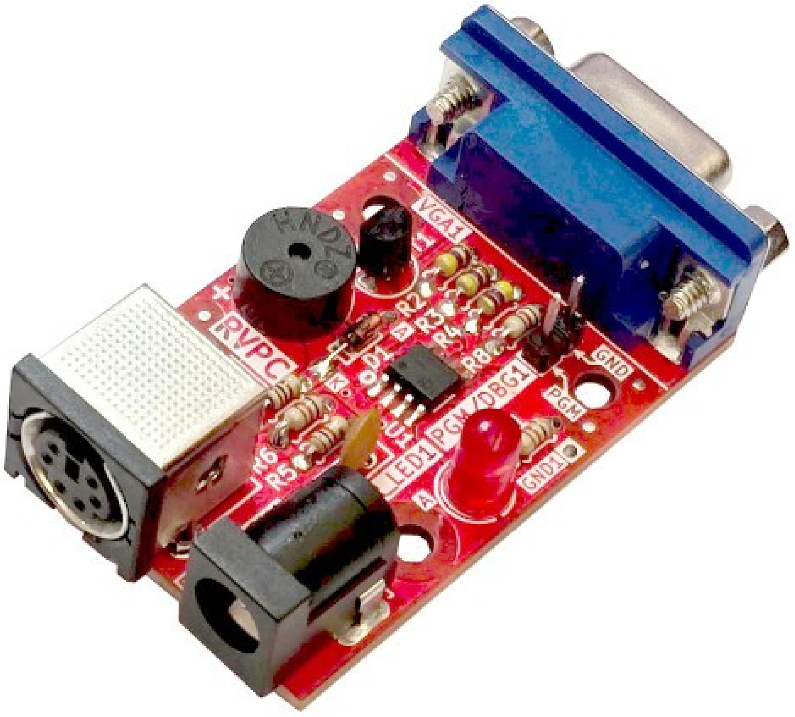


RVPC

User Manual

Document v1.3 Nov 2024

www.olimex.com

Table of Contents

What is RVPC?

RVPC is an attempt to produce a very low-cost EURO 1.00 educational computer with RISC-V processor which has everything one complete computer has: keyboard input, VGA display output, and audio output.

The idea of RVPC was born from this TuxCon 2024 talk https://youtu.be/YlYE9a7zsqY.

The goal set was:

1. Easy to solder DIY kit;
2. Complete all in one RISC-V computer with bare minimum Woz like monitor which will allow you to learn the RISC-V instructions by poking, peeking and disassembling the memory
3. Price of EUR 1.00!

Here is the result:

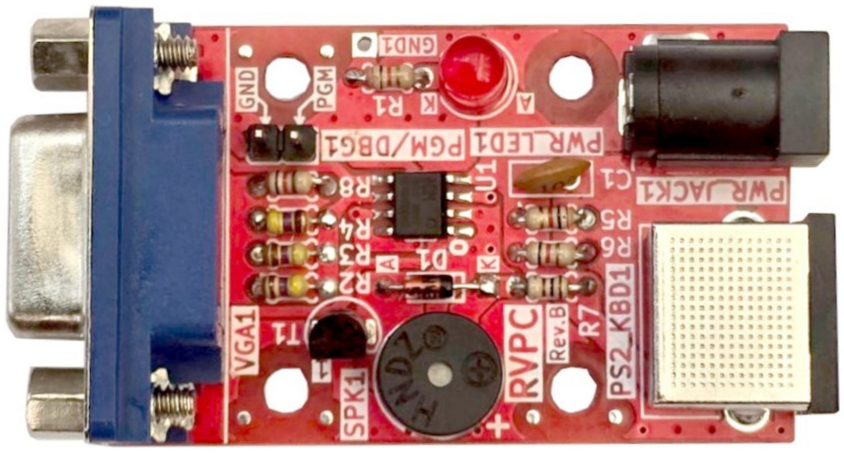
CH32V003 in SO8 package – for easy soldering was chosen. It has just 6 GPIOs:

- PS2 takes two GPIOs

- VGA takes three GPIOs – Vsync, Hsync and RGB

- Audio buzzer is connected to the last GPIO

All done using beginner-friendly PTH components!

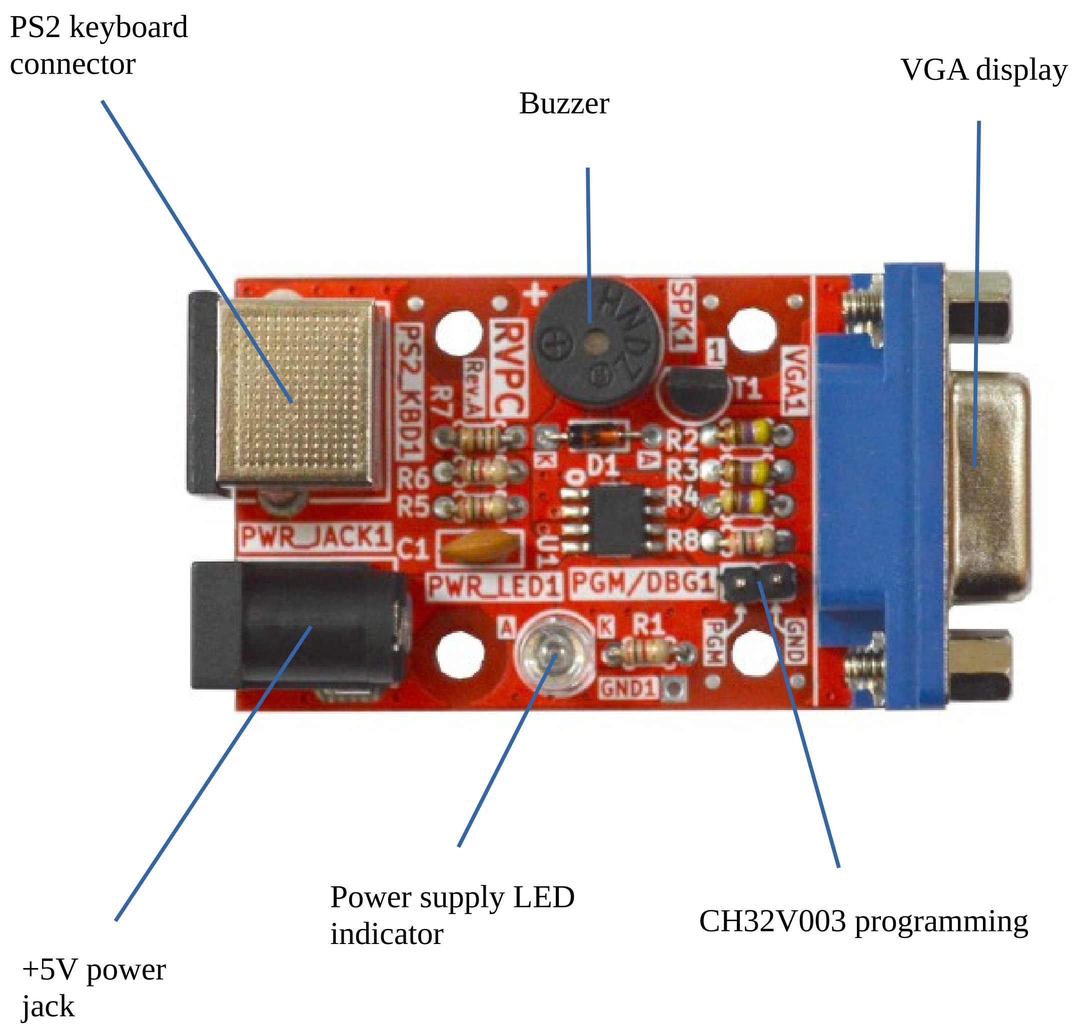


Order codes for RVPC and accessories:

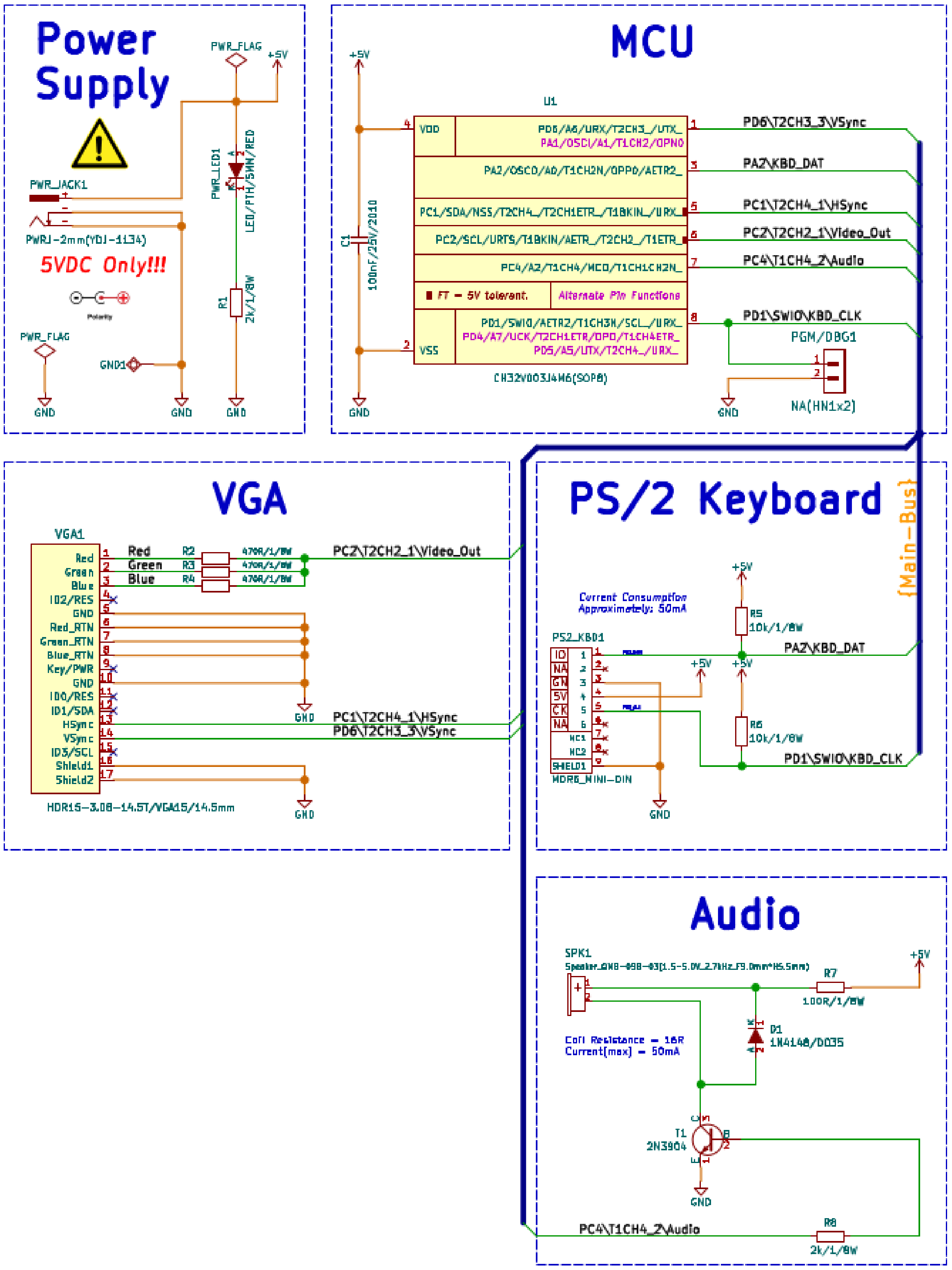
RVPC	Do It Yourself soldering kit
SY0605E	5V power supply adapter
PS2-KEYBOARD	PS2 keyboard
ESP32-S2-DevKitLiPo-USB	ESP32-S2 development board which can be used as CH32V003 programmer
USB-CABLE-A-MICRO-1.8M	USB cable for the programmer

HARDWARE

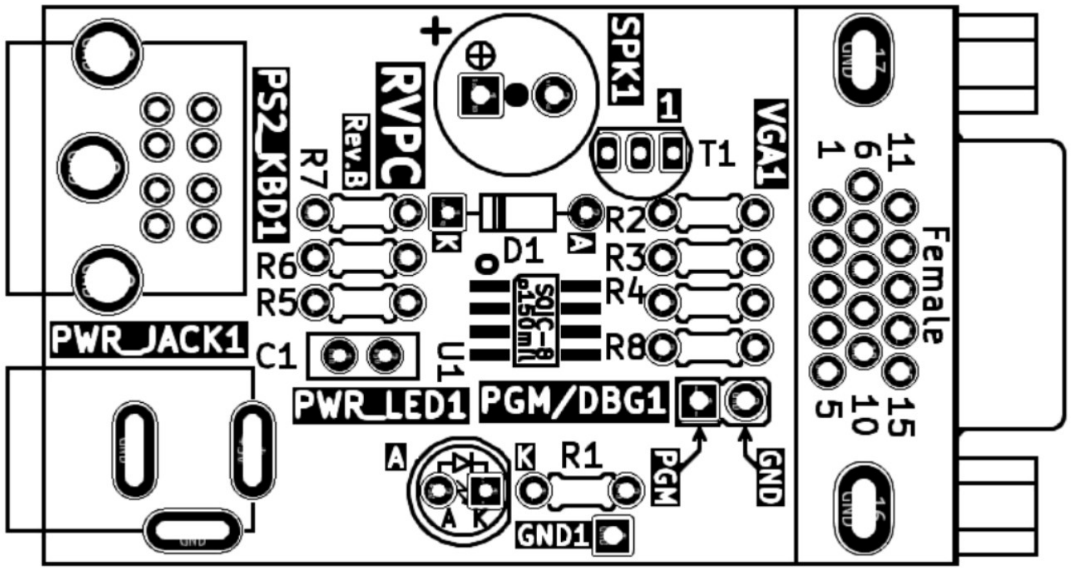
RVPC layout:



RVPC schematic:



RPVC assembly cheat sheet:



Soldering Order	Quantity	Reference	Value	Description
1	1	U1	CH32V003J4M6(SOP8)	Microcontroller point to silk point
2	1	R7	100R/1/8W	BROWN-BLACK-BROWN-GOLD
3	3	R2, R3, R4	470R/1/8W	YELLOW-PURPLE-BROWN
4	4	R1, R5, R6, R8	2k/1/8W	RED-BLACK-RED-GOLD
5	1	D1	1N4148/DO35	The black line mark to K(square pad)
6	1	C1	100nF/25V/2010	104
7	1	T1	2N3904	N-P-N transistor
8	1	PWR_LED1	LED/PTH/5MM/RED	Longest pin to A(round pad)
9	1	SPK1	QMB-09B-03(1.5-5.0V_2.7kHz)	Speaker
10	1	PGM/DBG1	HN1x2	2 pin header
11	1	PS2_KBD1	MDR6_MINI-DIN	PS2 connector
12	1	VGA1	HDR15-3.08-14.5T/VGA15	VGA connector
13	1	PWR_JACK1	PWRJ-2mm(YDJ-1134)	Power Jack

After assembly remember to close jumper GRN1 (solder together the pads of GRN1) to enable the green VGA signal.

Necessary tools:

Wire Cutters:	we recommend PGC-TR25 these are sharp and light
Tweezers:	we recommend PGC-00SA
Soldering iron:	CHN-SLD802 is budged solution, SLD-FAST-75W is professional solution
Soldering wire:	we recommend Solder-Wire-SAC0307-0-8

General tips for soldering:

1. Switch on the soldering iron, setup the working temperature to 350 C. Wait until the soldering iron reaches the temperature – there is a LED indicator which will pulse when the temperature is reached;

2. Before soldering, clean the soldering tip with wet sponge from the black residues;

3. Never touch the heated soldering tip or body with bare hands;

4. Be careful to not touch surrounding objects with the soldering iron heated body or tip (cables, table, cloths, etc);

5. Do not leave the soldering iron unattended;

6. Place the electronic component on its place, watch out if there is polarity or orientation;

7. Touch the component pad with the soldering tip which you want to solder and wait 3-4 seconds to heat up;

8. Feed a little from the soldering wire between the soldering tip and the pad until the component lead is flooded with tin and it’s shinny and glossy;

9. If the soldering is not shinny but dull please re-solder with colophon.

SOFTWARE:

Below is our setup under Linux:

Install – packages

$ apt-get install build-essential libnewlib-dev gcc-riscv64-unknown-elf libusb-1.0-0-dev libudev-dev gdb-multiarch

Install – Visual Studio Code

Described here: https://code.visualstudio.com/docs/setup/linux

Install – Platform IO

Described here: https://platformio.org/install/ide?install=vscode

Install – CH32V-Platform

https://github.com/Community-PIO-CH32V/ch32-pio-projects?tab=readme-ov-file#installing-the-ch32v-platform

by default the platformio generates only .elf file, to build firmware.bin and firmware.elf select

> PlatformIO > PROJECT TASKS > Default > Advanced > Verbose build

Sample Beeper project is in RVPC repository.

Prepare the CH32V003 programmer

ESP32-S2-DevKitLiPo-USB can be used as programmer.

The repo with all files required to prepare the ESP32-S2 board as programmer is here – after you have the resources the algorithm to prepare the programmer is:

1. Press and hold the Boot button and connect the USB cable, the yellow LED will stay ON

check with

```
$ ls /dev/ttyA*
```

which is the ttyACM it's usually 0 or 1

execute this command:

```
$ python3 ./rvpc/esptool/esptool.py -p /dev/ttyACM0 -b 460800 --before=no_reset --after=no_reset write_flash --flash_mode dio --flash_freq 80m --flash_size 4MB 0x1000 ./rvpc/esp32s2/bootloader.bin 0x10000 ./rvpc/esp32s2/usb_sandbox.bin 0x8000 ./rvpc/esp32s2/partition-table.bin
```

check if the programmer is already OK with

```
$ dmesg
```

you have to see this message:

```
hid-generic 0003:303A:4004.0015: input,hidraw5: USB HID v1.11 Gamepad [CNLohr ESP32-S2 CH32V003Programmer] on usb-0000:00:14.0-2/input0
```

which means the ESP32-S2-DevKitLipo-USB now acts as a programmer and can be used with the demo project above from PlatformIO, but first you have to enable it with:

```
$ sudo cp ./rvpc/tools/ch32v003fun/minichlink/99-minichlink.rules /etc/udev/rules.d/
```

```
$ sudo udevadm control --reload-rules && sudo udevadm trigger
```

Now you can use GPIO6 and GND to connect to RVPC programming connector PGM-GND

Now CH32V003 flashing will work directly from PlatformIO.

If you want to use command line this is the command:

```
./rvpc/tools/ch32v003fun/minichlink/minichlink -w ./firmware.bin 0x08000000
```

Notice that if you wish to use the command line programming you'd need to build the minichlink, install these libraries:

```
sudo apt-get install libudev-dev libusb-1.0-0-dev
```

then navigate to:

\rvpc\tools\ch32v003fun\minichlink and type `make`

Using Raspberry Pi Pico as a programmer

Raspberry Pi Pico can be used as a programmer.

The programmer firmware is located here: https://github.com/aappleby/picorvd. Make sure you install the dependencies, especially GDB as you'll need it for uploading the RVPC firmware later.

You can either build it yourself, or use a precompiled binary from their CI. The prebuilt binaries can be found in the **Actions** tab on the repository, selecting the most recent commit and scrolling to the bottom of the page where the **Artifacts** section is. There should be a single artifact named **PicoRVD.uf2**, which is the firmware build. If the downloaded file is a ZIP, make sure to unzip the firmware and discard the ZIP file.

Get your RPi Pico, hold down the BOOTSEL button and plug the USB cable into your computer. A new USB mass storage device should appear, copy the PicoRVD firmware file into it and wait for a moment. You don't need to safely eject/remove it, since it will disconnect itself as soon as it receives the firmware file. It will then immediately boot the PicoRVD firmware and the Pico will be ready for flashing your RVPC.

RVPC firmware uploading is done by the Remote debugging feature of GDB. In order to automate it, you can use the following shell script:

```
#!/bin/bash

set -e

if [ -z "${1}" ]
then
   echo 'usage: '"${0}" path/to/firmware.elf >&2
   exit 1
fi

gdb-multiarch -ex 'target extended-remote /dev/ttyACM0' -ex 'load' -ex
'detach' -ex 'quit' "${1}"
```

To upload a new RVPC firmware, you need the .elf file, not the .bin file as GDB does not have enough information how to upload it. Before uploading a new firmware, restart your RVPC by power cycling it, then power-cycle your Pico and in case it appears as mass storage device, upload the PicoRVD firmware again. Then use the shell script, adjusting the /dev/ttyACM0 port according to the actual serial port name (you can find it in dmesg). After successfully uploading a new RVPC firmware, you need to power-cycle the RVPC, in order to start the program.

Create project:

If you create a new project to enable the ESP32-S2 programmer you should edit platformio.ini and add this line

```
upload_protocol = minichlink
```

This is not needed for demo projects, since it’s already added to the demo projects at our GitHub.

RVPC monitor demo RVMON (default)

The chip of RVPC comes programmed with this RVPC monitor demo. All demos can be found here:

https://github.com/OLIMEX/RVPC/tree/main/SOFTWARE

This is WOZ-like monitor which allow you to display memory, write to memory and execute code.
The list of all commands are displayed with “?”

Towers of Hanoi demo

This is original game created by Curtis Whitley who wrote the VGA display code for RVPC.

Towers of Hanoi Interactive demo

This is modified version which uses the keyboard so user can tell which disk to which tower go.

TETRIS demo

This is the very popular game made for RVPC.

Revision History

Revision 1.0 June 2024

Revision 1.1 September 2024

Revision 1.2 October 2024 add assembly instructions

Revision 1.3 November 2024 improved instructions