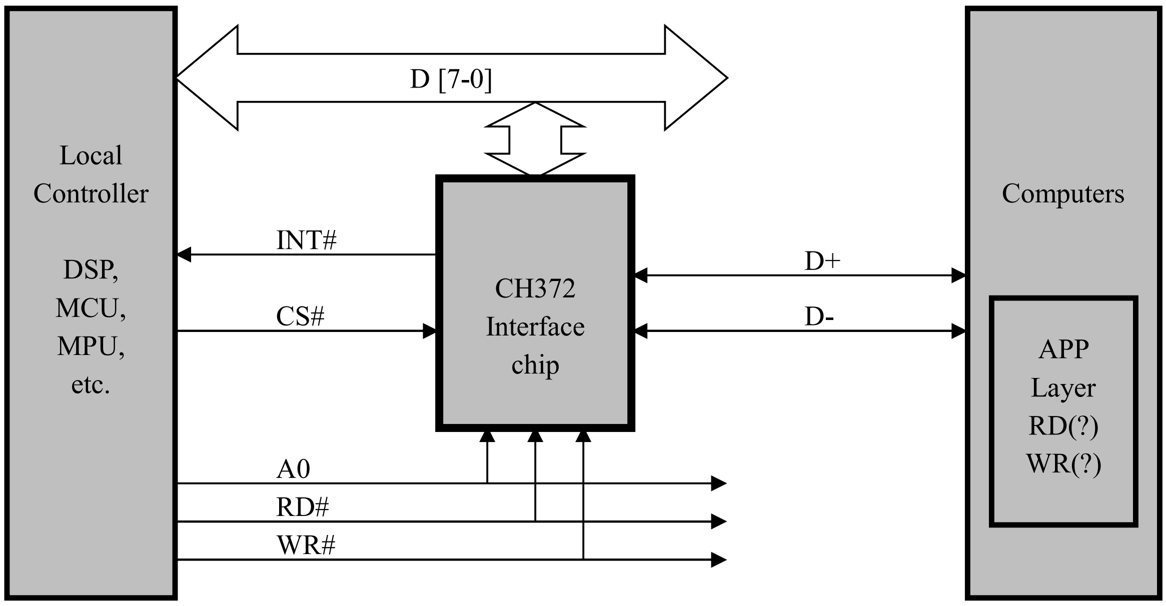
USB Bus Interface Chip CH372

Datasheet
Version: 4
http://wch.cn

1. Overview

CH372 is a universal device interface chip for USB bus. It is an upgraded product of CH371 and a simplified version of CH375. At the local side, CH372 has an 8-bit data bus, a read, write and chip selection control line and an interrupt output, and can be easily hooked to the system bus of DSP/MCU/MPU and other controllers. In the computer system, the necessary software of CH372 provides a simple and easy operation interface, and the communication with MCU at the local side is just like reading and writing files.

CH372 has a built-in underlying protocol in USB communication, and has a convenient built-in firmware mode and a flexible external firmware mode. In the built-in firmware mode, CH372 automatically processes all transactions of the default endpoint 0. The local MCU is only responsible for data exchange, so MCU program is very simple. In the external firmware mode, the external MCU processes various USB requests according to needs, so as to realize the devices conforming to various USB class specifications.

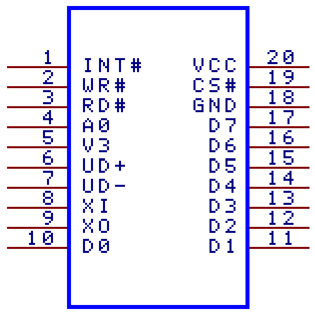


2. Features

- Full-speed USB device interface, compatible with USB V2.0, plug-and-play, only crystal and capacitors are required for peripheral components.
- Provide a pair of primary endpoints and a pair of auxiliary endpoints, and support control transmission, bulk transmission and interrupt transmission.
- Convenient built-in firmware mode and flexible external firmware mode.
- Shield the relevant USB protocol in the built-in firmware mode, and automatically complete the standard USB enumeration configuration process. The local side controller is not required to do any processing, simplifying the firmware programming of MCU.
- Universal Windows driver, providing device-level interfaces and providing API application-level interfaces through DLL.
- The product manufacturer can self-defined Vendor ID and Product ID.
- Universal local 8-bit data bus and a 4-wire control: read strobe, write strobe, chip selection input and interrupt output.

- The size of upload and download buffers for the primary endpoint is respectively 64 bytes, and the size of upload and download buffers for the auxiliary endpoint is respectively 8 bytes.
- Support the supply voltage of 5V or 3.3V, and support the low-power mode.
- CH372C has a built-in clock, and no external crystal is required.
- CH372 is a simplified version of CH375. On the basis of CH375, CH372 reduces the functions such as USB host mode and serial communication mode, so the hardware cost is lower. However, other functions are fully compatible with CH375, and WDM driver and DLL dynamic linking library of CH375 can be directly used.
- Small SSOP-20 lead-free package adopted, compatible with RoHS, pin compatible with CH374T chip.

3. Package



Package	Width of Plastic		Pitch of Pin		Instruction of Package	Ordering Information
SSOP-20	5.30mm	209mil	0.65mm	25mil	Tight 20-pin patch	CH372C
SSOP-20	5.30mm	209mil	0.65mm	25mil	Tight 20-pin patch	CH372B

Difference: CH372B must be externally connected with a crystal and an oscillating capacitor;
CH372C can be externally connected to a crystal and a capacitor, or use the built-in clock directly without external connection.

4. Pins

Pin No.	Pin Name	Pin Type	Description
20	VCC	Power	Positive power, an external 0.1uF power decoupling capacitor is required.
18	GND	Power	Ground, required to be connected to ground wire of USB bus
5	V3	Power	Connected to the VCC input external power at the supply voltage of 3.3V The external capacity is 0.01uF～0.1uF decoupling capacitor at 5V supply voltage
8	XI	Input	Input terminal of the crystal oscillator, requires an external crystal and oscillation capacitor. For the built-in clock mode of CH372C, XI shall be connected to GND
9	XO	Output	Inverted output terminal of the crystal oscillator, requires an external crystal and oscillation capacitor. For the built-in clock mode of CH372C, XO shall be suspended

6	UD+	USB signal	USB bus D+ data line
7	UD-	USB signal	USB bus D- data cable
17～10	D7～D0	Bi-directional Three-state	8-bit bidirectional data bus, built-in weak pull-up resistor,
3	RD#	Input	Read strobe input, active low, built-in pull-up resistor
2	WR#	Input	Write strobe input, active low, built-in pull-up resistor
19	CS#	Input	Chip selection control input, active low, built-in weak pull-up resistor
1	INT#	Output	Interrupt request signal output, active low
4	A0	Input	Address line input, distinctive command port and data port, built-in weak pull-up resistor, Write commands when A0=1; write and read data when A0=0

5. Commands

For the data in this datasheet, those with suffix B are binary number, and those with suffix H are hexadecimal number. Otherwise, it is a decimal number.

MCU referred to in this datasheet is basically applicable to DSP or MCU/MPU/SCM, etc.

This datasheet provides the commands that may be used in the built-in firmware mode. For other commands, please refer to Datasheet (II).

Code	Command name	Input Data	Output data	Command purpose
01H	GET_IC_VER		Version number	Get the chip and firmware versions
03H	ENTER_SLEEP			Go into the low-power sleep suspended state
05H	RESET_ALL		(Wait for 40mS)	Execute hardware reset
06H	CHECK_EXIST	Any data	Bitwise NOT	Test working status
0BH	CHK_SUSPEND	Data 10H		Set and check the USB bus Mode of suspended state
		Check mode		
12H	SET_USB_ID	VID low bytes		Set the USB vendor ID (VID) and product ID (PID)
		VID high bytes		
		PID low bytes		
		PID high bytes		
15H	SET_USB_MODE	Mode code	(Wait for 20uS) Operation status	Set USB working mode
22H	GET_STATUS		Interrupt status	Get the interrupt status and cancel the request

23H	UNLOCK_USB			Release the current USB buffer
27H	RD_USB_DATA0		Data Length	Read the data block from the endpoint buffer of the current USB interrupt
			Data stream	
28H	RD_USB_DATA		Data Length	Read the data block and release the current buffer from endpoint buffer of current USB interrupt
			Data stream	
2AH	WR_USB_DATA5	Data Length		Write the data block to upload buffer of USB endpoint 1
		Data stream		
2BH	WR_USB_DATA7	Data Length		Write the data block to upload buffer of USB endpoint 2
		Data stream		

If the output data of the command is in the operation status, refer to the following table.

Status code	Status name	Status description
51H	CMD_RET_SUCCESS	Operated Successfully
5FH	CMD_RET_ABORT	Operation failure

5.1. Command GET_IC_VER

This command is used to get the chip and firmware versions. One byte of data returned is the version number, the bit 7 is 1, the bit 6 is 0, and the bits 5-0 are the version number. If the returned value is 0B7H, remove 1 of bit 7, and the actual version number will be 37H.

5.2. Command ENTER_SLEEP

This command enables CH372 to be in the low-power sleep suspended state (this function is not supported on some models of chips). After entering the low power state, the clock of CH372 stops vibrating, thus saving power. The low power state is not quitted until one of the following two situations is detected: first, the signal is detected on the USB bus; second, MCU writes new commands to CH372 (commands without input data, such as GET_IC_VER).

Typically, it takes several milliseconds for CH372 to exit the low-power state and return to normal operation. When fully restored to normal operation, CH372 will generate the event interrupt USB_INT_WAKE_UP.

5.3. Command RESET_ALL

This command enables CH372 to perform a hardware reset. Typically, hardware reset is completed within 40mS.

5.4. Command CHECK_EXIST

This command is used to test the working state to check whether CH372 is working properly. This command needs to input 1 data, which can be any data. If CH372 is working properly, the output data of CH372 will be the bitwise reverse of the input data. For example, if the input data is 57H, the output data will be A8H. In addition, CH372 normally reads the data 00H from its parallel port before receiving no command after its reset.

5.5. Command CHK_SUSPEND

This command is used to set the mode to check the suspended state of the USB bus. This command requires to input two data, respectively the data 10H and the check mode. There are two check modes: 00H indicates that the USB suspended state is not checked (the default value after power-on or reset); 04H indicates that the USB suspended state is checked at interval of 50mS.

The suspended state of the USB bus includes two situations: First, the USB signal line is physically disconnected and there is no USB signal at all; second, the USB host stops sending SOF signals, that is, the USB host requires the USB device to enter the suspended state. CH372 will generate an event interrupt SB_INT_USB_SUSPEND when the USB bus suspended state is checked.

5.6. Command SET_USB_ID

This command is used to set the USB Vendor ID and product ID. This command requires to input 4 data, which are the low 8 bits of VID, the high 8 bits of VID, the low 8 bits of PID, and the high 8 bits of PID in sequence. If ID is required to be set, this command must be executed before the command SET_USB_MODE.

5.7. Command SET_USB_MODE

This command is used to set USB working mode. This command needs to input 1 data, which is a mode code:

Switch to the USB device mode not enabled (default mode after power-on or reset) when the mode code is 00H;
Switch to the enabled USB device mode and external firmware mode when the mode code is 01H;
Switch to the enabled USB device mode and built-in firmware mode when the mode code is 02H.

In USB device mode, "not enabled" means that the pull-up resistor of USB bus D+ is disabled, which is equivalent to disconnecting the USB device; "enabled" means that the pull-up resistor of the USB bus D+ is active, which is equivalent to connecting the USB device, so that the USB host can detect the existence of the USB device. USB device plugging event can be simulated by setting whether it is enabled or not.

Under normal circumstances, the USB working mode is set within 20uS, and the operation status is output after completion.

Please refer to Datasheet (II) for the external firmware mode.

5.8. Command GET_STATUS

This command is used to get the interrupt status of CH372 and notify CH372 to cancel the interrupt request. After CH372 requests an interrupt from MCU, MCU gets the interrupt status through the command, analyzes the interrupt cause and processes.

Interrupt status byte	Name	Analysis and description of interrupt status		
Bits 7-4	(Reserved bit)	Always 0000		
Bits 3-2	Current transaction	00 = OUT	10 = IN	11 = SETUP
Bits 1-0	Current endpoint	00 = Endpoint 0	01 = Endpoint 1	10 = Endpoint 2
		11 = USB bus reset		

The interrupt statuses are analyzed and described below. In the USB device mode of the built-in firmware mode, MCU only needs to process the interrupt status marked in gray in the table, and CH372 automatically processes other interrupt statuses.

Interrupt status value	Status name	Analysis and description of interrupt causes
03H，07H，0BH，0FH	USB_INT_BUS_RESET1～USB_INT_BUS_RESET4	USB bus reset detected (Bit1 and bit0 of the interrupt status value are 11)
0CH	USB_INT_EP0_SETUP	Receiver of the endpoint 0 receives the data.

		SETUP successfully
00H	USB_INT_EP0_OUT	Receiver of the endpoint 0 receives the data. OUT successfully
08H	USB_INT_EP0_IN	Transmitter of the endpoint 0 transmits the data. IN successfully
01H	USB_INT_EP1_OUT	Assistant endpoint/endpoint 1 receives the data. OUT successfully
09H	USB_INT_EP1_IN	Interrupt endpoint/endpoint 1 transmits the data. IN successfully
02H	USB_INT_EP2_OUT	Bulk endpoint/endpoint 2 receives the data. OUT successfully
0AH	USB_INT_EP2_IN	Bulk endpoint/endpoint 2 transmits the data. IN successfully
05H	USB_INT_USB_SUSPEND	USB bus suspend event (in case of CHK_SUSPEND)
06H	USB_INT_WAKE_UP	Wake-up from sleep (in case of ENTER_SLEEP)

5.9. Command UNLOCK_USB
This command is used to release the current USB buffer.CH372 firstly locks the current buffer to prevent buffer from being covered before requesting an interrupt from MCU and suspends all USB communication until MCU releases the current buffer through the command UNLOCK_USB or only after reading the data through the command RD_USB_DATA. The command cannot be executed more or executed less.

5.10. Command RD_USB_DATA0
This command is used to read the data block from the endpoint buffer of the current USB interrupt. The output data firstly read is data block length, namely, the number of bytes of subsequent data streams. The effective value of the data block length is 0 to 64. If the length is not 0, MCU must read the subsequent data one by one from CH372. The only difference between this command and the command RD_USB_DATA is that the latter will automatically release the current USB buffer after reading (equivalent to adding the command UNLOCK_USB).

5.11. Command RD_USB_DATA
This command is used to read the data block from the endpoint buffer of the current USB interrupt and release the current buffer. The output data firstly read is data block length, namely, the number of bytes of subsequent data streams. The effective value of the data block length is 0 to 64. If the length is not 0, MCU must read the subsequent data one by one from CH372. After reading the data, CH372 automatically releases the current USB buffer, so as to continue receiving data from the USB host.

5.12. Command WR_USB_DATA5
This command writes the data block to the upload buffer of the USB endpoint 1, which is the interrupt endpoint in the built-in firmware mode. The input data firstly written is data block length, namely, the number of bytes of subsequent data streams. The effective value of the data block length is 0 to 8. If the length is not 0, MCU must write the subsequent data one by one to CH372.

5.13. Command WR_USB_DATA7
This command writes the data block to the upload buffer of the USB endpoint 2, which is the bulk endpoint in the built-in firmware mode. The input data firstly written is data block length, namely, the number of bytes of subsequent data streams. The effective value of the data block length is 0 to 64. If the length is not 0, MCU must write the subsequent data one by one to CH372.

6. Functional Specification

6.1. General Description

CH372 kit includes CH372 and CH372 driver for the computer side. At the local side, CH372 automatically processes the basic transactions in the USB communication through the built-in firmware program; at the computer side, software such as driver and dynamic link library provide application layer interfaces to the computer application layer.

6.2. Hardware of Local Side

The passive parallel interface of CH372 includes: 8-bit bidirectional data buses D7-D0, read strobe input pin RD#, write strobe input pin WR#, chip selection input pin CS#, interrupt input pin INT# and address input pin A0. Through the passive parallel interface, CH372 can be easily hooked to the system bus of 8-bit DSP or MCU, and can coexist with a number of peripheral devices.

RD# and WR# of CH372 can be connected to the read strobe output pin and write strobe output pin of MCU respectively. CS# is driven by the address decoding circuit, and can be used for device selection when MCU has multiple peripheral devices. The interrupt request of INT# output is active at low level and can be connected to the interrupt input pin or ordinary I/O pin of MCU. MCU can get the interrupt request in interrupt mode or query mode.

When WR# is at high level and CS# and RD# and A0 are at low level, the data in CH372 are output through D7-D0. When RD# is at high level and CS#, WR# and A0 are at low level, data on D7-D0 is written to CH372. When RD# is at high level, CS# and WR# are at low level and A1 is at high level, the data on D7-D0 is written to CH372 as command codes.

CH372 occupies two address bits. When A0 pin is at high level, select command port write a new command, or read the interrupt flag; when A0 pin is at low level, select the data port to read and write the data.

The following table is the truth table of the parallel port I/O operation (X in the table means that this bit is not concerned, and Z means that three states of CH372 are disabled).

CS#	WR#	RD#	A0	D7-D0	Actual operation on CH372
1	X	X	X	X/Z	CH372 is not selected, and no any operation is made
0	1	1	X	X/Z	Although selected, no any operation is made
0	0	1/X	1	Input	Write a command code to the command port of CH372
0	0	1/X	0	Input	Write a command code to the data port of CH372
0	1	0	0	Output	Read data from the data port of CH372
0	1	0	1	Output	Read the interrupt flag from the command port of CH372B/C, and the bit 7 is equivalent to INT# pin

UD+ and UD- pins of CH372 shall be directly connected to the USB bus. If a fuse resistor or inductor or ESD protection device is connected in series for chip safety, the AC and DC equivalent series resistors shall be within 5Ω.

CH372 has a built-in power on reset circuit. Generally, no external reset is required.

When CH372B works normally, 12MHz clock signal shall be provided for it externally. Generally, the clock signal is generated by the built-in inverter of CH372B through the crystal stable frequency oscillator. The peripheral circuit is only required to be connected with a crystal with a nominal frequency of 12MHz between XI and XO pins, and connected with a high frequency oscillating capacitor to the ground for XI and XO pins respectively. If the 12MHz clock signal is inputted directly from the outside, it shall be inputted from the XI pin, and the XO pin is suspended.

CH372C supports two modes: external clock and built-in clock. For the external clock mode, refer to the aforesaid external 12MHz crystal and capacitor of CH372B; in the built-in clock mode, the XI pin shall be connected to GND and the XO pin shall be suspended to omit the external crystal and the oscillation capacitor.

CH372B and CH372C support supply voltage of 3.3V or 5V. When a 5V operating voltage is used, VCC pin of CH372 will input an external 5V power supply, and V3 pin shall be connected to an external power decoupling capacitor with a capacity of 0.01uF to 0.1uF. When 3.3V operating voltage is used, the V3 pin of the CH372 shall be connected to the VCC pin, and an external 3.3V power supply shall be inputted at the same time, and the operating voltage of other circuits connected to the CH372 shall not exceed 3.3V.

6.3. Internal Structure

CH372 integrates PLL frequency multiplier, USB interface SIE, data buffer, passive parallel interface, command interpreter, general firmware program and other major components.

PLL frequency multiplier is used to multiply 12MHz external input clock frequency to 48MHz as the USB interface SIE clock.

USB interface SIE is used for completion of physical USB data receiving and sending, automatic processing of bit tracking and synchronization, NRZI encoding and decoding, bit stuffing, conversion between parallel data and serial data, CRC data check, transaction handshake, error retry and USB bus status detection, etc.

The data buffer is used to buffer data sent and received by USB interface SIE.

The passive parallel interface is used to exchange data with the external DSP/MCU.

The command interpreter is used to analyze and execute various commands submitted by DSP/MCU.

Universal firmware programs are used to automatically process various standard transactions of USB default endpoint 0.

There are 5 physical endpoints inside CH372:

Endpoint 0 is the default endpoint and supports upload and download. The size of upload and download buffers is respectively 8 bytes.

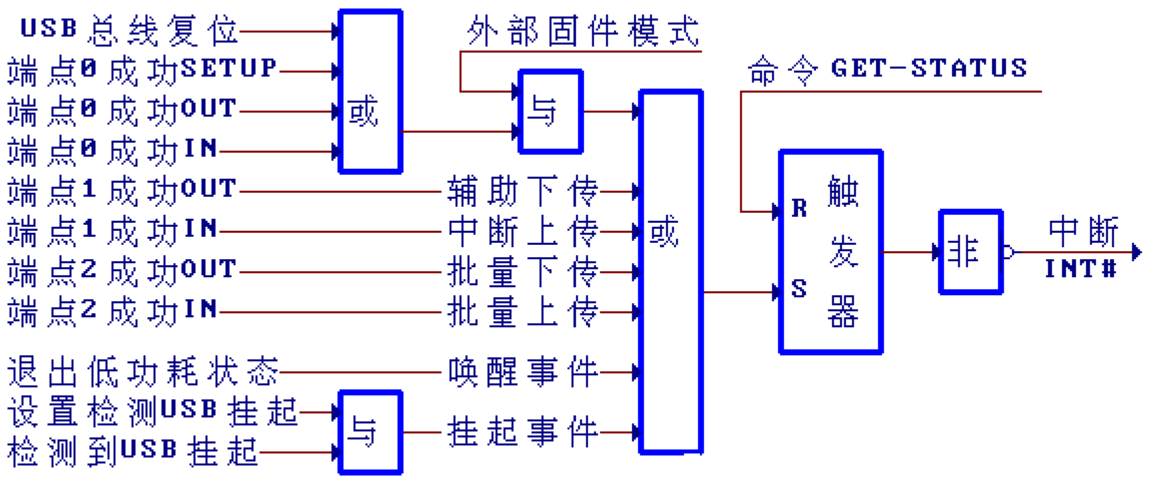
Endpoint 1 includes the upload endpoint and the download endpoint. The size of upload and download buffers is respectively 8 bytes. The number of the upload endpoint is 81H and the number of the download endpoint is 01H;

Endpoint 2 includes the upload endpoint and the download endpoint. The size of upload and download buffers is respectively 64 bytes. The number of the upload endpoint is 82H and the number of the download endpoint is 02H.

In the built-in firmware mode, the upload endpoint of the endpoint 2 serves as the bulk data sending endpoint, the download endpoint of the endpoint 2 serves as the bulk data receiving endpoint, the upload endpoint of the endpoint 1 serves as the interrupt endpoint, and the download endpoint of the endpoint 1 serves as the auxiliary endpoint.

In the external firmware mode, endpoint 0 is the default endpoint, endpoint 1 and endpoint 2 can be selected for use according to the needs of the USB product, and their use can be respectively defined by the descriptors provided by the external firmware. Typically, endpoint 2 is used as the primary endpoint for data transmission, and endpoint 1 can be used as the auxiliary endpoint if the USB product requires it.

The diagram below shows the interrupt logic inside CH372.



6.4. MCU Software of Local Side

CH372 occupies two address bits. When A0 pin is at high level, select command port write the command; when A0 pin is at low level, select the data port to read and write the data.

MCU reads and writes CH372 through an 8-bit parallel port. All operations are composed of a command code, several input data and several output data. Some commands do not need input data, and some commands do not have output data. The command operation steps are as follows:

① Write the command code to the command port when A0 is 1;
② If the command has input data, write the input data in sequence when A0 is 0, one byte at a time;
③ If the command has output data, read the output data in sequence when A0 is 0, one byte at a time;
④ The command is completed. Pause or go to ① to continue to execute the next command.

CH372 is specially used to process USB communication. After receiving or sending the data, CH372 informs MCU to process in the interrupt mode.

The processing steps for MCU to receive data through CH372 are as follows:

① When receiving the data from the USB host, CH372 first locks the current USB buffer to prevent it from being covered by the subsequent data, then sets the INT# pin to low level and requests an interrupt from MCU.
② When entering into the interrupt service program, MCU first executes the command GET_STATUS to get the interrupt status;
③ CH372 restores INT# pin to a high level and cancels the interrupt request after the command GET_STATUS is completed;
④ The interrupt status obtained by the above command GET_STATUS is "download successful", so MCU executes the command RD_USB_DATA to read the received data from CH372;
⑤ After the command RD_USB_DATA is completed, CH372 releases the current buffer to continue USB communication;
⑥ MCU exits the interrupt service program.

The processing steps for MCU to send data through CH372 are as follows:

① MCU executes the command WR_USB_DATA to write the data to be sent to CH372;
② CH372 passively waits for the USB host to pick up data when required;
③ When picking up the data from the USB host, CH372 first locks the current USB buffer to prevent data from being sent repeatedly, then sets the INT# pin to low level and requests an interrupt from MCU.
④ When entering into the interrupt service program, MCU first executes the command GET_STATUS to get the interrupt status;

⑤ CH372 restores INT# pin to a high level and cancels the interrupt request after the command GET_STATUS is completed;
⑥ The interrupt status obtained by the above command GET_STATUS is "upload successful", so MCU executes the command WR_USB_DATA to write another set of data to be sent to CH372. If there is no subsequent data to be sent, MCU will not necessarily execute the command WR_USB_DATA.
⑦ MCU executes the command UNLOCK_USB;
⑧ After the command UNLOCK_USB is completed, CH372 releases the current buffer to continue USB communication;
⑨ MCU exits the interrupt service program;
⑩ If MCU has written another set of data to be sent, go to ②; otherwise, end.

6.5. Software Interfaces of Computer Side

CH372 provides an application layer interface on the computer side. The application layer interface is a functional application oriented API provided by CH372 dynamic linking library DLL. All APIs have operation status returned after call, but may not have response data.

API provided by the CH372 dynamic linking library includes device management API, data transmission API and interrupt processing API.

For instructions on API parameters, please refer to CH372DLL.H or CH375DLL.H.

Device Management API:
Open device: CH375OpenDevice
Close device: CH375CloseDevice
Get the USB device descriptor: CH375GetDeviceDescr
Get the USB configuration descriptor: CH375GetConfigDescr
Reset USB device: CH375ResetDevice
Set USB data read and write timeout: CH375SetTimeout
Set exclusive use of current CH375 device: CH375SetExclusive
Set internal buffer upload mode: CH375SetBufUpload
Inquire the number of existing data packets in the internal upload buffer: CH375QueryBufUpload
Set the event notification program when USB device is plugged and unplugged: CH375SetDeviceNotify

Data Transmission API:
Read data block (data upload): CH375ReadData
Write data block (data download): CH375WriteData
Abort data block read operation: CH375AbortRead
Abort block write operation: CH375AbortWrite
Write auxiliary data (auxiliary data download): CH375WriteAuxData

Interrupt Processing API:
Read interrupt data: CH375ReadInter
Abort interrupt data read operation: CH375AbortInter
Set interrupt service routine: CH375SetIntRoutine

6.6. End-to-end USB Transmission (for Application Layer Design Reference)

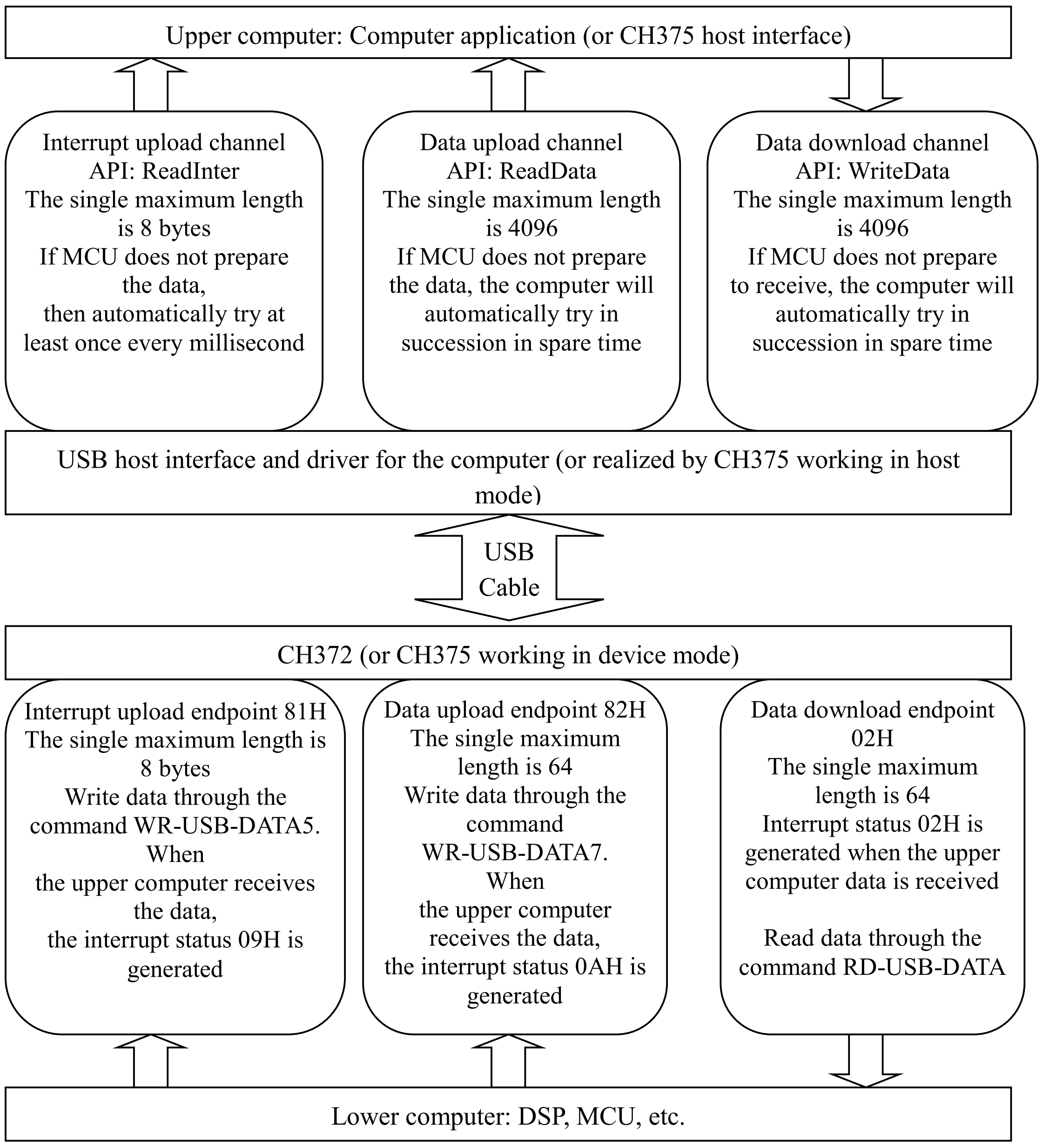
CH372 provides an end-to-end connection between the computer application layer and the local MCU. On this basis, designers of USB products can choose two communication modes: unidirectional data flow mode as well as request and response mode. The former uses unidirectional data flow with opposite directions for communication, which has relatively high data transmission rate, but it is not easy to synchronize the data.

The latter uses the inquiry mode of active request and passive response for communication, the data is automatically synchronized, interaction and controllability are good, the program design is simple, but the data transmission rate is relatively low.

6.6.1. Logical Structure

CH372 provides 4 independent end-to-end logical transmission channels, which are respectively called data upload channel, data download channel, interrupt upload channel and auxiliary data download channel. CH372 neither defines the purpose of each channel, nor defines its data format, so the USB product designer can define its purpose according to the need, and agree the data format in each transmission channel between the upper computer and the lower computer.

The logical structure diagram below applies only to the built-in firmware mode, so the default endpoint 0 is not included. In addition to the three channels in the figure, CH372 also provides an auxiliary data download channel, similar to the data download channel, but there are four differences: API of the upper computer is WriteAuxData; the auxiliary data download endpoint of the lower computer is 01H; the single maximum length is 8 bytes; the interrupt status is 01H.


Upper computer: Computer application (or CH375 host interface)
Interrupt upload channel API: ReadInter The single maximum length is 8 bytes If MCU does not prepare the data, then automatically try at least once every millisecond
Data upload channel API: ReadData The single maximum length is 4096 If MCU does not prepare the data, the computer will automatically try in succession in spare time
Data download channel API: WriteData The single maximum length is 4096 If MCU does not prepare to receive, the computer will automatically try in succession in spare time
USB host interface and driver for the computer (or realized by CH375 working in host mode)
USB Cable
CH372 (or CH375 working in device mode)
Interrupt upload endpoint 81H The single maximum length is 8 bytes Write data through the command WR-USB-DATA5. When the upper computer receives the data, the interrupt status 09H is generated
Data upload endpoint 82H The single maximum length is 64 Write data through the command WR-USB-DATA7. When the upper computer receives the data, the interrupt status 0AH is generated
Data download endpoint 02H The single maximum length is 64 Interrupt status 02H is generated when the upper computer data is received Read data through the command RD-USB-DATA
Lower computer: DSP, MCU, etc.

6.6.2. Unidirectional Data Flow Mode

The unidirectional data flow mode uses an upload data flow and a download data flow for unidirectional data communication, and the two data flows are completely independent of each other.

The download data flow is initiated by the computer application layer through the data download API. CH372 takes 64 bytes as a group, divides a large data block into multiple groups and submits them to MCU. If the application layer sends 150 bytes of data block, MCU will be interrupted for 3 times. 64 bytes will be gotten respectively in the first two times, and 22 bytes will be gotten in the last time.

There are two modes to initiate the upload data flow: one mode is inquiry mode, which means that the computer application layer regularly initiates the data flow in inquiry mode; the other mode is the spurious interrupt mode, which means that MCU notifies the computer application layer with the interrupt data and then the computer application layer initiates. Because the USB bus adopts a master-slave structure, the USB device can upload data to the computer only if the computer actively contacts the USB device.

In a system where the upload data flow is initiated in inquiry mode, the computer application layer always tries to read the data through the upload API. When MCU has no data to be uploaded, the computer application layer will wait (if the USB read timeout is set, it will exit). In fact, the thread of the application layer program will be suspended by the operating system. When MCU needs to upload data, the data shall be written into the upload buffer of CH372 bulk endpoint, and then the computer application layer automatically takes the data, and then CH372 notifies MCU of successful upload in an interrupt mode, so that MCU continues to upload subsequent data. In this mode, it is recommended that the internal buffer be set through CH375SetBufUpload for uploading.

In a system where the uploaded data flow is initiated in a spurious interrupt mode, a spurious interrupt service program is set when the computer application layer initializes, and then the application layer does not need to be involved in the uploaded data flow. When needing to upload data, first MCU writes the data into the upload buffer of the bulk endpoint, and then writes the interrupt characteristic data into the upload buffer of the interrupt endpoint. Within 1 millisecond (theoretical value), the spurious interrupt service program corresponding to the interrupt characteristic data is activated, and notifies the main program to call the data upload API to get the uploaded data block. During this period, MCU will receive two interrupts that CH372 notifies, first the endpoint successful upload interrupt and then the bulk endpoint successful upload interrupt.

6.6.3. Request and Response Mode

Request and response mode uses a download active request and an upload passive response for interactive bidirectional data communication. Download and upload correspond one to one and are correlated.

Active request refers to the data request downloaded from the computer application layer to MCU, and passive response refers to the response data uploaded to the computer application layer after MCU receives the data request. All communications are initiated by the computer application layer, and then ended when the response is received from MCU. The complete process includes:

① The computer application layer sends the data request to CH372 according to the pre-agreed format;
② CH372 notifies MCU in an interrupt mode;
③ MCU enters the interrupt service program, gets the interrupt status of CH372 and analyzes it;
④ If it is upload, release the current USB buffer, and then exit the interrupt program;
⑤ If it is download, read the data block from the data download buffer;
⑥ Analyze the received data blocks, prepare the response data, or exit the interrupt program before processing;
⑦ MCU writes the response data into the bulk endpoint of the upload buffer, and then exit the interrupt program;
⑧ CH372 returns the response data to the computer;

⑨ The computer application layer receives the response data.

7. Parameters

7.1. Absolute Maximum Value

Critical value or exceeding the absolute maximum value may cause the chip to work abnormally or even be damaged.

Name	Parameter description		Min.	Max.	Unit
TA	Operating Ambient temperature	CH372B or CH372C, VCC=5V, external clock	-40	85	℃
		CH372B or CH372C, VCC=3.3V, external clock	-40	85	
		CH372C, VCC=5V or 3.3V, internal clock	-20	70	
TS	Ambient temperature during storage		-55	100	℃
VCC	Supply voltage (VCC connects to power, GND to ground)		-0.5	6.5	V
VIO	Voltage on the input or output pins		-0.5	VCC+0.5	V

7.2. Electrical Parameters

Test Conditions: TA=25℃, VCC=5V, excluding the pins connected to the USB bus.
(If the supply voltage is 3.3V, all current parameters in the table need to be multiplied by a factor of 40%)

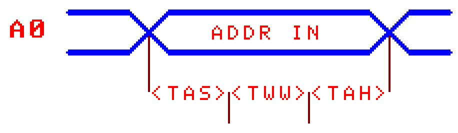
Name	Parameter description		Min.	Typ.	Max.	Unit
VCC	Power supply voltage	CH372B/C, V3 is not connected to VCC	4.2	5	5.4	V
		CH372B/C, V3 is connected to VCC	3.1	3.3	3.6	
ICC	Operating Total supply current	CH372B, VCC=5V		10	30	mA
		CH372C, VCC=5V		7	25	
		CH372B, VCC=3.3V		6	15	
		CH372C, VCC=3.3V		4	13	
ISLP	Supply current at the low power status I/O pin suspended/ internal pull-up	VCC=5V		0.15	0.25	mA
		VCC=3.3V		0.05	0.15	
VIL	Low level input voltage		-0.5		0.7	V
VIH	High level input voltage		2.0		VCC+0.5	V
VOL	Low level output voltage (4mA draw current)				0.5	V
VOH	High level output voltage (4mA output current)		VCC-0.5			V
IUP	Input current at the input terminal of built-in pull-up resistor		40	80	160	uA
VR	Voltage threshold of power-on reset		2.1	2.6	3.0	V

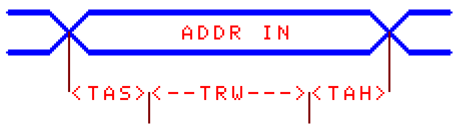
Note: During CH372 reset, INT# pin only provides the high level output current of 80uA.

7.3. Timing Parameters

Test Conditions: TA=25℃, VCC=5V, refer to the attached figure
(RD means that RD# signal is valid and CS# signal is valid; perform read operation when RD#=CS#=0)
(WR means that WR# signal is valid and CS# signal is valid, perform write operation when WR#=CS#=0)

Name	Parameter description	Min.	Typ.	Max.	Unit
FCLK	Input clock signal frequency of the XI pin	11.98	12.00	12.02	MHz
FCLKI	Internal frequency in CH372C built-in clock mode	11.80	12.00	12.20	MHz
TPR	Internal power-on reset time	10	30	40	mS
TE1	Execution time of command RESET_ALL	0.1	30	40	mS
TE2	Execution time of command SET_USB_MODE		10	20	uS
TE3	Execution time of command SET_ENDP?		2	4	uS
TE0	Execution time of other commands		1.5	2	uS
TSX	Interval time between CH372 command code and command code	1.5			uS
TSC	Interval time between CH372 command code and data	1.5		1000	uS
TSD	Interval time between CH372 data and data	0.6		1000	uS
TWW	Wtite pulse width	60		10000	nS
TRW	Read pulse width	60		10000	nS
TAS	Address setup time before Reada HIGH or Write HIGH	5			nS
TAH	Address hold time after Read HIGH or Write HIGH	5			nS
TIS	Data setup time before Write HIGH	0			nS
TIH	Data hold time after Write HIGH	5			nS
TON	Data valid after Read LOW	0		30	nS
TOF	Data hold after Read HIGH	0		20	nS
TINT	The time from receiving GET_STATUS to INT# pin cancel interrupt request		1.5	3	uS
TWAK	Wake-up time when exiting from low-power state	2	6	10	mS





8. Application

8.1. Connection to USB Bus/External Clock (Figure below)

P1 is a USB port. The USB bus includes a pair of 5V power lines and a pair of data signal lines. Generally, the +5V power line is red, the ground line is black, the D+ signal line is green, and the D- signal line is white. The supply current provided by the USB bus can generally reach 500mA. In general, the USB bus can be directly used to supply 5V power for the low-power-consumption USB products. If the standing power supply is provided for the USB product through other power supply methods, CH372 shall use the standing power supply together with MCU, and the power supply of the USB bus shall be disconnected. If the USB bus power supply is required to be used at the same time, the 5V power wire of the USB bus can be connected with the 5V standing power supply of the USB product through the resistor R1 with the resistance of about 1Ω, and the ground wires of the two shall be directly connected.

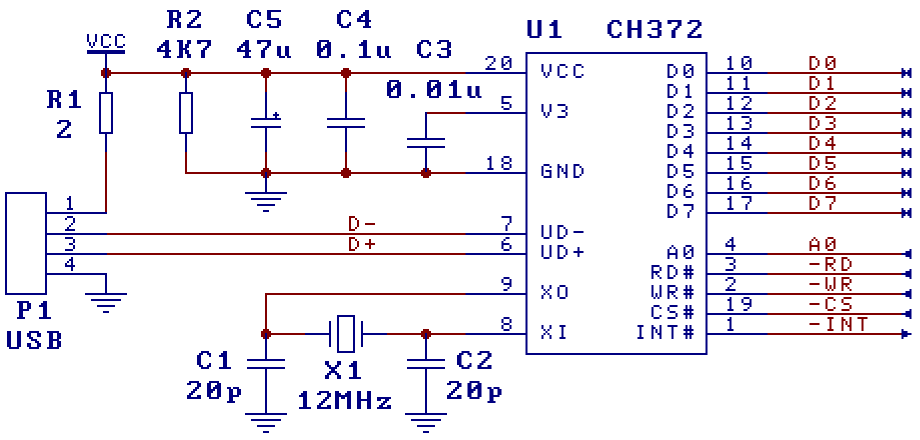
In the figure, the optional resistor R2 is used to release the electric energy in the electrolytic capacitor C5 in time after power-off, so that VCC can be reduced to 0V in time and CH372 can be reliably reset during power-on when the power supply is switched on next time. To ensure that CH372 is reset reliably, the rise time of the supply voltage from 0V to 5V shall be less than 100mS. So the capacity of the capacitor C5 and the resistance of the resistor R1 shall not be too large.

Capacitor C3 is used for decoupling the internal power node of CH372. C3 is a monolithic or high-frequency ceramic capacitor with a capacity of 0.01μF to 0.1μF. Capacitors C4 and C5 are used for decoupling the external power supply, and C4 is a monolithic or high-frequency ceramic capacitor with a capacity of 0.1μF. Crystal X1, capacitors C1 and C2 are used in the clock oscillation circuit of CH372. The frequency of crystal X1 is 12MHz, and C1 and C2 are monolithic or high-frequency ceramic capacitors with a capacity of 15-30pF.

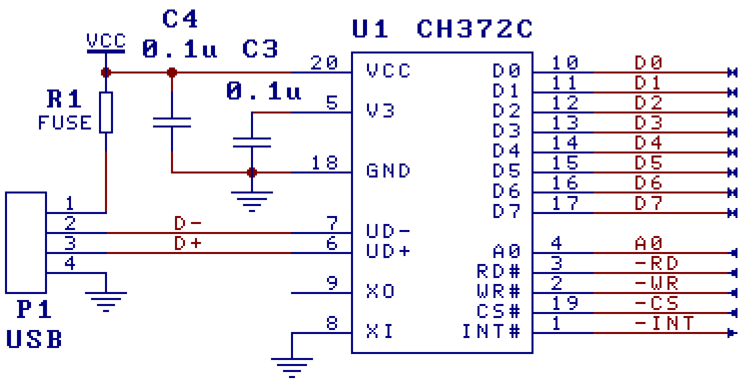
If CH372B is replaced with CH372C with a built-in clock for the finished board, remove X1 and C1, and replace C2 with a 0Ω resistor.

If the supply voltage of CH372 is 3.3V, V3 pin shall be short-circuited with VCC pin to jointly input 3.3V voltage, and the capacitor C3 can be eliminated.

It shall be noticed that the decoupling capacitors C3 and C4 shall be as close as possible to the connected pins of CH372 when the printed circuit board PCB is designed; the D+ and D- signal lines shall be close to parallel wiring, and ground wire or covered copper shall be provided on both sides to reduce the external signal interference; the length of the signal lines related to the XI and XO pins shall be shortened as far as possible to reduce the interference. The ground wire or covered copper shall surround the relevant components.

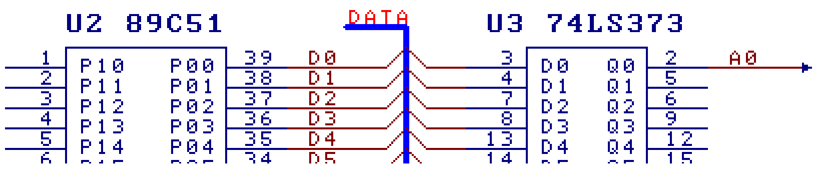


8.2. Connection to USB Bus/Internal Clock (Figure below)



If CH372C with built-in clock is used, the external crystal and the oscillating capacitor can be omitted, so that the circuit is more concise.

8.3. Connection to MCU Bus (Bus Extension)



CH372 has a universal passive parallel interface and can be directly connected to a variety of DSP, MCU, etc. In the typical application circuit of MCS-51 series MCU, CH372 can be directly hooked to the system bus of MCU U2 through D7-D0, -RD, -WR, -CS and A0 of the 8-bit passive parallel interface.

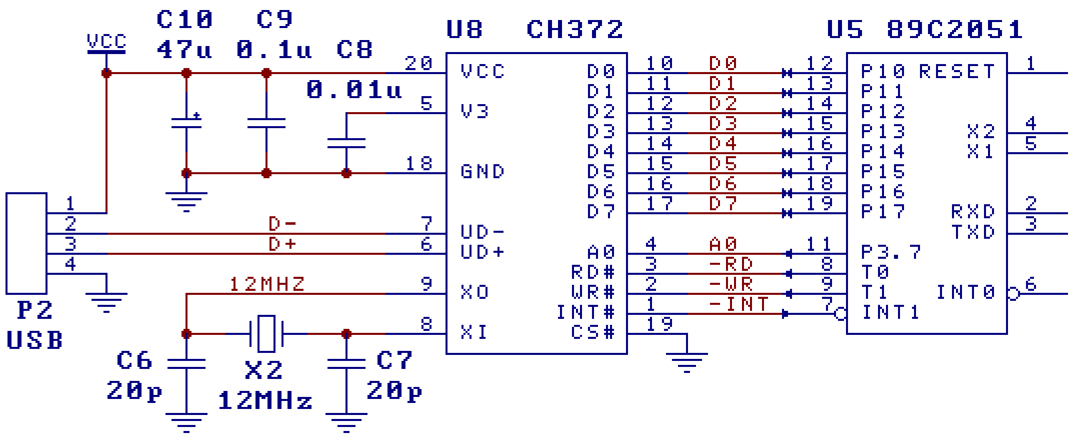
If MCS-51 MCU does not use U3 to latch the addresses A7-A0, the address line A0 of CH372 can be driven through P20 and other pins of U2. Of course, the port address in MCU program shall be modified accordingly. U4 is used for simple address decoding to generate the required chip selection signal. In the figure, the chip selection address range of CH372 is B000H-BFFFH. In fact, CH372 only needs to occupy two addresses: address BXX1H for writing commands and address BXX0H for reading and writing data.

8.4. Connection to MCU I/O (Not bus Extension)

In MCU system without external extended bus, MCU can also simulate the 8-bit parallel timing sequence operation of CH372 through the common I/O pin.

In the typical application circuit of the common MCS-51 series simplified MCU, CS# of CH372 is fixed at low level and always in the chip selection state. As 8-bit bidirectional data bus, P1 port of U5 can control each I/O pin to simulate the parallel timing sequence for data exchange with CH372 in MCU program.

The circuit in the figure is simplified, which can be used for software dog, USB encryption lock, USB to serial port or RS485, etc.



8.5. Interface Program of MCU

Please refer to CH372 evaluation board information for details.