武汉芯源半导体有限公司
WUHAN XINYUAN SEMICONDUCTOR CO., LTD

CW32 ISP protocols used in the Bootloader

Application note

Rev 1.0

www.whxy.com

Introduction

A section of the CW32 microcontroller on-chip FLASH memory is used to store the BootLoader , which is factory programmed on the chip. The user can use the ISP mode provided by the BootLoader to easily erase and burn the CW32 microcontroller on-chip FLASH main memory via the UART serial port.

This application note will describe how to enter the CW32 microcontroller ISP mode, the ISP protocol used and the details of each command supported.

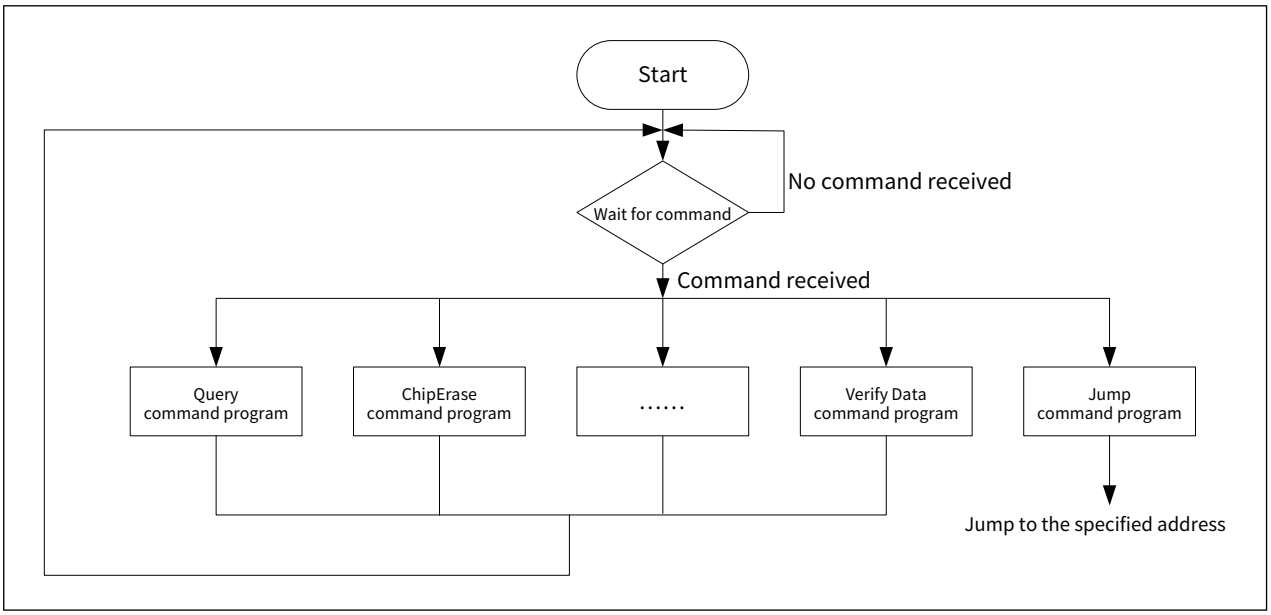
Contents

1 How to enter ISP mode on the chip

- Steps for a chip with a BOOT pin:
  1. Set the chip in RESET state;
  2. Supply a high level to the chip's BOOT pin;
  3. Release the chip from RESET state;
  4. The chip enters ISP mode.
- Steps for chip without BOOT pin:
  1. Set the chip in RESET state;
  2. Supply a 50KHz square wave to the chip's RXD (SWDIO);
  3. Release chip from RESET state and delay 5ms;
  4. The chip enters ISP mode.

2 ISP mode workflow

Figure 2-1 ISP mode workflow diagram



When the CW32 microcontroller enters ISP mode, the system will wait for the serial port to receive commands. Once the system receives the command, it will execute the corresponding program operation according to the command type.

3 ISP communication protocol format

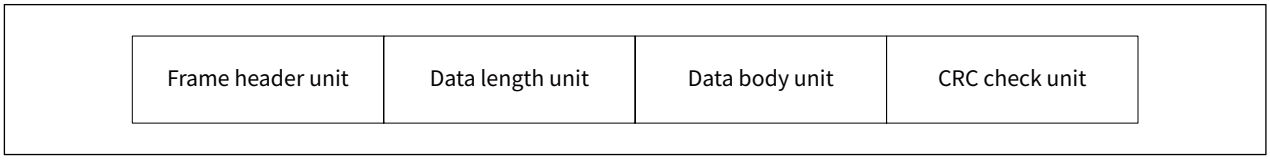
3.1 Data transmission requirements

Asynchronous half-duplex communication with 8 data bits, 1 stop bit, no parity bit and an initial rate of 115200 BPS.

3.2 Data format

The protocol interacts in the form of data frames. A complete data frame consists of four parts: the frame header unit, the data length unit, the data body unit and the CRC check unit, as shown in the following diagram:

Figure 3-1 Data frame format



- Frame header unit

  1 byte in length, indicating the start of a data frame, fixed as hexadecimal number 0x65.

- Data length unit

  1 byte length, indicating how many bytes are in the data body unit, takes a value from 0 to 255.

- Data body unit

  Variable length, for the actual application layer data/instructions.

- Check unit

  2 bytes length, the checksum value for all data in the header unit, data length unit and data body unit. The 16-bit CRC-16/X25 algorithm x16+x12+x5+1(0x1021) recommended by CCITT is used to generate the 2-byte CRC checksum (low byte sent and received first, high byte sent and received second). The sender must generate a 2-byte CRC checksum based on the data to be sent, the receiver receives the complete data frame and generates a new CRC checksum based on the received data, if the new CRC checksum is equal to the received checksum then the data frame is valid, otherwise a "check error" response is sent back to the sender.

3.3 Response flag

This protocol uses half-duplex communication. As the active initiator of a command, it needs to receive an response flag back from the passive receiver before it can proceed with subsequent operations. The first byte of the data body unit to which the receiver sends an response is the response flag. The response flag code is shown in the following table:

Table 3-1 Definition of response flag code

Code	Description	Note
0x00	Success	
0x80	Check error	Need to resend the command
0x90	Command not supported	
0x91	Parameter not supported	
0x92	No read permission	
0x93	No write permission	
0x94	No erase permission	
0x95	No verify permission	
0x96	No jump permission	
0x98	Failed to write data to Flash	
0x99	Failed to BlankCheck	

3.4 Example of sending and receiving data

Example of sending and receiving data:

Sender: 0x65 0x01 0x10 0x65 0xF3

Receiver: 0x65 0x09 0x00 0x18 0x00 0x08 0x00 0x01 0x01 0x06 0x00 0xBA 0x2B

Where yellow is the frame header unit, blue is the data length unit, green is the data body unit and grey is the CRC check unit. The first byte of the data body unit of the receiver is the response flag.

4 ISP mode command set

The following table lists the supported ISP commands:

Table 4-1 ISP command set

Command name	Command format	Command description
Query	0x10	Query the model number of the chip, the version of the bootloader, and the clock frequency of UCLK. Return: <Response flag + UCLK + BootLoaderId + ChipName>. UCLK is 2 bytes, low byte first, in MHz. BootLoaderId is 2 bytes, low byte first. ChipName is the ASCII code of the commercial model of the chip, of indeterminate length.
PPS	0x11 DIVN	Change the baud rate of the chip to UCLK / DIVN. Return: <Response flag>. DIVN is 2 bytes, the lower byte is sent first. Example: To set the baud rate to 115200bps when the clock frequency of UCLK is 6MHz, DIVN = 6000000/115200 = 52, DIVN is 0x34 0x00
Set BaseAddr	0x20 0x00 0x00 BasedAddr	Set the base address (32bit) of the read/write operation. Return: <Response flag>. BasedAddr is 4 bytes, the lower byte is sent first. 0x000xxxxx represents the code storage area. 0x2000xxxx represents the RAM area.
FLASH_Blank Check	0x22	Check if the FLASH area is all 0xFF. Return: <Response flag>. *Caution: This command only checks the FLASH area, not the OTP area and SDK area.*
FLASH_ChipErase	0x24 SdkZoneKey	Erases the entire FLASH area. For the chip without SDK area: SdkZoneKey is any value. For chips with SDK area: – Only erase the FLASH outer of the SDK area when SdkZoneKey is all FF. – Erase SDK area together when SdkZoneKey is correct. – Do not erase any area when the SdkZoneKey is incorrect. Return: <Response Flag>.
FLASH_SectorErase EE_SectorErase	0x26 Offset	Erase the page where the BaseAddr + Offset address is located. Return: <Response flag>. Offset is 2 bytes, the lower byte is sent first. *Caution: After erasing, the data on the page where the BaseAddr + Offset address is stored will be FF, and the data in the OTP area and SDK area will not be erased.*

Command name	Command format	Command description
WriteRamFun	0x27 Offset Data1-DataN	Write N bytes of data with 0x20000000 + Offset as the start address. Return: <Answer flag>. Offset is 2 bytes, low byte first, some chips must execute this instruction before WriteData instruction can be executed, value and data fields are related to chip model: The current supported CW32L010 has an Offset as 0x0800 and a data field as F0B50300124C012020602025980711D108E0676A2F40FCD11E68864214D1091D1B1D121F042A05D308681860F1E7491C5B1C521E002A09D008781870676A2F40FCD11E788642F2D00020F0BD0120F0BD0020024038B583185B1EEC24E400A04202D31634A34209D380246403A0420DD3FB25ED002D682C19A34207D2FFF7C0FF002801D0002032BD982032BD932032BD。
Write Data	0x28 Offset Data1-DataN	Write N bytes of data to BaseAddr + Offset and verify that the data read and written are the same (N = 1 - 248). Return: <Response flag>. Offset is 2 bytes and the lower byte is sent first. *Caution 1: Write data to Flash or RAM or EE or OTP according to the address range, data cannot be written to the SDK area.* *Caution 2: When BaseAddr+Offset and the number of bytes to be programmed are both Word aligned, call Word Program operation, otherwise call Byte Program operation. Word program takes only half the time of Byte Program.* *Caution 3: When writing data to the EE, the data must be aligned with the start address of the page and the length of the data must be an integer multiple of the page size. The corresponding area needs to be erased before data can be written to the EE.*
Read Data	0x29 Offset Cnt	From BaseAddr + Offset, read CNT bytes. Return: <Response flag + CNT bytes of data>. Offset is 2 bytes, the lower byte is sent first. Cnt is 1 byte and takes a value in the range 0-255. *Caution: Reading a specific address can obtain the commercial number, the capacity of Flash, the capacity of Ram, the FlashPageSize, the number of pins, but cannot read the data in the SDK area. See the digital signature section of the device user manual for details.*
Verify Data	0x2A Offset ByteCnt	Calculates the CRC value for each ByteCnt of data starting from BaseAddr + Offset. Return: <Response flag + CRC check value>. Offset, ByteCnt and CRC are all 2 bytes, with the lower byte being sent first. ByteCnt takes a value in the range 8 - 65535.

Command name	Command format	Command description
GetPreprocess Message	0x2C	Gets the FLASH preprocessing information, which is responded to by a preprocessing HEX written by the client. Returns: <Response flag + Addr + Data1-DataN> or just the response flag. Addr is 4 bytes, low byte first. Data1-DataN is a maximum of 64 bytes. The programmer needs to save Addr and Data1-DataN in memory. *Caution: This command is applicable to the programmer, the chip is not supported.*
WritePreprocess Message	0x2D	The programmer needs to write the Addr and Data1-DataN in memory to Flash. Return: <Response flag>. Addr and Data1-DataN are obtained by the GetPreprocessMessage command. *Caution: This command is applicable to the programmer, the chip is not supported.*
SetFlashReadOut Level	0x30 RdLevel	Set the read protection level of the chip. Return: <Response flag + RdLevel>. RdLevel of 0x00 for Level0; RdLevel of 0x01 for Level1; RdLevel of 0x02 for Level2; RdLevel of 0x03 for Level3; A RdLevel of 0x55 returns only the current protection level. Level0: SWD can read and write FLASH, ISP can read and write FLASH; Level1: SWD can downgrade, ISP can downgrade; Level2: SWD port is physically disconnected, ISP can downgrade; Level3: SWD port is physically disconnected, ISP port is physically disconnected and cannot be re-burned. *Caution: Some chips only support Level0 and Level2.*
Jump	0x40 0x00 0x00 Addr	Jump to the address specified by Addr to execute the program. Addr is 4 bytes, with the lower byte sent first. *Caution: Addr can only be 0x2000xxxx or 0x00000000.*

5 Revision history

Table 5-1 Document revision history

Date	Revision	Changes
June 18, 2024	Rev 1.0	Initial release.