Running Doom on the RP2040

Equipment

- Raspberry Pi Pico H Raspberry Pi Pico - Pimoroni
- Pimeroni Pico VGA Demo Base Pimoroni Pico VGA Demo Base - Pimoroni
- USB keyboard (we used the Raspberry Pi Keyboard) Raspberry Pi Keyboard (UK layout) - Pimoroni
- Aux audio cable (Male to male 3.5 mm)
- 2 x USB to micro USB cables
- Female to male VGA cable
- TV/Monitor

Additional equipment may be required based on the inputs to your TV or monitor.

- If your TV/monitor has a VGA input you can just plug your VGA cable in directly to your device. You may also be able to plug the audio cable in if your chosen device also supports audio, if not you could plug it into an external speaker.
- A nice solution is to convert both the VGA and audio DAC signals from the VGA demo board into a single HDMI output than can be plugged straight into any TV with a HDMI input. This requires a VGA/Stereo Audio to HDMI converter (we found one by Multibao off Amazon) and a standard HDMI cable.

Software

We need to flash two files to the RP2040. The first one is called “doom_tiny_usb.uf2” which gives your Pico the ability to run doom (the drivers etc). The second one is the doom game itself “doom1.whx”.

Step 1 – Downloading the Doom Files

Download both files from Graham Sanderson’s Github (he’s a Raspberry Pi employee):

“doom_tiny_usb.uf2” Releases · kilograham/rp2040-doom · GitHub

“doom1.whx” GitHub - kilograham/rp2040-doom: Fully-featured Doom port for the Raspberry Pi RP2040 microcontroller

There a lot of files on these pages but you only need to download the two mentioned. Save the uf2 file to C:\Users\[your_name], we will change the location of the other file later.

Step 2 – Setting up File Directories and Flashing the First File

Now we need to install the Raspberry Pi Pico toolchain. [1] Raspberry Pi has a useful getting started guide for the Pico that covers installation steps for the major operating systems.

Getting started with Raspberry Pi Pico

On Linux you can run a single script that will install everything for you. On macOS, you need to use Homebrew to install the toolchain, which is only a few commands in the terminal.

The process for Windows is a bit more involved, so we will tackle it below:

1) Lets get organised! Create a new file named VSARM in the top level of your C drive. Next inside this file create 4 files called 'armcc', 'lib', 'mingw' and 'sdk'.
2) Install GitBash for Windows https://git-scm.com/download/win (all the default installation options are fine).
3) Now we are ready to flash the "doom_tiny_usb.uf2" to our pico! To do this hold down the bootsel button on your pico as you plug it into your laptop or computer.
4) Find which drive letter the RPI-RP2 drive is mounted to – (ours was d, just check on your file explorer and replace the /d/ in the line below accordingly)
5) Enter the command and the file will be flashed:

```
cp doom_tiny_usb.uf2 /d/
```

6) Create a new folder called "pico" in the sdk folder we created earlier.
7) Enter the following commands in GitBash:

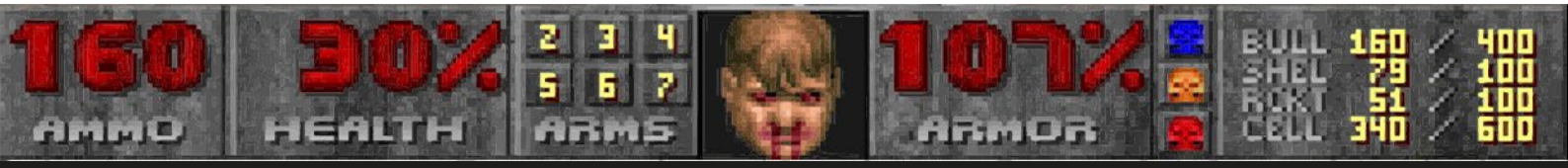
```
cd /c/VSARM/sdk/pico
git clone -b master https://github.com/raspberrypi/pico-sdk.git
cd pico-sdk
git submodule update --init
cd ..
git clone -b master https://github.com/raspberrypi/pico-examples.git
```

The pico-examples aren't necessary for getting doom running, so feel free to omit these commands, but they provide a useful reference and starting point for any future Pico projects!

8) We will also update the environment variables for out pico-sdk.
    - In the Windows search bar, enter env. Click on Edit the system environment variables.
    - In that window, click on Environment Variables…
    - Under User variables for <username>, select Path and click Edit.
    - Variable name: PICO_SDK_PATH
    - Variable value: C:\VSARM\sdk\pico\pico-sdk

Step 3 – Flashing the Second File

Now we have installed the Raspberry Pi Pico toolchain and flashed our first file. For the "doom1.whx" file we want to flash to a specific point in the flash storage on the Pico – this requires a tool called Picotool.



1) We are going to need another version of Git which will help us build Picotool on Windows. Install Git for Windows SDK here https://gitforwindows.org/#download-sdk (The latest 64-bit version). Run the installer, accepting all the defaults. This will open a command prompt window and begin to download/install Git for Windows SDK. We are going to use a feature of this version of Git called Pacman- this is a package manager that will let us easily install the libraries and software we need.
2) Run C:\git-sdk-64\git-bash.exe and enter the commands:

```
pacman -Syu
pacman -Su
pacman -S mingw-w64-x86_64-toolchain git make libtool pkg-config
autoconf automake texinfo wget
```

When prompted hit enter to install all the packages. Then enter Y twice to the consecutive questions. Enter the following commands to install libusb-1.0.23:

```
cd ~/Downloads
wget http://repo.msys2.org/mingw/x86_64/mingw-w64-x86_64-
libusb-1.0.23-1-any.pkg.tar.xz
pacman -U mingw-w64-x86_64-libusb-1.0.23-1-any.pkg.tar.xz
```

At the next question enter Y again to continue with the installation. Now we are ready to build Picotool!

3) In the GitBash window enter the following commands to install picotool:

```
cd /c/VSARM/sdk/pico
git clone -b master
https://github.com/raspberrypi/picotool.git
cd picotool
mkdir build
cd build
cmake -G "MinGW Makefiles" -DPC_LIBUSB_INCLUDEDIR="/c/git-sdk-64/
mingw64/include/libusb-1.0" ..
make
cp /c/git-sdk-64/mingw64/bin/libusb-1.0.dll .
cp /c/git-sdk-64/mingw64/bin/libgcc_s_seh-1.dll .
```

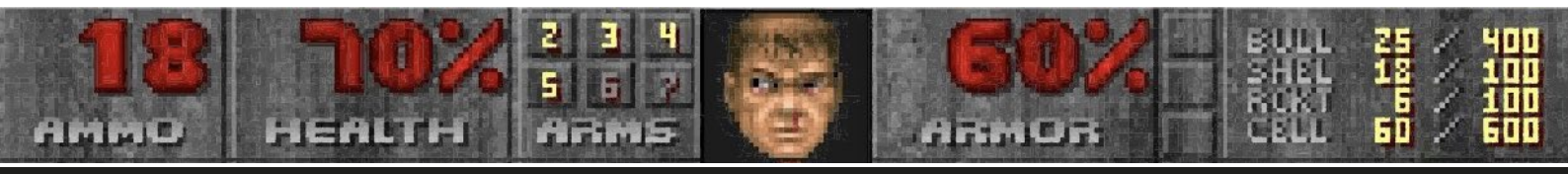
Do a quick test to check Picotool is working okay, enter the command and it should spit out some info and then exit:

```
.picotool.exe
```

4) We need to install a driver to send commands over USB to the Pico from Picotool. Here are some instructions from How to Build OpenOCD and Picotool for the Raspberry Pi Pico on Windows - Shawn Hymel:

Put your Pico board into bootloader mode (press and hold BOOTSEL and plug in the USB cable). It should enumerate on your computer as a storage drive.

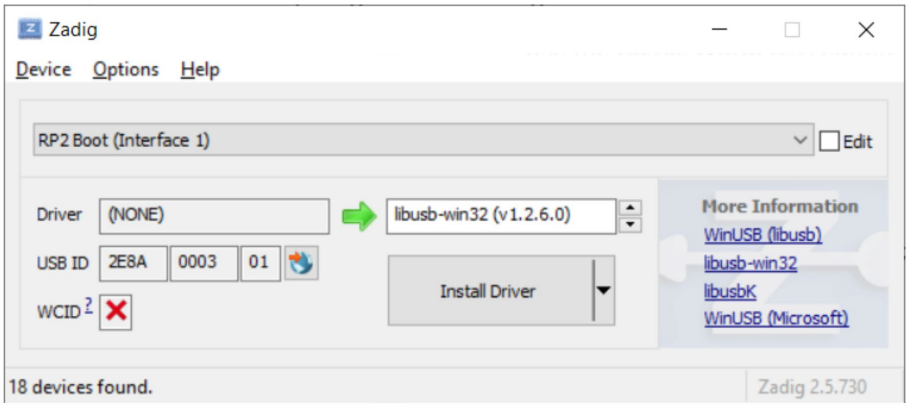
Head to https://zadig.akeo.ie and download Zadig



Run Zadig. Select **Options > List All Devices**

You should see 2 Pico devices in the drop-down menu: RP2 Boot (Interface 0) and RP2 Boot (Interface 1). **Select RP2 Boot (Interface 1)**

The driver should be listed as *(NONE)*. Select **libusb-win32** (some version) as the replacement driver.



Click **Install Driver**. When it's done the current driver should be listed as '**libusb-win32**'.

If you break something in Zadig (which we did!), you can fix it. For example, the Pico will not enumerate as a mass storage device drive in Bootloader mode. Zadig will show something other than "USBSTOR" as the driver for RP2 (Interface 0).

Here is how to delete the bad drivers and let Windows fix the mess you made:

1. Put Pico into Bootloader mode and open Device Manager
2. Click **View > Devices by container**
3. Expand RP2 Boot
4. For all entries under RP2 Boot:
   - Right click on entry and select uninstall device
   - Check "Delete the driver software for this device (if asked)
   - Click Uninstall
5. Unplug Pico
6. Hold BOOTSEL and plug it back in to put it into bootloader mode again
7. Windows should automatically reinstall all the correct drivers

5) Save a copy of doom1.whx to /c/VSARM/sdk/pico/picotool/build
6) With the Pico in bootloader mode (hold bootsel down), open the Git SDK window and enter the following commands:

```
/c/VSARM/sdk/pico/picotool/build/picotool.exe
picotool load -v -t bin doom1.whx -o 0x10042000.
```

The second file will have been flashed to the Pico and doom is ready!

ENJOY!