level converter and the 3.3v side is then connected to the Pi's pin 5 (GPIO03).

The buttons/switches, piezo buzzer, and LED are all optional.

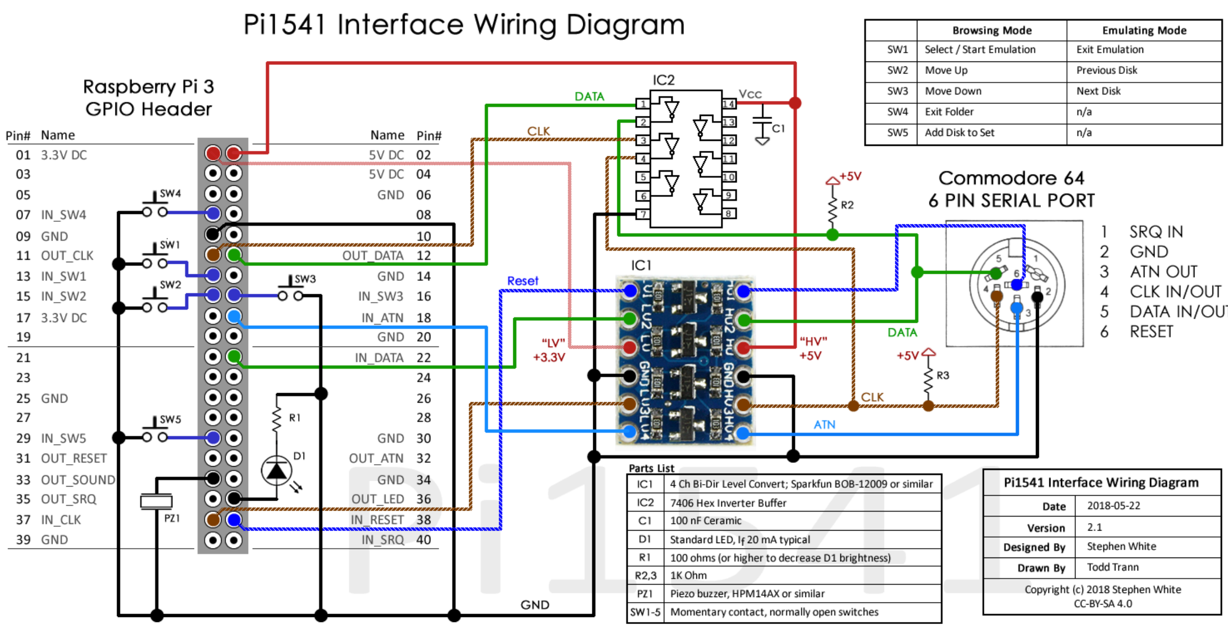
Note: "The FET-method to adapt a 3.3V open-drain bus to a 5V-open-drain-bus requires the RPi to sink all the current that the 5V side sources. Not a big deal if it's the only drive connected, but the IEC bus allows connecting multiple drives, printers and plotters at the same time. Using the FET method, all that current of each pull-up resistor needs to be sinked by the RPi pins, which will eventually break."

Kris Sekula has made an awesome write up showing just how easy it is to make a cable that will work:
https://mygeekyhobby.wordpress.com/2018/08/12/pi1541-in-30min-with-almost-no-soldering

Option B (multiple devices on serial bus)

This option uses a 7406 just like all Commodore equipment and can therefore support more devices connected to the serial bus.

Note: you can also use a 7405, 7416, 74LS05, 74LS06 or a 74LS16



Again, the buttons/switches, piezo buzzer and LED are all optional.

If you build option B you will need to place the line "splitIECLines = 1" in the options.txt file found in the root folder of the SD card.

If building option B then please use genuine branded parts. People are reporting that the use of no named Chinese hex inverters can lead to some disk images not working. Ghosts'n Goblins Arcade can be used to test the capability of your hex inverter. If the title screen is corrupt then please substitute the hex inverter IC for a genuine branded version.

Pin Out

3.3 VDC		**1**	**2**		5 VDC
SDA1 (I2C BUS 1)	GPIO 2	**3**	**4**		5 VDC
SDC1 (I2C BUS 1)	GPIO 3	**5**	**6**		GND
IN SW4	GPIO 4	**7**	**8**	GPIO 14	-
GND		**9**	**10**	GPIO 15	-
OUT CLK (IEC BUS)	GPIO 17	**11**	**12**	GPIO 18	OUT DATA (IEC BUS)
IN SW1/ROTARY SEL	GPIO 27	**13**	**14**		GND
IN SW2/ROTARY UP	GPIO 22	**15**	**16**	GPIO 23	IN SW3/ROTARY DOWN
3.3 VDC		**17**	**18**	GPIO 24	IN ATTN (IEC BUS) 3.3V
-	GPIO 10	**19**	**20**		GND
-	GPIO 9	**21**	**22**	GPIO 25	IN DATA (IEC BUS) 3.3V
-	GPIO 11	**23**	**24**	GPIO 8	-
GND		**25**	**26**	GPIO 7	-
SDA0 (I2C BUS 0)	GPIO 0	**27**	**28**	GPIO 1	SCL0 (I2C BUS 0)
IN SW5	GPIO 5	**29**	**30**		GND
OUT RESET	GPIO 6	**31**	**32**	GPIO 12	OUT ATN
OUT SOUND	GPIO 13	**33**	**34**		GND
-	GPIO 19	**35**	**36**	GPIO 16	OUT LED ACTIVITY
IN CLK (IEC BUS)	GPIO 26	**37**	**38**	GPIO 20	IN RESET (IEC BUS) 3.3V
GND		**39**	**40**	GPIO 21	-

Buttons

You can connect momentary contact buttons that can aid in the use of the Pi1541. These are optional for Pi 3 (not required if you are going to use a USB keyboard) and are required for Pi 0/1.

The buttons are connected like so:

- SW1: Reset (or Select) Pi's pin 13 (GPIO27)
- SW2: Previous Disk (or Move Up) Pi's pin 15 (GPIO22)
- SW3: Next Disk (or Move Down) Pi's pin 16 (GPIO23)
- SW4: Exit Folder Pi's pin 7 (GPIO4)
- SW5: Insert Disk Pi's pin 29 (GPIO5)

The buttons are active low so that the other side of the button is connected to ground (Internal pullups are used so you don't need resistors).

Pi1541 Hat

Many vendors sell Pi1541 Hats preassembled, as parts kits, or just as PCBs.

A good list of projects and PCBs is at:
https://8bithardware.wixsite.com/website/product-page/pi1541-hat

You can make your own Pi1541 Hat using some of these parts:

- "Raspberry Pi 3 Model 3 (or 3B+) with suitable power supply" x1
- "Female S Terminal 6 pin din PCB" x2
- "2x20 pin PCB female header" x1 (2.54mm pitch)
- "I2C Bi-Directional Logic Level Converter" x1 (4 or 8 channels)
- "Micro SD card 8-32GB" x1
- "5cm x 7cm Perfboard" x1
- "6x6x4.5mm 4 Pin DIP PCB Momentary Switch" x5 (optional)

Rotary Encoder

You can include a KY-040 Rotary Encoder for browser menu up/down and select. Connect as follows:

15	GPIO 22	Menu Up	Encoder Pin A (CLK)
16	GPIO 23	Menu Down	Encoder Pin B (DT)
13	GPIO 27	Enter/Select	Encoder Push Button (SW)

Enable it in the options as RotaryEncoderEnable=1.

NOTE: Using an encoder is incompatible with button remapping. You must use the default values of Enter=1, Up=2, Down=3, Back=4, and Insert=5.

NOTE: This has only been tested using a Raspberry Pi 3. Please see dmRotary.h for full implementation details.

Preparing the SD Card

1. The micro SD Card should ideally be a 4 or 8 GB card. If you use larger, you must partition it at 8 GB. The card should be formatted as FAT32.

2. Download the Raspberry PI Firmware
   https://github.com/raspberrypi/firmware/archive/1.20180919.zip

3. Copy bootcode.bin, fixup.dat, and start.elf from the Raspberry Pi Firmware into the root of the SD card.

4. Download the SD card file from the website and extract the files onto the SD card. https://cbm-pi1541.firebaseapp.com/Pi1541.zip

5. Download and copy the latest kernel file (the Pi1541 software) from the Pi1541 site as appropriate for Pi 3, Pi 2, or Pi 0/1.

6. Copy a valid 1541 (and optionally, 1581) ROM file into the root folder. You can specify the name in the options.txt file but it defaults to d1541.rom, dos1541, d1541II. Jiffy.bin for 1541, and 1581-rom.318045-02.bin for 1581

   You can obtain ROMS from the VICE emulator (https://vice-emu.sourceforge.io/) and from http://www.zimmers.net/anonftp/pub/cbm/firmware/drives/new/index.html

   If you wish to use JiffyDOS, you can purchase valid ROMs for 1541 and 1581 at Retro Innovations:
   http://store.go4retro.com/jiffydos-1581-dos-rom-overlay-image/
   http://store.go4retro.com/jiffydos-1541-dos-rom-overlay-image/

7. Optionally, copy of a file that contains the CBM font ROM into the root folder, with the filename chargen. You can obtain it from the VICE emulator (eg vice-3.1\C64\chargen)

8. Copy your D64, G64, etc. images into the 1541 folder.

Configuring config.txt

Raspberry Pi 0

Refer to the video by Lagom Effort Electronics Channel for a walk-through: https://youtu.be/8Gs1daxDKa8

In config.txt, set the following settings:

```
kernel_address=0x1f00000
arm_freq=1100
over_voltage=8
sdram_freq=500
sdram_over_voltage=2
force_turbo=1
boot_delay=1
```

Raspberry Pi 1

Refer to the video by Lagom Effort Electronics Channel for a walk-through: https://youtu.be/8Gs1daxDKa8

In config.txt, set the following settings:

```
kernel_address=0x1f00000
arm_freq=1100
over_voltage=8
sdram_freq=500
sdram_over_voltage=2
core_freq=500
temp_limit=75
force_turbo=1
boot_delay=1
```

Raspberry Pi 3

In config.txt, set the following settings:

```
kernel_address=0x1f00000
force_turbo=1
```

Configuring Options.txt

Modify the options.txt file as appropriate for your hardware and setup. Configuration is done in the form of “key = value”. Key names are case-insensitive.

Key	Default	Example	Description
HARDWARE OPTIONS			
deviceID	8	deviceID=9	Change the drive device number
splitIECLines		splitIECLines=1	If you are using split line hardware (option B), enable this
invertIECInputs		invertIECInputs=1	If you are using some type of hardware that requires the lines to be inverted, enable this
invertIECOutputs		invertIECOutputs=0	Use this if you are using a 7407 chip
RotaryEncoderEnable	0	RotaryEncoderEnable=1	Enable KY-040 Rotary Encoder hardware
ROM OPTIONS			
ROM1 alternate: ROM	d1541.rom, dos1541, d1541II. Jiffy.bin	ROM1=d1541.rom	Specify the default/first 1541 ROM. Switch back to it with F1 or button 1 + 2
ROM2		ROM2=Jiffy.bin	Specify the second 1541 ROM. Select with F2 or button 1 + 3
ROM3		ROM3=	Specify the third 1541 ROM. Select with F3 or button 1 + 4
ROM4		ROM4=	Specify the fourth 1541 ROM. Select with F4 or button 1 + 5
ROM5		ROM5=	Specify the fifth 1541 ROM. Select with F5
ROM6		ROM6=	Specify the sixth 1541 ROM. Select with F6
ROM7		ROM7=	Specify the seventh 1541 ROM. Select with F7
ROM1581	1581-rom.318045-02.bin	ROM1581=JiffyDos_1581.bin	Specify the 1581 ROM here that will be used when opening .D81 files.

FILENAME OPTIONS			
OnResetChangeToStartingFolder		OnResetChangeToStartingFoler=1	If using CBMFileBrowser then it is best to specify this option. When the computer resets, the Pi will always revert back to the root folder ready to load CBMFileBrowser again.
AutoMountImage		AutoMountImage=fb.d64	If you use FB64 (CBMFileBrowser) and want to use a fast loader cartridge (AR6, EFL, FC3) to load it, then use this option to automatically mount it.
LowercaseBrowseModeFilenames		LowercaseBrowseModeFilenames=1	If you use FB64 (CBMFileBrowser) and want Pi1541 to send all filenames as lower case.
AutoBootFB128	0	AutoBootFB128=1	Auto-boot FB128 (CBMFileBrowser) in 128 mode
128BootSectorName		128BootSectorName=bootsect.128	If you are using a 128, you can auto-boot anything with this option. Overrides AutoBootFB128.
StarFileName		StartFileName=somefile	You can specify what file will be loaded by LOAD "*" in browse mode.
AutoBaseName		AutoBaseName=autoname	The file base for auto-images from Alt-N (autoname001d64, autoname002.d64, etc.)
NewDiskType	d64	NewDiskType=g64	Change the type of disk image created by Alt-N
SOUND OPTIONS			
SoundOnGPIO		SoundOnGPIO=1	Enable piezo buzzer (if you have Pi1541 hat/hardware)
SoundOnGPIODuration		SoundOnGPIODuration=1000	Length of buzz in micro seconds
SoundOnGPIOFreq		SoundOnGPIOFreq=1200	Frequency of buzz in Hz
LCD OPTIONS			
KeyboardBrowseLCDScreen		KeyboardBrowseLCDScreen=1	If you are using the OLED screen and would like PageUp and PageDOwn keys to work with it (rather than HDMI screen) [Pi 3]

<table>
<tr><td>LCDName</td><td></td><td>LCDName=ssd1306_128x64
LCDName=ssd1306_128x32
LCDNam =sh1106_128x64</td><td>Specify the OLED screen</td></tr>
<tr><td>LCDLogoName</td><td></td><td>LCDLogoName=1541ii
LCDLogoName=1541classic
LCDLogoName=customfile.raw</td><td>Specify the startup logo for the OLED [Pi 3]</td></tr>
<tr><td>i2cBusMaster</td><td>0 for non-split/option A
1 for split/option B</td><td>i2cBusMaster=0
i2cBusMaster=1</td><td>Specify which pine for the OLED screen. For non-split/option A it uses SDA pin 27 and SCL pin 28. For split/option b it uses SDA pin 3 amd SC: [ome 5/</td></tr>
<tr><td>i2cLcdAddress</td><td></td><td>i2cLcdAddress=60</td><td>The i2C display address in decimal. Addresses on PCBs are usually shifted 1 bit left from the actual address..
<table><tr><td>Config</td><td>Actual</td><td>PCB Printed</td></tr><tr><td>60</td><td>0x3C</td><td>0x78</td></tr><tr><td>61</td><td>0x3D</td><td>0x7A</td></tr></table></td></tr>
<tr><td>i2cLcdFlip</td><td>0</td><td>i2cLcdFlip=1</td><td>Rotate i2c screen 180 degrees</td></tr>
<tr><td>i2cLcdOnContrast</td><td></td><td>i2cLcdOnContrast=127</td><td>Adjust the contrast on the i2c LCD screen</td></tr>
<tr><td>i2cScan</td><td>0</td><td>i2cScan=1</td><td>Scan i2C Bus and display addresses on screen</td></tr>
<tr><td>i2cLcdUseCBMChar</td><td>0</td><td>i2cLcdUserCBMChar=1</td><td>Use the CBM font on the LCD</td></tr>
<tr><td colspan="4">INPUT OPTIONS</td></tr>
<tr><td>buttonEnter</td><td>1</td><td>buttonEnter=1</td><td>Remap the physical button. Numbers correspond to the standard board layout. Not compatible with RotaryEncoderEnable.</td></tr>
<tr><td>buttonUp</td><td>2</td><td>buttonUp=2</td><td>Remap the physical button. Numbers correspond to the standard board layout. Not compatible with RotaryEncoderEnable.</td></tr>
<tr><td>buttonDown</td><td>3</td><td>buttonDown=3</td><td>Remap the physical button. Numbers correspond to standard board layout. Not compatible with RotaryEncoderEnable.</td></tr>
</table>

buttonBack	4	buttonBack=4	Remap the physical button. Numbers correspond to the standard board layout. Not compatible with RotaryEncoderEnable.
buttonInsert	5	buttonInsert=5	Remap the physical button. Numbers correspond to the standard board layout. Not compatible with RotaryEncoderEnable.
DISPLAY OPTIONS			
ChargenFont alternate: Font		ChargenFont=chargen	Use the Commodore chargen font
DisplayPNGIcons		DisplayPNGIcons=1	You can create 320x200 PNG files with the same name as your disk images. This will be displayed on the Pi's screen.
DisplayTemperature	0	DisplayTemperature=1	This will display the temperature of the Pi's CPU. It should be about 52C and anything above 65C is bad.
DisplayTracks	0	DisplayTracks=1	Displays a visualization of the data of a disk image on the Pi
GraphIEC		GraphIEC=1	Displays IEC bus activity on the bottom of the Pi's screen
ScreenWidth	512	ScreenWidth=512	Experiment with display resolution of the Pi's composite video out [Pi 3]
ScreenHeight	384	ScreenHeight=384	Experiment with display resolution of the Pi's composite video out [Pi 3]
scrollHighlightRate	0.07	scrollHighlightRate=0.07	The rate (in seconds) a long selection/filename is scrolled
BEHAVIOR OPTIONS			
DisableSD2IECCommands	0	DisableSD2IECCommands=1	If you would like to disable browse mode completely
IgnoreReset	0	IgnoreReset=1	Ignore Reset command
QuickBoot	0	QuickBoot=1	Faster startup
RAMBoard	0	RAMBoard=1	8k drive RAM expansion at 0x8000
ShowOptions	0	ShowOptions=1	Show some options on startup

Keyboard Shortcuts (not supported on Pi 0/1)

Selecting and Managing Images

A-Z, 1-0	Select an entry beginning with the corresponding key
HOME	Select first entry
END	Select last entry
BACKSPACE	Backs out of folder and clears selections
ALT-A	Automount image (defined in AutoMountImage option)
ALT-N	Create new D64 image (using AutoBaseName option)
ALT-W	Toggle write protection
ENTER	Mount the image and enter Emulation Mode
ESC	Exit Mounted Image, Exit Emulation Mode, Clear Selections

Managing Lists

INSERT	Add image to the selected list
ALT-ENTER	Add image to the selected list (useful for keyboards without an INSERT key)
ALT-L	Create autoswap.lst file of the selected images and ROM

Changing Device IDs

F8	Set Device ID to 8
F9	Set Device ID to 9
F10	Set Device ID to 10
F11	Set Device ID to 11

Changing ROMs

ROM filenames are set in options ROM1 - ROM7

F1	Select ROM1
F2	Select ROM2
F3	Select ROM3
F4	Select ROM4
F5	Select ROM5
F6	Select ROM6
F7	Select ROM7

Other

C= + SHIFT	Switch computer to lower-case mode (if you see garbage in CBMFileBrowser)
L-CTRL + L-ALT + DEL	Reboot the Raspberry Pi

Using Pi-Hat Buttons

Button actions can be redefined in the options file (buttonEnter, buttonUp, buttonDown, buttonBack, buttonInsert).

Browse Mode

❍ 1	❍ 2	❍ 3	❍ 4	❍ 5
Enter (Select)	Up (Previous)	Down (Next)	Back (Exit)	Insert into List

Emulation Mode

❍ 1	Exit emulation mode
❍ 2	Swap to previous disk
❍ 3	Swap to next disk
↓HOLD ❍ 1 + Press ❍ 2	Select ROM 1
↓HOLD ❍ 1 + Press ❍ 3	Select ROM 2
↓HOLD ❍ 1 + Press ❍ 4	Select ROM 3
↓HOLD ❍ 1 + Press ❍ 5	Select ROM 4
↓HOLD ❍ 5 + Press ❍ 1	Set Device ID to 8
↓HOLD ❍ 5 + Press ❍ 2	Set Device ID to 9
↓HOLD ❍ 5 + Press ❍ 3	Set Device ID to 10
↓HOLD ❍ 5 + Press ❍ 4	Set Device ID to 11

Using Browse Mode

Browse mode provides limited-compatibility with SD2IEC.

Options:

1. Use USB Keyboard to select the image and press ENTER

2. Use PiHat buttons to navigate

3. Use the special version of CBM-Browser (FB64, FB128, FB16, etc.) downloaded from the Pi1541 website (https://cbm-pi1541.firebaseapp.com/fb.zip). This is especially useful for loading PRG files. The special version has been modified to add extra error checking while waiting for the Pi to load the image.

   Note: Images with names longer than 16 chars will only show the first 16 chars in CBM-Browser. The first one will be selected. You can use the Pi1541 screen to work around this.

   Note: When using G64 images most of them autorun with LOAD "*",8,1 and some even have empty directories, which can confuse CBM Browser. Once you have loaded the image, quit out of CBM-Browser by pressing Q, and then you can do the normal LOAD "*", 8, 1.

   Note: Some software prevents their directory from being displayed via LOAD "$", 8. These will not work in CBM-Browser.

4. Use standard DOS command (replace 8 with your device ID):
   LOAD "$",8 would list all images in the current folder

   To select a specific image:
   OPEN 1,8,15:PRINT#1, "CD:ZORK.D64":CLOSE1

5. Use the limited SD2IEC Commands (see section below)

After selecting the image, it will enter Emulation Mode and you can use standard commands like LOAD "*", 8, 1 or LOAD "MYPROG.PRG",8 and RUN.

If you select a PRG file, the Pi1541 will automatically mount it inside a new D64 image.

SD2IEC Commands

You can use limited SD2IEC (https://www.sd2iec.de) commands in Browse mode. The easiest way is to use JiffyDOS or a DOS Wedge and use the commands with the syntax @COMMAND. Without using a wedge, you need to use the syntax:

OPEN 15,10,15,”COMMAND”:CLOSE 15

For space considerations, the documentation will most show the wedge-style format with a few exceptions.

Note: Any command not listed here is not currently supported.

References to device 8 should be changed to your current device ID (8, 9, 10, 11).

Directory Filters

To show only directories, both =B (CMD-compatible) and =D can be used.

Example: LOAD “$:*=B”,8 or @$

On a real Commodore drive D matches everything.

To include hidden files in the directory, use *=H (on a 1541 this doesn't do anything. sd2iec marks hidden files with an H after the lock mark, i.e. "PRG<H" or "PRG H").

CMD-style "short" and "long" directory listings with timestamps are supported ("$=T"), including timestamp filters. Examples: @$=T for timestamps, @$=T:*=L for long timestamps, and @$=T:*=<01/30/10 (show files younger than January 30, 2010).

Partition Directory

Partitions are not supported by Pi1541 commands, although you can switch the entire Pi1541 device to use another drive with the CP command.

Subdirectory Access (CD/MD/RD)

Subdirectory access is compatible with the syntax used by the CMD drives, although drive/partition numbers are completely ignored.

@CD:⬅ Alternate: @CD⬅	Changes into the parent dir (⬅ is the left arrow on the keyboard)
@CD:foo Alternate: @CD/foo	Changes into foo
@CD//foo	Changes into \foo
@CD/foo/:bar Alternate: CD/foo/bar	Changes into foo\bar
@MD/foo/:bar Alternate: MD//foo/:bar	Creates bar in foo
@RD:foo	Deletes foo

You can use wildcards anywhere in the path. To change into an M2I or D64 image the image file must be the last component in the path, either after a slash or a colon character.

MD uses a syntax similar to CD and will create the directory listed after the colon (:) relative to any directory listed before it.

RD can only remove subdirectories of the current directory.

CD is also used to mount/unmount image files. Just change into them as if they were a directory and use CD:_ (left arrow on the C64) to leave. Please note that image files are detected by file extension and file size and there is no reliable way to see if a file is a valid image file.

Change Partition (CP)

You can use these partition commands to switch the pi1541 data drive:

@CP0	Switch to SD card
@CP1	Switch to first USB drive
@CP2	Switch to second USB drive
@CP3-CP#	Switch to # USB drive

File Copy Command (C:)

Should be CMD compatible. The syntax is:

@C:targetname=sourcename[,sourcename2,...]

You can use this command to copy multiple files into a single target file in which case all source files will be appended into the target file. Parsing restarts for every source file name which means that every source name is assumed to be relative to the current directory. You can use wildcards in the source names, but only the first file matching will be copied.

Copying REL files should work, but isn't tested well. Mixing REL and non-REL files in an append operation isn't supported.

Direct Info (DI), Direct Read (DR), Direct Write (DW)

Not Supported

Get Partition Information (G-P)

Not Supported

Positioning (P)

Positioning doesn't just work for REL files but also for regular files on a FAT partition. When used for regular files the format is

"P"+chr$(channel)+chr$(lo)+chr$(midlo)+chr$(midhi)+chr$(hi)

which will seek to the 0-based offset hi*2^24+midhi*65536+256*midlo+lo in the file. If you send less than four bytes for the offset, the missing bytes are assumed to be zero.

New Disk (N)

See "Creating a Blank Image" in this documentation.

Renaming (R)

Renaming files should work the same as it does on CMD drives, although the errors flagged for invalid characters in the name may differ.

The format is:

@R:targetfile=sourcefile

You can also do:
OPEN1,8,15:PRINT#1,"R1:newname=oldname":CLOSE1
or in BASIC 7.0:
RENAME "oldname" TO "newname", U8

Name matching is fully supported, directories are ignored. Wildcards can use ? or *.

Scratching (S) (Deleting) Files

Format:

@S:file1[,file2,...]

You can also do:
OPEN1,8,15:PRINT#1,"S:filename[,:secondfilename]":CLOSE1
or in BASIC 7.0:
SCRATCH "filename", U8

Name matching is fully supported, directories are ignored. Wildcards can use ? or *.

Scratching of multiple files separated by , is also supported with no limit to the number of files except for the maximum command line length (usually 100 to 120 characters).

Example:
OPEN 1,8,15:PRINT#1,"S:JUNK,C?*.BAS":CLOSE 1

Real Time Clock (T-R, T-W)

If your hardware features RTC support the commands T-R (time read) and T-W (time write) are available. If the RTC isn't present, both commands return 30,SYNTAX ERROR,00,00; if the RTC is present but not set correctly T-R will return 31,SYNTAX ERROR,00,00.

Both commands expect a fourth character that specifies the time format to be used. T-W expects that the new time follows that character with no space or other characters in between. For the A, B and D formats, the expected input format is exactly the same as returned by T-R with the same format character; for the I format the day of week is ignored and calculated based on the date instead.

The possible formats are:

"A"SCII:

"SUN. 01/20/08 01:23:45 PM"+CHR$(13)

The day-of-week string can be any of "SUN.", "MON.", "TUES", "WED.", "THUR", "FRI.", "SAT.". The year field is modulo 100.

B"CD or "D"ecimal:

Both these formats use 9 bytes to specify the time. For BCD everything is BCD-encoded, for Decimal the numbers are sent/parsed as-is.

Byte 0: Day of the week (0 for sunday)
Byte 1: Year (modulo 100 for BCD; -1900 for Decimal, i.e. 108 for 2008)
Byte 2: Month (1-based)
Byte 3: Day (1-based)
Byte 4: Hour (1-12)
Byte 5: Minute (0-59)
Byte 6: Second (0-59)
Byte 7: AM/PM-Flag (0 is AM, everything else is PM)
Byte 8: CHR$(13)

When the time is set a year less than 80 is interpreted as 20xx.

"I"SO 8601 subset:

"2008-01-20T13:23:45 SUN"+CHR$(13)

This format complies with ISO 8601 and adds a day of week abbreviation using the same table as the A format, but omitting the fourth character. When it is used with T-W, anything beyond the seconds field is ignored and the day of week is calculated based on the specified date. The year must always be specified including the century if this format is used to set the time. To save space, sd2iec only accepts this particular date/time representation when setting the time with T-WI and no other ISO8601-compliant representation.

Device Changing (U0)

Device address changing with "U0>"+chr$(new address) is supported, Other U0 commands are currently not implemented.

Block Reading/Writing (U1/U2/B-R/B-W)

Not Supported

Buffer Pointer (B-P)

Not Supported

Bus Protocol (UI+/UI-)

Switching the slightly faster bus protocol for the VIC-20 on and off works, it hasn't been tested much though.

Soft/Hard Reset (UI/UJ)

@UI for warm/soft reset just sets the "73,..." message on the error channel,

@UJ for cold/hard reset closes all active buffers but doesn't reset the current directory, mounted image, swap list or anything else.

@U<Shift-J> performs a real hard reset. This command causes a restart of the Raspberry Pi processor. <Shift-J> is character code 202.

Extended Commands (X)

Not Supported except X?

Memory Read (MR), Memory Write (MW), Memory Execute (ME)

Not Supported

Multiple Disks/Sides

To use multiple disks/sides, you select multiple images (INSERT or ALT-ENTER, or button 5) to add an image into a mount list.

Once the first image has been mounted/loaded, you can change to the next one by using the number keys on the keyboard or by using the up/down buttons (2/3) on the hat.

You can also use predefined lists of disk images that end in .LST. You simply load the .LST and use it. To save your current mount list as a .LST, hit ALT-L.

See this video by Jarkko Lehti for more explanation: https://www.youtube.com/watch?v=3tIAPX9ziDM

Using Emulation Mode

Example DOS commands:

DOS	JiffyDOS	
LOAD “$”, 8 LIST	@$	Get a list of files (on JiffyDOS, select drive with @#DEVNUM or CTRL-D)
LOAD “*”, 8, 1	🠅 filename	Load a basic program and run it
LOAD “MYPROG.PROG”,8,1	🠅 filename	Run a specific named file

Read-Only Mode

Set the SD Card to read-only to make all images read only.

Toggle a single image to read-only or read-write by pressing ALT-W

Creating a Blank Image

Pi1541 only supports creating D64 and G64 images.

Press ALT-N on the keyboard to create a new D64 image (it will use the AutobaseName config option for naming).

You can also use:
@N:diskname,XYZ
or
OPEN 1,8,15,”N:FILENAME,XYZ”:CLOSE 1
Or in BASIC 7:
HEADER “FILENAME.D64”, XYZ, U8

For above examples, XYZ is a 3-char disk ID and could be any number, and 8 is the drive device ID.

NOTE: If you are emulation mode, you will overwrite your current disk.

For games that require a blank disk, you can add your new blank image into the .LST file.

To create D81 images, create them outside of Pi1541 (like in VICE) and copy them to the SD card.

Using Tape Images

Pi1541 will copy out all the PRG files inside a T64 and put them inside a D64 that is then used for emulation.

Note: currently only PRG files inside the T64 container are supported.

Note: If you use FB (CBM-FileBrowser) and would like to use T64 files you will need to use the versions found at https://cbm-pi1541.firebaseapp.com/fb.zip

Note: Currently any changes are NOT written back to the T64.

Using JiffyDOS

To use JiffyDOS, set your ROM1 or 1581ROM to a ROM containing JiffyDOS, and refer to the JiffyDOS manual or examples elsewhere in this document

Fast Loaders

Cartridges

Supports Action Replay, Epyx Fastloader, and Final Cartridge.

Note: Does not work with Epyx Fastloader when in browse mode. Set FB64 as your automount image then use LOAD “*”, 8, 1

Place a disk image containing FB64 in the 1541 folder of the SD card.

Add the following line to the options.txt file:
autoMountImage = fb.d64

Now whenever the emulated drive is reset, the FB64 image will be automatically selected and mounted. You can use your fast loader cartridge of your choice to load it. Once loaded and running, you can back out of the FB64 image and browse the SD card as usual.

Software Loader

LOAD “$”, 8
LOAD “EPYX.PRG”
RUN

Then navigate (with keyboard, Pi1541 Hat Buttons 2/3)

Select the image (ENTER on keyboard, or switch 1)

LOAD “*”, 8
RUN

Reset

Reset by exiting an image/Emulation Mode, or by using the Reset button on the Pi Hat, or by power-cycling the Raspberry Pi. You can also add a reset switch on your cable.

The emulated 1541 may otherwise not reset when you power-cycle your Commodore computer, depending on model and revision.

Limitations as of v1.24

Pi 0/1 Version does not support:

- HDMI screen
- USB keyboard
- Emulated drive sounds
- USB external drives

Blank Disk Creation:

- Only D64 and G64 images are supported.

1571/D71 support:

- Commodore 1571 is not supported.

1581/D81 support:

- C128 Burst mode is not supported.
- Only 512 byte physical sectors are supported.
- You cannot specify alternate ROMs.

T64 Support:

- Only PRG files inside the T64 container are supported.
- Changes are not written back to the T64.

SD2IEC Support is limited. Features not supported:

- Partitions
- U0 only supports device address changing.
- Block reading/writing (U1/U2/B-R/B-W)
- Direct sector access (DI, DR, DW)
- Buffer Pointer (B-P)
- Extended commands (X)
- Memory read (M-R), Memory Write (M-W), Memory Execute (M-E)

USB Drive Support:

- Drives must be FAT32.
- Only the first partition can be used.
- Drives must be inserted before the Pi is powered Up.
- All configuration and ROM files still need to be on the SD card (firmware files will be copied to the SD card).
- Will not work with Pi 0/1

Piezo Buzzers:

- Only buzzers without generators are supported

Epyx Fastload cartridge will not work in browse mode. Set FB64 as your automount image then use LOAD “*”, 8, 1

Troubleshooting

Please ensure that you use an adequate power supply for the Pi. The Raspberry Pi Foundation recommends a 2.5A minimum. So far, 100% of problems encountered by others in setting up and getting PI1541 working were down to them using an incorrect power supply or attempting to power all kinds of exotic devices through the Pi's USB ports whilst trying to use it. If the Pi displays the thunder bolt icon then the power supply is insufficient and Pi1541 will probably not work.

If building option B then please use genuine branded parts. People are reporting that the use of no named Chinese hex inverters can lead to some disk images not working. Ghosts'n Goblins Arcade can be used to test the capability of your hex inverter. If the title screen is corrupt then please substitute the hex inverter IC for a genuine branded version.

Be aware that there are a lot of suspect NIBs and G64s out there that don't work even when transfered to real floppies (even inside c64pp)

Some Demos can crash some C64s that are susceptible to the VSP bug. This is not caused by Pi1541.

Acknowledgements

nbla000 for giving me permission to add what I needed to the CBM-Browser.

Pete Rittwage for help, support, advice, code and testing.

Petros Kokotis and Mateusz Malina for above and beyond testing. The high level of compatibility would not have been achieved if it was not for these two guys.

Todd Trann for his testing, as well as, help with diagrams and images.

I also greatly appreciated the advice, testing and feedback from Rene, Greg Dunlap, penfold42 and Thomas Christoph.

Michael Long for user manual compilation.

License