RP2040 RunCPM Emulator with USB Keyboard & HDMI screen

Created by Jeff Epler


Last updated on 2023-08-29 04:56:19 PM EDT
# Table of Contents

**Overview**
- Parts 3

**Set Up Arduino**
6

**Set up the SD Card**
8

**Soldering and Wiring**
10

**Code the Emulator**
12
- Compiling the Emulator Code
- Booting Up
- A Note about Overclocking

**Code the Keyboard Converter**
14

**3D Printing and Assembly**
16

**Using CP/M**
18
- ED.COM, the CP/M editor
- ASM.COM, the CP/M assembler

**Playing Zork**
22
Overview

The CP/M operating system was created in 1974 and ran on the 8080 and Z-80 microprocessors. Before the rise of the IBM PC & MS-DOS, it was considered a powerful operating system, especially for home computers.

There are a number of CP/M emulators & simulators. One of them, RunCPM by MockbaTheBorg, was previously adapted to run on the RP2040 family of microcontrollers by guidol70. This project adds USB for keyboard input & HDMI for display output. The resulting computer is entirely self-contained and doesn't need a host computer to operate.

This project uses two Adafruit Feathers. Why's that? Both the digital video output and the USB Host are implemented using the RP2040's powerful PIO peripheral—but each task requires almost all of the chip's capacity in order to do so. That made it necessary to split the project in half. One of the Feathers does the CPU and OS emulation plus the video signal generation. The other reads keystrokes and transmits them over a UART connection to the first board. It's not a Beowulf cluster, but it's still a cool technique to expand the capabilities in this project.

After putting the project together, you'll want to do something with it! There's a great deal of software for CP/M readily available on the internet. This guide will cover where to download some classic & modern CP/M software. With an assembler or a BASIC interpreter, you can even write your own new CP/M software right on the device.
Parts

**Adafruit Feather RP2040 with DVI Output Port - Works with HDMI**
Wouldn't it be cool if you could display images and graphics from a microcontroller directly to an HDMI monitor or television? We think so! So we designed this RP2040 Feather that...
https://www.adafruit.com/product/5710

**Adafruit Feather RP2040 with USB Type A Host**
You're probably really used to microcontroller boards with USB, but what about a dev board with two? Two is more than one, so that makes it twice as good! And...
https://www.adafruit.com/product/5723

**Adalogger FeatherWing - RTC + SD Add-on For All Feather Boards**
A Feather board without ambition is a Feather board without FeatherWings! This is the Adalogger FeatherWing: it adds both a battery-backed Real Time Clock and micro SD...
https://www.adafruit.com/product/2922
1 x Hand removing/installing micro SD card from SD adapter SD/MicroSD Memory Card (8 GB SDHC)
Add mega-storage in a jiffy using this 8 GB class 4 micro-SD card. It comes with a SD adapter so you can use it with any of our shields or adapters. Preformatted to FAT so it works out...
https://www.adafruit.com/product/1294

1 x 512MB micro SD Memory Card
Add storage in a jiffy using this 512MB microSD card. Preformatted to FAT32, so it works out of the packaging with our projects. Works great with any device in the...
https://www.adafruit.com/product/5252

1 x USB MicroSD Card Reader/Writer - microSD / microSDHC / microSDXC
This is the cutest little microSD card reader/writer - but don't be fooled by its adorableness! It's wicked fast and supports up to 64 GB SDXC cards! Simply slide the card into...
https://www.adafruit.com/product/939

HDMI 5" Display Backpack - Without Touch
It's a mini panel-mountable HDMI monitor! So small and simple, you can use this display with any computer that has HDMI output, and the shape makes it easy to attach to a case or rail....
https://www.adafruit.com/product/2232

FeatherWing Tripler Mini Kit - Prototyping Add-on For Feathers
This is the FeatherWing Tripler - a prototyping add-on and more for all Feather boards. This is similar to our
Add a good quality, slim keyboard to your Raspberry Pi, Beagle Bone Black or other mini-computer with this sleek black chiclet keyboard. It's a full QWERTY keyboard with a wireless...

1 x **Angled shot Mini Wireless Keyboard - Black w/ Batteries**

This cable is not only super-fashionable, with a woven pink and purple Blinka-like pattern, it's also made for USB C for our modernized breakout boards, Feathers, and...

1 x **Pink and Purple Woven USB A to USB C Cable - 1 meter long**

This cable is not only super-fashionable, with a woven pink and purple Blinka-like pattern, it's also fully reversible! That's right, you will save seconds a day by...

1 x **Fully Reversible Pink/Purple USB A to micro B Cable - 1m long**

Connect two HDMI devices together and save space with this basic flat HDMI 1.4 cable. It has nice molded grips for easy installation, and is 1 foot long (~30 cm). This cable is...

1 x **HDMI Flat Cable - 1 foot / 30cm long**

Totaling 380 pieces, this M2.5 Screw Set is a must-have for your workstation. You'll have enough screws, nuts, and hex standoffs to fuel your maker...

Set Up Arduino

This project uses the Adafruit PicoDVI and USB Host Feather on the Earle Philhower III Arduino core for programming — an optional package that makes most RP2040 boards work in the Arduino environment. If you’ve previously followed any guides for Adafruit RP2040-based boards in Arduino, you likely already have this installed... just check that you’re up to date with the latest (3.2.1 or newer).

If that sounds unfamiliar, [this guide walks through the process](https://www.adafruit.com/product/3299).
Once installed, the Arduino IDE Tools → Board menu will include a rollover for “Raspberry Pi RP2040 Boards,” and you can find and select whatever board type you’re using (e.g. Feather RP2040 DVI or Raspberry Pi Pico).

Next, the PicoDVI library can be installed from the Arduino Library Manager. From the Sketch menu...

Sketch → Include Library → Manage Libraries...

Enter “picodvi” in the search field and look for PicoDVI - Adafruit Fork in the results. Click “Install.”.

Our version of PicoDVI depends on the Adafruit_GFX library. The Library Manager should install this automatically if not already present, but if using an older version of the Arduino IDE you might need to search for and install it manually.

This is the Adafruit “fork” of the original PicoDVI project, meaning as much of the original code is preserved with minimal changes. What we’ve done is add an Arduino-compliant C++ wrapper to make this command-line library work with the friendlier Arduino IDE, and implemented simple raster framebuffers for drawing. All the original stuff is there if you want to dig in and learn, though the original examples as written won’t build in the Arduino IDE.
Here's our fork on GitHub, and Luke Wren's original project.

For the USB Keyboard Co-Processor, search for and install the Pico PIO USB library by sekigon-gonnoc. It depends on Adafruit TinyUSB Library which should be installed automatically unless you're on a very old version of the Arduino IDE.

Once you've installed the required libraries, you can close the library manager window and continue.

**Set up the SD Card**

The RunCPM emulator is very particular about how files are organized!

CP/M uses drive letters (A, B, ...) and user numbers (0, 1, ...). The structure of files on the SD card has to mimic this arrangement and all filenames must be in upper case ASCII only, with no more than 8 letters in the filename and no more than 3 letters in the extension.

So, for example, if you want the file "example.hex" to appear in the "B" drive for the default user (user zero), you would create the folder "B/0" and put the file in it as "EXAMPLE.HEX". A filename like "example.txt" (lowercase) "EXAMPLE.TEXT" (incorrect length of extension) or any filename that contains non-ASCII characters will not be accessible by RunCPM.

For a starter set of files, grab the standard CP/M 2.2 files from retroarchive.org. Create the folder "A/0", and copy the files from that zip to A/0. It's important that you don't create any other intermediate folders!

If you like the classic text adventure game Zork, retroarchive has you covered. You can follow a similar process but put these files a B/0 on the SD card.
With that done, safely remove or eject the SD card from your computer and stick it in the adalogger's micro SD card slot.

For your convenience, and because some browsers don't like downloading from the non-https retroarchive site, you can also download using the links below:

- STDCPM22.ZIP
- zork123_80.zip

Once you're done, the folder & file structure should look like this:

```
A
  \-- ASM.COM
      \-- BIOS.ASM
      \-- CBIOS.ASM
      \-- CCBIOS.ASM
      \-- CCBOOT.ASM
      \-- -CCSCPM.251
      \-- CCSINIT.COM
      \-- CCSYSGEN.COM
      \-- CPM24CCS.COM
      \-- DDT.COM
      \-- DEBLOCK.ASM
      \-- DISKDEF.LIB
      \-- DUMP.ASM
      \-- DUMP.COM
      \-- ED.COM
      \-- GENMOD.COM
      \-- HELLO.HEX
      \-- LOAD.COM
      \-- MOVCPM.COM
      \-- PIP.COM
      \-- RLLOCBIOS.COM
      \-- STAT.COM
      \-- STDBIOS.ASM
      \-- SUBMIT.COM
      \-- SYSGEN.COM
      \-- XSUB.COM

B
  \-- FILE_ID.DIZ
      \-- ZORK1.COM
      \-- ZORK1.DAT
      \-- ZORK2.COM
      \-- ZORK2.DAT
      \-- ZORK3.COM
      \-- ZORK3.DAT

4 directories, 33 files
```
Soldering and Wiring

(New to soldering? There's a guide for that!) (1)

Solder male headers on the two Feathers and the FeatherWing, with the long pins sticking out the bottom. (Also shown: The Micro SD card & wireless keyboard receiver have been inserted.)

Solder female headers on the top of the FeatherWing Tripler, noting the small outlines which show where the long & the short headers must be placed. (Also shown: these are the locations where you'll later screw the tripler into the 3D printed bracket.)

Insert the boards in the feather tripler, double checking orientation and position. The order can be whatever you like. Take care, and make sure all the pins make good contact. If the connections are "off by one", the electronics can become damaged.
There are several cable connections needed. Connect a USB C power / data cable to just one of the feathers. The second feather receives power via the Feather Tripler.

When you upload code to one of the feathers, make sure the USB C cable is plugged into the correct one. If you make a mistake, no fear: just re-program both feathers to get the software back in the right place.

Connect a HDMI cable from the HDMI Feather to the HDMI monitor of your choice. Here, it's connected to the 5" display backpack.

The 5" Display Backpack receives power from a USB Micro B cable.
Insert the SD card and USB keyboard (shown here is the receiver for a wireless keyboard).

---

**Code the Emulator**

**Compiling the Emulator Code**

Begin by grabbing the code. Unzip it in your Arduino folder, then open it with the Arduino IDE.

**Download Arduino Sketch Source Code (zip)**

Under Tools, there are a number of selections you have to make:

- For board, choose "Adafruit Feather DVI"
- For optimization, choose "Optimize even more (-O3)"
- For flash speed choose "W25Q080 QSPI /4"
- For USB stack choose "Adafruit TinyUSB"
Booting Up

Now, plug your Adafruit Feather DVI in to your computer with a USB cable while holding down the BOOT button on the Feather to enter bootloader mode. Then select "upload sketch" in Arduino. Once the sketch is uploaded, plug the video out on the Feather to a monitor using a HDMI cable and reset the Feather with the reset button. You will see the RunCPM boot screen.

Using the Serial Monitor in Arduino you can run CP/M commands like "DIR". Follow the next steps so that you can use a USB keyboard connected to the Feather USB Host instead of the serial connection.
A Note about Overclocking

Generating digital video signals takes a lot of oomph! This sketch overclocks your RP2040 board to 148MHz, and increases the core voltage to 1.20V. The software documented here pushes the RP2040 microcontroller beyond its design specifications. Just like PC overclocking, there's some risk of reduced component lifespan, though the extent (if any) can't be precisely quantified and could vary from one chip to another. Proceed at your own discretion. It also makes CP/M emulation blisteringly fast compared to the 4MHz Z80A!

Code the Keyboard Converter

A retro computer is no fun without a way to type into it. Happily, the Adafruit Host Feather can be used to convert almost any USB keyboard into a keyboard for the RunCPM emulator.

Start by grabbing the keyboard converter sketch from GitHub:

Keyboard Converter Arduino Sketch

Now, make the appropriate settings in the Tools menu:

- For board, choose "Adafruit Feather RP2040 USB Host"
- For USB stack choose "Adafruit TinyUSB"
Un-plug the emulator/video feather from your computer and plug the USB Host feather into your computer using a USB cable while holding down the BOOT button. It's important that the correct board is plugged in at this step, so that the correct board is updated! Select "upload sketch".

After the upload is complete, open the serial monitor and plug a compatible USB keyboard into the USB A port on the USB Host Feather. When you type letters on the keyboard, they will be shown on the serial monitor window. The bytes are also being transmitted on the board's RX pin for RunCPM to receive.

Note that due to technical limitations, a keyboard's capslock and numlock LEDs (if present) will not light, but these keys do function in the normal fashion.

The keyboard is set to a QWERTY US English layout and can only be changed by editing the source code and recompiling.
3D Printing and Assembly

If you're using your RunCPM emulator with the 5" Display Backpack, there's a 3D printable bracket you can mount everything to.

Print the bracket on its side, as shown. Use a layer height of 0.2mm to 0.3mm. On most printers, no support is required. PLA and PETG are very suitable for this print. Use the download button below to grab the 3D printable STL file as well as the FreeCad design file:

cpmish-display-backpack-bracket.zip
From the M2.5 mounting hardware box, pick out:

- 16 of the M2.5 x 4mm Screws (the shortest screw)
- 8 of the 6mm long F-F hex standoffs (the shortest standoff)

Begin by installing the stand-offs:

The outermost 4 holes are for the display. At each position, insert a screw from the back and tighten a stand-off on the front. The remaining holes support a number of different boards & configurations. Find the inner columns of 6 holes each and place stand-offs on the back in the first and last positions, fastening from the front with screws.

Next, attach the display on the front with the connectors on the right and hand-tighten the 4 screws on the front.
Finally, remove the boards from the feather tripler and attach the Feather tripler on the back side of the bracket, orienting it so that the DVI connector will be on the same edge as the display. Counting from the bottom of the tripler, screw in the first and fourth sets of holes into the standoffs.

Re-install the Feathers and FeatherWing, connect the display, keyboard and power. Then power on!

Using CP/M

CP/M is a text-oriented, command-line interface. Only a few simple utilities were included with the original base system.

The original CP/M manuals are online. This version has been scanned and is pretty easy on the eyes but has occasional weird mistakes introduced by the optical character recognition process. There's also a good quality scan on archive.org. If you want to get in depth about how to use ED, PIP, ASM, and the rest, this is a good place to start.

Similar to MS-DOS, you can select among drives with the commands "A:" and "B:". If you followed the instructions when setting up the SD card, the A: drive will be filled
with the CP/M operating system and some utilities; the B: drive has several games in the Zork series.

Most references will show the CP/M prompt as `A>` or `B>` but RunCPM also displays the "user number", which defaults to 0: `A0>` or `B0>`. (Drives can actually go up to letter "P", which is the 16th letter)

Another feature of RunCPM is that when searching for a program, A0 is searched after the current drive and user are searched, so think of A0 as the place you install programs.

The easiest way to move and copy files is probably to put the SD card in a real computer and use the file manager on that computer. In CP/M the multi-purpose program PIP could be used to copy a file to a new name or to a new drive. For example, to copy the program ASM.COM from the A drive to the B drive, you would write

`PIP B:=A:ASM.COM`

The command REN renames files, for instance if you need to correct a mis-typed filename:

`REN EXAMPLE.TXT=EXMAPLE.TXT`

To erase (delete) a file, use the ERA command (the extension $$$ is often used for temporary files):

`ERA TEMPFILE.$$`

**ED.COM, the CP/M editor**

The original CP/M editor, ED, is a line oriented editor that feels very primitive by modern standards. Have you encountered a joke about how the classic Unix editor vi is for beginners? ED says "hold my beverage!"

- To begin inserting lines, use the "i" command
- When done inserting lines, use Ctrl+Z on its own line to return to command mode
- Write your file & exit with the "e" command.
Here's a sample ED session, with the parts that the user types shown in **bold code** and the stuff that RunCPM prints in *regular code*:

```
RunCPM [v6.0] => CCP:[INTERNAL v3.0] TPA:64K
B0>ed hello.txt
  : *i
  1: now is the time for all cool people
  2: to write text in ED.COM
  3: and then save it to disk
  4: (type ctrl-z here)
  : *e
```

```
RunCPM [v6.0] => CCP:[INTERNAL v3.0] TPA:64K
B0>type hello.txt
now is the time for all cool people
to write text in ED.COM
and then save it to disk
```

It's also possible to read back a file and make changes, but showing how to use ED fluently is outside the scope of this guide.

**ASM.COM, the CP/M assembler**

Using ED, or by preparing the file on your host computer and copying it, put the following content into HELLO.ASM on the B drive for user 0 (B\0\HELLO.ASM is the path on the host computer):

```
bdos equ 0005H
org 0100H
;
  lxi d,message
  mvi c,9
  call bdos
  ret
;
message db 'Hello world.', 10, 13, '$'
```

Like most CP/M programs, this program begins (has its origin) at the hex address 0100 (256 in decimal). It calls into the operating system ("bdos", at address 0005 hex) with the command register set to 9 (print string), and the address argument set to the
message. bdos command 9 prints each byte starting at its address argument until a 
"$" is encountered. 10 and 13 represent the "carriage return" and "line feed" special 
characters. When call bdos finishes, the program exits by running the return 
instruction.

Note that in order for this file to work properly, it has to be created with "DOS-style 
line endings", or CRLF line endings. This may require special settings in your text 
editor, so feel free to grab a correct copy using the link below.

Now, assemble it to create hello.hex, which you can show with the TYPE command:

```
B0>asm hello
CP/M ASSEMBLER - VER 2.0
0118
000H USE FACTOR
END OF ASSEMBLY
CPM [v6.0] => CCP:[INTERNAL v3.0] TPA:64K
B0>type hello.hex
:100100001109010E09CD0500C948656C6C6F207797
:080110006F726C642E0A0D24CD
:0000000000
```

To convert it to an executable program, load it using the debugger DDT, then 
immediately exit DDT by jumping to the warm start address 0 with the g0 command, 
and finally save the program using the SAVE command (the count argument, 1, is the 
number of 128-byte units occupied by the program). After doing this, it is possible to 
run the HELLO.COM program:

```
B0>ddt hello.hex
DDT VERS 2.2
NEXT PC
0118 0000
-g0

RunCPM [v6.0] => CCP:[INTERNAL v3.0] TPA:64K
B0>save 1 hello.com
..
B0>hello
Hello world.
```
(In real CPM usage, it would be more typical for the LOAD program to be used instead of the DDT program, but for an unknown reason this did not work in RunCPM when the author tested it, so the method using DDT is shown instead.)

You can also develop CP/M software on your host computer with software suites like the [z88 development kit](https://www.z88.de/) or the [Amsterdam Compiler Kit](https://www.amd.de/), though this is beyond the scope of this guide.

---

### Playing Zork

I think it's time to admit it: The real goal all along was to play games. That's comparatively simple, simply change to the B: drive and run the command ZORK1:

```
A0>B:
B0>zork1
```

**ZORK I: The Great Underground Empire**

Copyright (c) 1981, 1982, 1983 Infocom, Inc. All rights reserved.

ZORK is a registered trademark of Infocom, Inc.

Revision 88 / Serial number 840726

---

**West of House**

You are standing in an open field west of a white house, with a boarded front door. There is a small mailbox here.

```
>open mailbox
Opening the small mailbox reveals a leaflet.
```

```
>take
(leaflet)
Taken.
```

I hope you don't mind a spoiler about this game: I've never won it myself, but I've been told it's important to pick up that leaflet, it'll be useful later.