RISC-V MCU development boards

Table of contents

Introduction

"How do I get my feet wet with RISC-V?" is a very common question, which is often answered "Buy a RISC-V development board, and practice bare metal development."

This document provides guidance on part selection, as well as pointers to useful resources. It is not a comprehensive guide of RISC-V MCU, but rather a short list of easily approachable parts. To be listed in this guide, parts **must** meet several criteria:

- have decent English documentation,
- have open-source or freely downloadable supporting software,
- be easily available from anywhere in the world, e.g. through AliExpress or LCSC,
- have cheap development boards as easily available as the chips,
- not require a Chinese mobile phone number to download software and documentation.

License

Manufacturer selection

MCU manufacturers have different product strategies, leading to different product ranges. An easy and efficient method is to select a manufacturer whose strategy matches your needs, and then see which of their parts best suits your project.

Manufacturer	*Wide supply voltage range*	*Wide package choice*	*Lots of I/O pins*	*BLE*	*WiFi*	*Zigbee*
BouffaloLab				x	x	x
Espressif				x	x	x
GigaDevice			x			
WCH	x	x	x	x		

Note: manufacturers are sorted in alphabetical order.

BouffaloLab

BouffaloLab only manufactures RISC-V chips, so they're undoubtedly committed to this platform. They have a wide range of IoT-oriented MCU, some with SiFive IP, others with T-Head IP. The documentation is good, but their SDK and code examples are a bit chaotic.

However, Pine64 has released a well-documented development board (PineCone) and a module (PineNut), both based on the BL602 MCU, and Lee Lup Yuen has produced excellent training material on this board, which comes in complement to the community built by Pine64 around their products.

It is highly recommended to become quite familiar with the PineCone/BL602's ecosystem before attempting to use other chips from BouffaloLab (e.g. BL702, BL616), with which you will be left a bit on your own.

Third-party development board: PineCone

Purchase link: https://pine64.com/product/pinecone-bl602-evaluation-board/

Documentation: https://wiki.pine64.org/wiki/PineCone

Review: https://lupyuen.github.io/articles/pinecone

Tutorials: https://lupyuen.github.io/articles/book

Documentation, SDK and tools

Data sheets & reference manuals: https://github.com/bouffalolab/bl_docs

SDK:

https://github.com/bouffalolab/bl_mcu_sdk

https://github.com/bouffalolab/bl_iot_sdk

https://dev.bouffalolab.com/home/

Flashing tool (BlDevCube): https://github.com/bouffalolab/flash_tools

GitHub: https://github.com/bouffalolab

Lots of interesting information not found in BouffaloLab documents: https://github.com/pine64/

Espressif

Espressif is famous for their Xtensa-based ESP32 and ESP32-S3 modules, but they also announced their new developments would be made on RISC-V, so interesting things are to be expected from them. Like BouffaloLab, they focus on IoT-oriented MCU.

Their current RISC-V product range includes the ESP8685, ESP32-C2, ESP32-C3, and the brand new ESP32-C6. The ESP32-C3 being quite cheap, its "downgraded" versions (ESP8685 & ESP32-C2) don't make sense unless you produce cost-sensitive devices (high volume, low margin).

Official development board: ESP32-C3-DevKitM-1

Purchase link: https://www.aliexpress.com/item/1005003989099547.html

Documentation: https://docs.espressif.com/projects/esp-idf/en/latest/esp32c3/hw-reference/esp32c3/user-guide-devkitm-1.html

Third-party development board: ESP32-C3FH4

Purchase link: https://www.aliexpress.com/item/1005004960064227.html

Documentation: https://github.com/WeActStudio/WeActStudio.ESP32C3CoreBoard

Third-party development board: YD-ESP32-C3

Purchase link: https://www.aliexpress.com/item/1005003613170790.html

Documentation: http://www.vcc-gnd.com/

Documentation, SDK and tools

Data sheets & reference manuals: https://www.espressif.com/en/support/documents/technical-documents?keys=&field_type_tid%5B%5D=785

(relevant documents are *ESP32-C3 Datasheet*, *ESP32-C3 Technical Reference Manual*, and *ESP32-C3 Series SoC Errata*)

SDK: https://www.espressif.com/en/support/download/all?keys=&field_type_tid%5B%5D=785

(SDK name is *ESP-IDF*)

GitHub: https://github.com/orgs/espressif

GigaDevice

While GigaDevice have largely developed their ARM MCU portfolio in the past years, the GD32VF103 is still their only one RISC-V MCU. It's a very interesting part, but unfortunately out-of-stock to date (Feb. 2023), and GigaDevice not being committed to RISC-V doesn't bode well of its future. Anyway, there are still development boards available for the GD32VF103 and plenty of articles and tutorials can be found on the web, so you might want to give it a try.

Official development board: GD32VF103C-START

Purchase link: https://www.lcsc.com/product-detail/C432220.html

Note: this evaluation board includes GigaDevice's GDLink programmer.

Third-party development board: Longan Nano

Purchase link: https://www.aliexpress.com/item/1005002542610332.html

Documentation: https://github.com/sipeed/Longan-DOC

Downloads: https://dl.sipeed.com/shareURL/LONGAN/Nano

Note: an additional JTAG adapter is needed to program the chip. If you don't already have one (e.g. JLink), you can buy Sipeed's USB-JTAG/TTL.

Purchase link: https://www.aliexpress.com/item/1005002714665888.html

Documentation, SDK and tools

Download URL: https://www.gd32mcu.com/en/download?kw=GD32VF1

Data sheets & reference manuals: relevant documents are *GD32VF103xx Datasheet* and *GD32VF103 User Manual*.

SDK: relevant archive is *GD32VF103 Firmware Library*.

WCH

WCH provides a much wider range of RISC-V MCU than any other manufacturer. For this reason, each chip has its own section below.

WCH's offer is divided in 2 families, the CH32V*xxx* series, which are the RISC-V equivalents of their ARM CH32F*xxx* series, and the CH*xxx* series, which use a slightly different peripheral set.

An interesting thing to note is the CH32V003 doesn't have an ARM equivalent, which can be a hint that, like Espressif, WCH will continue to concentrate their efforts on their RISC-V products.

The following table presents the main differences between WCH's RISC-V MCU:

Part	*Features*	*Max. frequency*	*Supply voltage*	*Notes*
CH32V003	RV32EC	48 MHz	2.7-5.5V	Low pin count (8, 16, 20)
CH32V103	RV32IMAC	80 MHz	2.7-5.5V	
CH32V203	RV32IMAC	144 MHz	2.4-3.6V	
CH32V208	RV32IMAC	144 MHz	2.4-3.6V	BLE
CH32V30*x*	RV32IMAFC	144 MHz	2.4-3.6V	Ethernet
CH582	RV32IMAC	80 MHz	2.3-3.6V	BLE, ultra low power

WCH also makes the CH573 which is comparable to but, in my opinion, less interesting than the CH582, and the CH569, which is specifically intended for communication applications and supports USB 3.0.

For further details, here is a selection table covering the whole WCH offer:
https://special.wch.cn/en/mcu/

Note: all WCH MCU SDK include the schematics of the official evaluation boards as reference design.

GitHub: https://github.com/openwch

Programmer / debugger

A proprietary programmer / debugger is needed to flash WCH chips, the WCH-LinkE.

Purchase link: https://www.aliexpress.com/item/1005004881582037.html

One is already included with the official evaluation kit for the CH32V003, and the CH32V003 + CH32V203 evaluation kit combo.

The flashing software, WCHISPTool, is unfortunately Windows-only and doesn't work under Wine.

Download link: http://wch-ic.com/downloads/WCHISPTool_Setup_exe.html

There's a portable command-line flashing tool, but it uses configuration files generated by WCHISPTool...

MounRiver IDE

MounRiver Community Edition is an Eclipse-based IDE supporting all RISC-V and ARM WCH MCU, available for Windows, Linux and Mac. It includes a (third?) portable flashing tool, which is convenient if you don't use Windows (see above), particularly to unlock the flash write protection to be able to program your chip.

WCH provides modified versions of OpenOCD and GCC with MounRiver, but these toolchains can also be downloaded separately, e.g. for installation on a continuous integration server.

Download link: http://www.mounriver.com/

CH32V003

Official development board

Purchase link: https://www.aliexpress.com/item/1005004895791296.html

Third-party development board: nanoCH32V003

Purchase link: https://www.aliexpress.com/item/1005005221751705.html

Documentation: https://github.com/wuxx/nanoCH32V003

Documentation and SDK

Data sheet: http://wch-ic.com/downloads/CH32V003DS0_PDF.html

Reference manual: http://wch-ic.com/downloads/CH32V003RM_PDF.html

Processor manual: http://wch-ic.com/downloads/QingKeV2_Processor_Manual_PDF.html

SDK: https://www.wch.cn/downloads/CH32V003EVT_ZIP.html

CH32V103

Official development board

Purchase links:

https://www.lcsc.com/product-detail/C2943983.html

https://www.lcsc.com/product-detail/C2943982.html

https://www.aliexpress.com/item/1005004607642695.html

Third-party development board: BluePill+ CH32V103

Purchase link: https://www.aliexpress.com/item/1005001474741936.html

Documentation: https://github.com/WeActStudio/WeActStudio.BluePill-Plus-CH32

Documentation and SDK

Data sheet: http://wch-ic.com/downloads/CH32V103DS0_PDF.html

Reference manual: http://wch-ic.com/downloads/CH32xRM_PDF.html

Processor manual: http://wch-ic.com/downloads/QingKeV3_Processor_Manual_PDF.html

SDK: https://www.wch.cn/downloads/CH32V103EVT_ZIP.html

CH32V203

Official development board

Purchase link: https://www.aliexpress.com/item/1005004493040662.html

Third-party development board: nanoCH32V203

Purchase link: https://www.aliexpress.com/item/1005004908206775.html

Documentation: https://github.com/wuxx/nanoCH32V203

Documentation and SDK

Data sheet: http://wch-ic.com/downloads/CH32V203DS0_PDF.html

Reference manual: http://wch-ic.com/downloads/CH32FV2x_V3xRM_PDF.html

Processor manual: http://wch-ic.com/downloads/QingKeV4_Processor_Manual_PDF.html

SDK: https://www.wch.cn/downloads/CH32V20XEVT_ZIP.html

CH32V208

Official development board

Purchase link: https://www.aliexpress.com/item/1005004924242063.html

Documentation and SDK

Data sheet: http://wch-ic.com/downloads/CH32V208DS0_PDF.html

Reference manual: http://wch-ic.com/downloads/CH32FV2x_V3xRM_PDF.html

Processor manual: http://wch-ic.com/downloads/QingKeV4_Processor_Manual_PDF.html

SDK: https://www.wch.cn/downloads/CH32V20XEVT_ZIP.html

CH32V305

Third-party development board: nanoCH32V305

Purchase link: https://www.aliexpress.com/item/1005005033298927.html

Documentation: https://github.com/wuxx/nanoCH32V305

Documentation and SDK

Data sheet: http://wch-ic.com/downloads/CH32V307DS0_PDF.html

Reference manual: http://wch-ic.com/downloads/CH32FV2x_V3xRM_PDF.html

Processor manual: http://wch-ic.com/downloads/QingKeV4_Processor_Manual_PDF.html

SDK: https://www.wch.cn/downloads/CH32V307EVT_ZIP.html

CH32V307

Official development board

Purchase link: https://www.lcsc.com/product-detail/C2943980.html

Third-party development board: YD-CH32V307VC

Purchase link: https://www.aliexpress.com/item/1005004367173443.html

Documentation: http://www.vcc-gnd.com/

Documentation and SDK

Data sheet: http://wch-ic.com/downloads/CH32V307DS0_PDF.html

Reference manual: http://wch-ic.com/downloads/CH32FV2x_V3xRM_PDF.html

Processor manual: http://wch-ic.com/downloads/QingKeV4_Processor_Manual_PDF.html

SDK: https://www.wch.cn/downloads/CH32V307EVT_ZIP.html

CH582

Official development board

Purchase links:

https://www.lcsc.com/product-detail/C2943981.html

https://www.aliexpress.com/item/1005005060737000.html

Third-party development board: YD-CH58x

Purchase link: https://www.aliexpress.com/item/1005004878787042.html

Documentation: http://www.vcc-gnd.com/

Documentation and SDK

Data sheet: http://wch-ic.com/downloads/CH583DS1_PDF.html

Processor manual: http://wch-ic.com/downloads/QingKeV4_Processor_Manual_PDF.html

SDK: https://www.wch.cn/downloads/CH583EVT_ZIP.html

Alternatives to WCH tools

If you would like to use mainstream GCC instead of WCH's, or to try an open-source flashing tool, the following Reddit thread will be of interest to you:

https://www.reddit.com/r/RISCV/comments/115u6i9/comment/j94xvpq/?utm_source=reddit&utm_medium=web2x&context=3