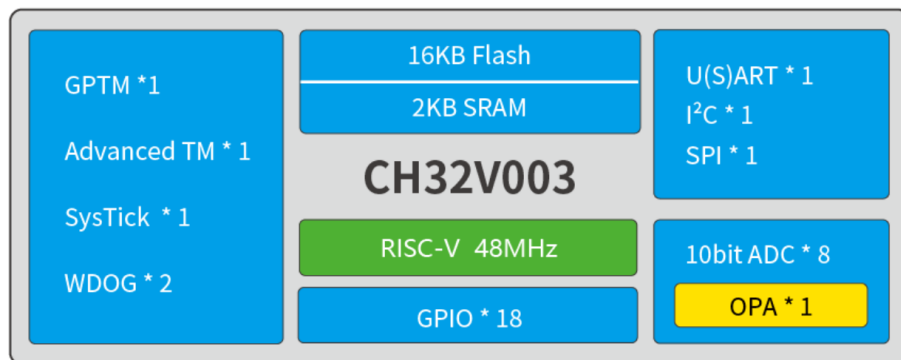


Chapter 1: Getting Started with CH32v003

1.1 Introduction:

CH32V003 series is based on QingKe RISC-V2A core design of industrial-grade general-purpose microcontroller, support 48MHz system main frequency, with wide voltage, 1-wire serial debug interface, low-power consumption, ultra-small package, etc. CH32V003 series built-in a group of DMA controller, a group of 10-bit ADC, a group of op-amp comparators, multiple timers and standard communication interfaces USART, I2C, SPI, etc.



Main Features

- QingKe 32-bit RISC-V2A processor, supporting 2 levels of interrupt nesting
- Maximum 48MHz system main frequency 2KB SRAM,
- 16KB Flash Power supply voltage: 3.3/5V
- Multiple low-power modes: Sleep, Standby
- Power on/off reset, programmable voltage detector
- 1 group of 1-channel general-purpose DMA controller
- 1 group of op-amp comparator
- 1 group of 10-bit ADC
- 1×16-bit advanced-control timer, 1×16-bit general-purpose timer
- 2 WDOG, 1×32-bit SysTick
- 1 USART interface,
- 1 group of I2C interface,
- 1 group of SPI interface 18 I/O ports, mapping an external interrupt 64-bit chip
- unique ID 1-wire serial debug interface (SDI)

The C-Type USB port cannot be used directly for programming purposes. We need a programmer/debugger device for to download program code on the microcontroller. There are different types of programming devices given by WCH but we have use one compatible with our microcontroller.

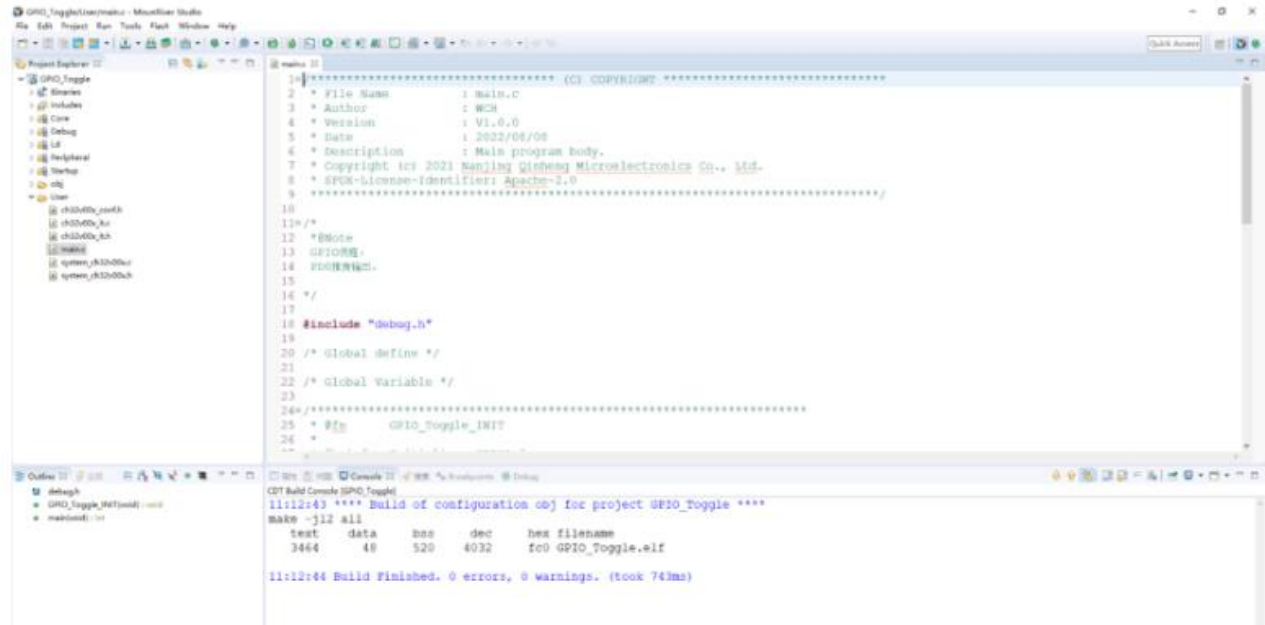
Table 6 Link supported chip model

Common chip models	WCH-Link	WCH-LinkE	WCH-DA PLink
CH32V003	×	√	×
CH32V10x/CH32V20x/cCH32V30x/CH569/CH573/CH583	√	√	×
CH32F10x/CH32F20x/CH579/friendly chips that support SWD protocol	√	√	√
friendly chips that support JTAG interface	×	√	√

For Programming CH32v003 we will be using the WCH-LINKE.

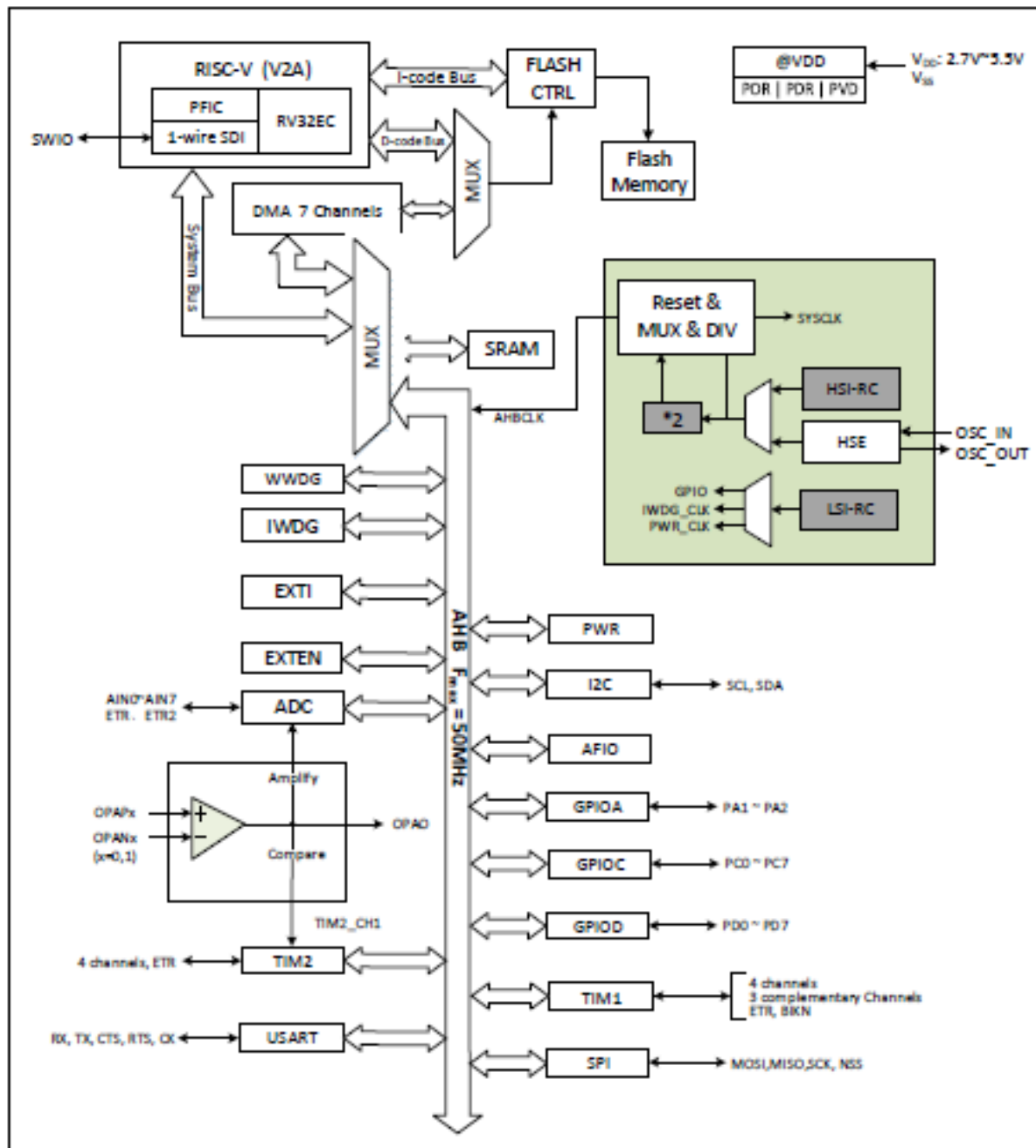


1.2 Programming Environment: Mounriver Studio

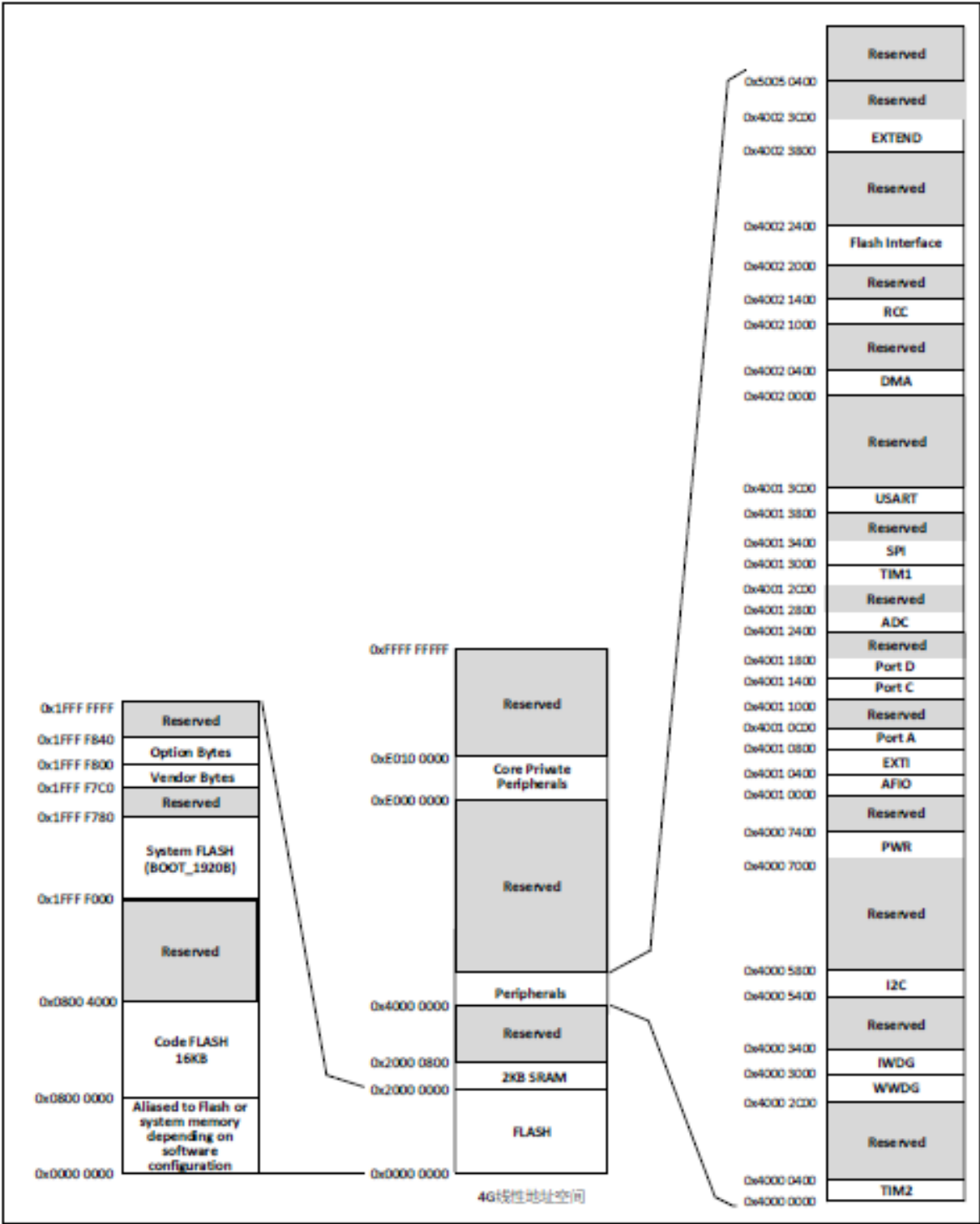


1.3 Bus Architecture

The CH32V003 series is designed based on the RISC-V instruction set, and its architecture interacts the core, arbitration unit, DMA module, SRAM storage and other parts through multiple buses. The design integrates a general-purpose DMA controller to reduce the CPU load and improve access efficiency, as well as data protection mechanisms, automatic clock switching protection mechanisms and other measures to increase system stability. The system block diagram is shown in Figure below.



1.4 Memory Architecture



- Memory Allocation Built-in 2KB SRAM, starting address 0x20000000, supports byte, half-word (2 bytes), and full-word (4 bytes) access.
- Built-in 16KB program Flash memory (CodeFlash) for storing user applications.
- Built-in 1920B System memory (bootloader) for storing the system bootloader (factory-cured bootloader).
- Built-in 64B space for vendor configuration word storage, factory-cured and unmodifiable by users.
- Built-in 64B space for user-option bytes storage.

1.5 Pinouts and Pin Definition

CH32V003F4P6

1	PD4/A7/UCK/T2CH1ETR/OPO/T1CH4ETR_	PD3/A4/T2CH2/AETR/UCTS/T1CH4_	20
2	PD5/A5/UTX/T2CH4_/URX_	PD2/A3/T1CH1/T2CH3_/T1CH2N_	19
3	PD6/A6/URX/T2CH3_/UTX_	PD1/SWIO/AETR2/T1CH3N/SCL_/URX_	18
4	PD7/NRST/T2CH4/OPP1/UCK_	PC7/MISO/T1CH2_/T2CH2_/URTS_	17
5	PA1/OSCI/A1/T1CH2/OPN0	PC6/MOSI/T1CH1CH3N_/UCTS_/SDA_	16
6	PA2/OSCO/A0/T1CH2N/OPP0/AETR2_	PC5/SCK/T1ETR/T2CH1ETR_/SCL_/UCK_/T1CH3_	15
7	VSS	PC4/A2/T1CH4/MCO/T1CH1CH2N_	14
8	PD0/T1CH1N/OPN1/SDA_/UTX_	PC3/T1CH3/T1CH1N_/UCTS_	13
9	VDD	PC2/SCL/URTS/T1BKIN/AETR_/T2CH2_/T1ETR_	12
10	PC0/T2CH3/UTX_/NSS_/T1CH3_	PC1/SDA/NSS/T2CH4_/T2CH1ETR_/T1BKIN_/URX_	11

Note: The multiplexed functions in the pin diagram are abbreviated.

Example: A: ADC, A7 (ADC_IN7)

T: TIME, T2CH4 (TIM2_CH4)

U: USART, URX (USART_RX)

OP: OPA, OPO (OPA_OUT), OPP1 (OPA_P1)

OSCI (OSCIN)

OSCO (OSCOOUT)

SDA (I2C_SDA)

SCL (I2C_SCL)

SCK (SPI_SCK)

NSS (SPI_NSS)

MOSI (SPI_MOSI)

MISO (SPI_MISO)

AETR(ADC_ETR)

1.6 Pin Description and Pin Alternate Function

Refer to the datasheet for detailed description of pins and their alternate functions.

1.7 Development Setup for CH32V003

MounRiver Studio IDE: The MounRiver IDE is a cross-platform IDE for developing applications for WCH RISC-V MCUs, including the CH32V003. It is available for Windows, macOS, and Linux.

How to install MounRiver IDE for CH32V003

The MounRiver IDE is a powerful and easy-to-use IDE for developing applications for the CH32V003 MCU. By following the steps, you can install and set up MounRiver IDE for CH32V003 development, create new projects, compile and program your applications, and debug your code.

For Installation follow the steps mentioned below: Download the latest version of MounRiver IDE. If you are facing any issue in downloading IDE from their website, you can download from here MounRiver IDE V1.9 and upgrade it.

Please note that as per WCH, software support is the best available for Windows OS and good on Linux but for macOS software is very little, so be careful

Windows: Extract the downloaded archive to a location on your computer. Open the MounRiver IDE directory and double-click on the MounRiverIDE.exe file to launch the IDE.

macOS: Extract the downloaded archive to a location on your computer. Open the MounRiver IDE directory and double-click on the MounRiverIDE.app file to launch the IDE.

Linux: Extract the downloaded archive to a location on your computer. Open a terminal and navigate to the MounRiver IDE directory. Run the following command to launch the IDE:
./MounRiverIDE

Installation of the MounRiver IDE is quite simple as any other software.

1.8 Creating a New Project:

To create a new project in MounRiver IDE for CH32V003 development, follow these steps:

1. Click on the **File** menu and select **New > Project**.
2. In the **New Project** dialog box, select the **MounRiver Project** and click **Next**.
3. In the **Project Properties** dialog box, enter a name for your project and select the CH32V003 MCU from the **Device** list.
4. Click **Finish** to create the new project.

Compiling and programming your application

To compile and program your application to the CH32V003 MCU, follow these steps:

1. Click on the **Project** menu and select **Build Project**.
2. Once the project has been compiled successfully, click on the **Flash** menu and select **Download**.
3. In the **Download** dialog box, select the compiled binary file from the **File** list and click **Open**.
4. Click **Download** to program the binary file to the CH32V003 MCU.

Debugging your application

To debug your application, follow these steps:

1. Connect the CH32V003 board to your computer using a USB cable.
2. Click on the **Debug** menu and select **Start Debugging**.
3. In the **Debug** perspective, you can set breakpoints, step through your code, and inspect the values of variables.

