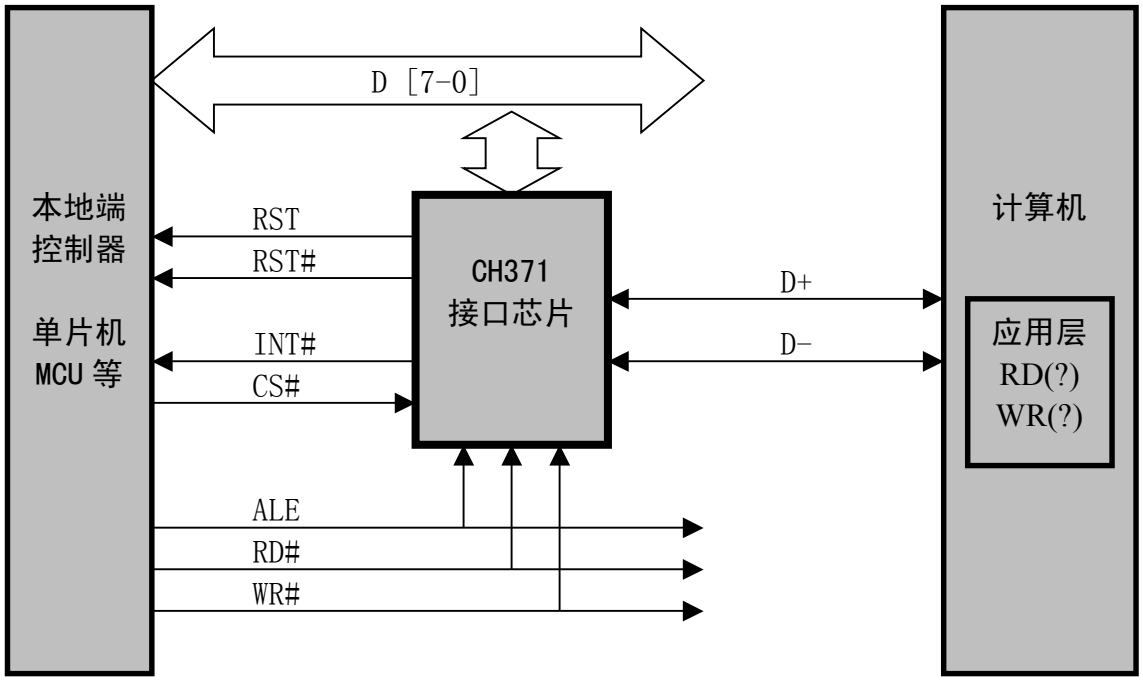
USB 总线接口芯片 CH371
中文手册

（第二版）

南京沁恒电子有限公司
TEL：　025-4599359
FAX：　025-4599759
www.winchiphead.com

1、概述

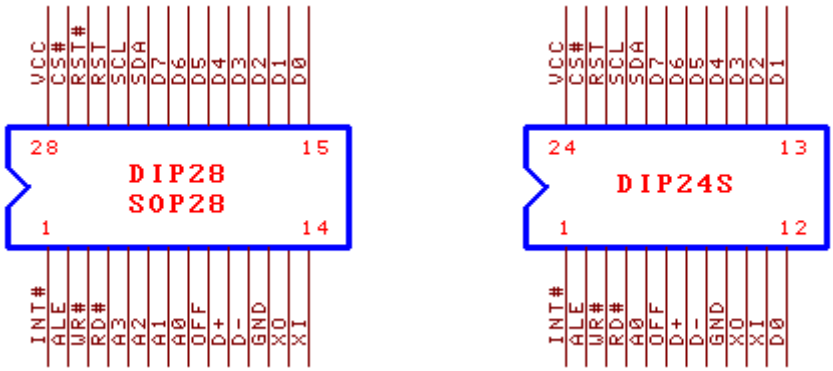
CH371 是一个 USB 总线的通用接口芯片。在本地端，CH371 具有 8 位数据总线和读、写、片选控制线以及中断输出，可以方便地挂接到单片机、DSP、MCU 等控制器的系统总线上；在计算机系统中，CH371 的配套软件提供了简洁易用的操作接口，与本地端的单片机通讯就如同读写硬盘中的文件。CH371 屏蔽了 USB 通讯中的所有协议，在计算机应用层与本地端控制器之间提供端对端的连接。基于 CH371，不需要了解任何 USB 协议或者固件程序甚至驱动程序，就可以轻松地将并口、串口的产品升级到 USB 接口。



2、特点

- 屏蔽 USB 协议，在计算机应用层与本地端之间提供端对端的连接。
- 两种通讯模式：单向数据流模式、请求加应答模式，支持伪中断。
- 自动完成 USB 配置过程，完全不需要本地端控制器作任何处理。
- 标准的 USB V1.1 接口，即插即用，D+引脚内置上拉电阻。
- 内置 4 个端点，支持 USB 的控制传输、批量传输、中断传输。
- 通用 Windows 驱动程序，提供设备级接口和应用层接口。
- 通用的本地 8 位数据总线，4 线控制：读选通、写选通、片选输入、中断输出。
- 占用 16 个地址，可选直接地址方式或者复用地址方式。
- 内置输入输出缓冲区，以中断方式通知本地端控制器传输数据。
- 内置硬件实现的 I^2C 主接口，应用层可以直接读写外挂的 I^2C 从设备。
- 在主控方式下可以提供 16 根输入信号线或者 12 根独立控制的输出信号线。
- 内置上电复位，提供高电平有效复位输出和低电平有效复位输出。
- 内置可选的看门狗电路 Watch-Dog，为本地端控制器提供监控。
- 可选多种封装：DIP28、SOP28、DIP24、CHIP。
- 底层协议说明以及整体方案请索取相关资料（需签署技术保密协议）。

3、封装



封装形式	宽度		引脚间距		说明
SOP28	7.62mm	300mil	1.27mm	50mil	标准的 28 脚贴片
DIP28	15.24mm	600mil	2.54mm	100mil	标准的 28 脚双列直插
DIP24S	7.62mm	300mil	2.54mm	100mil	标准的窄 24 脚双列直插

4、引脚

28 脚封装的引脚号	24 脚封装的引脚号	引脚名称	类型	引脚说明
28	24	VCC	电源	正电源
12	9	GND	电源	接地
14	11	XI	输入	晶体振荡的输入端，带偏置电阻
13	10	XO	输出	晶体振荡的反相输出端
10	7	D+	双向	USB D+数据线，内置上拉电阻可控
11	8	D-	双向	USB D-数据线
9	6	OFF	输入	关闭 D+上拉电阻，高有效，带下拉
22～15	19～12	D7～D0	双向	8 位双向数据总线，带上拉， 直接输入和独立控制输出
4	4	RD#	输入	读选通输入，低有效，带上拉， 同时用于看门狗的清除输入
3	3	WR#	输入	写选通输入，低有效，带上拉
27	23	CS#	输入	片选输入，低有效，带下拉
2	2	ALE	输入	地址锁存使能，高有效，带上拉， 下降沿锁存数据总线的复用地址
1	1	INT#	输出	中断输出，传输成功，低有效
5～8	5 部分支持	A3～A0	双向	4 位地址线输入，带上拉， 直接输入和独立控制输出
24	21	SCL	输出	I^2C 接口的时钟线
23	20	SDA	双向	I^2C 接口的数据线，开漏输出带上拉
25	22	RST	输出	上电复位和看门狗复位，高有效
26	不支持	RST#	输出	上电复位和看门狗复位，低有效

5、寄存器

5.1. 一般说明

本手册中的数据，以 B 结尾的为二进制数，以 H 结尾的为十六进制数，否则为十进制数，标注为 x 的位表示该位可以是任意值。

5.2. 只读寄存器

寄存器地址	寄存器名称	默认值	寄存器说明
02H	设备配置信息	00000000B	USB 设备的配置信息
06H	传输状态信息	xxxx000xB	数据传输的状态信息
07H	下传数据长度	0000xxxxB	低 4 位是正在下传的数据块长度
0FH～08H	数据下传缓冲区	xxH	数据块下传缓冲区，长度 8 字节
其它地址	保留	xxH	（禁止使用）

5.3. 只读寄存器的位说明

寄存器名称	位地址	位的使用说明
设备配置信息 (02H)	位 6～位 0	USB 设备地址，系统分配数值是 0 至 127
	位 7	USB 配置状态，数值为 1 则指示配置完成
传输状态信息 (06H)	位 0	数据传输方向，为 0 则下传，为 1 则上传
	位 1	传输成功标志，取反后作为中断输出 INT#
	位 4	数据块上传同步标志，每次上传成功后取反
	位 5	数据块下传同步标志，每次下传成功后取反
	位 6	中断数据上传同步标志，每次上传成功后取反
	位 7	控制传输同步标志，每次传输成功后取反
下传数据长度 (07H)	位 3～位 0	正在下传的数据块长度，有效值是 0 至 8

5.4. 只写寄存器

寄存器地址	寄存器名称	默认值	寄存器说明
02H	系统功能设定	00000000B	系统功能设定，使能或者禁止
06H	中断数据设定	00000111B	USB 中断端点的上传数据设定
07H	上传数据长度	00000000B	低 4 位是需要上传的数据块长度
0FH～08H	数据上传缓冲区	xxH	数据块上传缓冲区，长度 8 字节
其它地址	保留	xxH	（禁止使用）

5.5. 只写寄存器的位说明

寄存器名称	位地址	位的使用说明
系统功能设定 (02H)	位 7	看门狗使能，为 0 则禁用，为 1 则使能
中断数据设定 (06H)	位 2～位 0	位 2～0 为 111B，则数据与数据块上传相同； 位 2～0 为 000B，则数据未准备好，不上传； 位 2～0 为 001B～110B，是中断特征数据， 则上传中断数据设定寄存器，长度 1 字节
上传数据长度 (07H)	位 3～位 0	需要上传的数据块长度，有效值是 0 至 8， 数值为 15 或者 1111B 则指示数据未准备好

6、功能说明

6.1. 一般说明

与 ISA 总线、PCI 总线不一样，USB 总线虽然由硬件实现，但 USB 通讯协议以及数据传输主要由软件实现。尤其在计算机端，USB 控制器非常依赖于与其配套的驱动程序，多数控制传输的交互过程是在驱动程序的控制下实现的。当然，在本地端，CH371 芯片已经以硬件逻辑实现了控制传输的整个交互过程，简化了本地端控制器的工作量。本手册中所指的本地端控制器是指 USB 产品中与 CH371 芯片相连接的单片机、DSP、MCU 等。

CH371 套件包括 CH371 芯片和计算机端的 CH371 通用驱动程序。CH371 芯片以硬件逻辑而不是额外编写的固件程序实现了 USB 通讯协议，驱动程序则通过软件向计算机应用层提供了设备级接口以及应用层接口。CH371 套件屏蔽了 USB 通讯中的相关协议和驱动程序，在计算机应用层与本地端控制器之间提供端对端的连接。

除非设计人员主观需要，一般情况下，基于 CH371 套件设计 USB 产品不必考虑 USB 通讯协议、固件程序、驱动程序、自动配置过程、底层数据传输过程。设计人员所要做的工作与设计并口、串口的产品一样，包括两件事：一是从计算机的应用层发出数据传输请求并接收应答；二是当 USB 产品的控制器被通知有数据传输请求时，作出应答。

6.2. USB 设备配置和数据传输过程描述

6.2.1. 以下是 USB 即插即用的自动配置过程，由 CH371 芯片和驱动程序共同完成

① 带有 CH371 芯片的 USB 产品插入到计算机的 USB 插槽中；
② 检测到 USB 插入事件，操作系统有选择地复位 USB 产品；
③ 操作系统读取 USB 产品的设备描述符；
④ CH371 芯片返回设备描述符；
⑤ 操作系统根据设备描述符加载 CH371 驱动程序；
⑥ CH371 驱动程序读取 CH371 芯片的设备描述符和配置描述符；
⑦ CH371 芯片返回设备描述符和配置描述符；
⑧ CH371 驱动程序根据配置描述符请求操作系统对 CH371 芯片进行配置；
⑨ CH371 芯片被分配一个 USB 设备地址，并被指定一个 USB 配置；
⑩ CH371 芯片完成自动配置，CH371 驱动程序向应用层开放操作接口。

6.2.2. 以下是 USB 产品的应用层软件与 USB 产品的控制器之间的数据传输过程

① USB 产品的应用层软件发出数据传输请求；
② CH371 驱动程序将数据传输请求通过 USB 总线传递给 CH371 芯片；
③ CH371 芯片向 USB 产品的控制器（单片机、DSP、MCU）申请中断；
④ USB 产品的控制器进入中断程序，从 CH371 芯片获得数据传输请求；
⑤ USB 产品的控制器根据数据传输请求，将应答数据交给 CH371 芯片；
⑥ CH371 芯片将应答数据通过 USB 总线传递给 CH371 驱动程序；
⑦ CH371 驱动程序将应答数据返回给 USB 产品的应用层软件。

6.2.3. 以上均简化了 CH371 芯片与驱动程序之间的通讯过程，实际过程至少包括：

① CH371 驱动程序向 USB 控制器驱动程序发出请求（中间可能有集线器驱动）；
② USB 控制器驱动程序通过 USB 控制器在 USB 总线上发出请求；
③ CH371 芯片从 USB 总线上接收到请求（中间可能有集线器）；
④ 其它处理，例如，CH371 向 USB 产品的控制器转发请求；
⑤ CH371 芯片在 USB 总线上返回应答；

⑥ USB 控制器从 USB 总线上接收到应答（中间可能有集线器）；
⑦ USB 控制器将应答返回给 USB 控制器驱动程序；
⑧ USB 控制器驱动程序将应答返回给 CH371 驱动程序（中间可能有集线器驱动）。

6.3. 本地端的软硬件接口

6.3.1. 被动并行接口

CH371 芯片在本地端提供了通用的被动并行接口，包括：8 位双向数据总线 D7～D0、读选通输入 RD#、写选通输入 WR#、片选输入 CS#、中断输出 INT#以及地址锁存使能 ALE 或者 4 位地址线 A3～A0。通过被动并行接口，CH371 芯片可以很方便地挂接到多种单片机、DSP、MCU 的系统总线上，并与多个外围器件共存。

CH371 芯片的 RD#和 WR#为低电平有效，可以分别连接到单片机、DSP、MCU 等控制器的读选通输出引脚和写选通输出引脚。CS#为低电平有效，并且在 CH371 芯片中具有下拉电阻，当控制器没有连接其它外围器件时，可以悬空 CH371 芯片的 CS#作为默认片选；当控制器具有多个外围器件时，CS#可以由地址译码电路驱动。INT#为低电平有效，可以连接到控制器的中断输入引脚，每次数据传输成功后，INT#将输出低电平通知控制器，由于 INT#保持低电平的时间较短，所以建议控制器对中断输入采用下降沿触发方式。

CH371 芯片占用 16 个字节的空间，通过并行接口存取数据时需要事先指定 4 位的内部地址 IA3～IA0。上电复位后，CH371 芯片默认选择直接地址方式，即由 A3～A0 引脚输入地址直接作为内部地址；当 ALE 引脚检测到上升沿后，CH371 芯片自动切换为复用地址方式，即控制器将地址输出到数据总线的 D3～D0 上，由 CH371 芯片在 ALE 的下降沿将其锁存为内部地址。ALE 引脚在 CH371 芯片中具有上拉电阻，使用直接地址方式时，应该将 ALE 引脚悬空或者固定为高电平；使用复用地址方式时，ALE 为高电平有效，CH371 芯片在其下降沿锁存地址，应该连接到控制器的地址锁存使能引脚，例如 MCS-51 系列单片机的 ALE 引脚。D7～D0 作为双向三态数据总线，可以直接连接到控制器的数据总线上。

当 ALE 出现下降沿时，D3～D0 上的数据被 CH371 芯片作为地址锁存；当 CS#和 RD#都为低电平时，CH371 芯片中的数据被读出，D7～D0 输出指定单元的数据；当 CS#和 WR#都为低电平时，D7～D0 上的数据被写入 CH371 芯片中的指定单元。如果本地端控制器需要存取 CH371 芯片中的某个单元，应该通过直接地址方式或者复用地址方式事先指定该单元的地址，然后在 CS#、RD#、WR#的配合下读出数据或者写入数据。

6.3.2. I^2C 主接口

CH371 芯片在本地端提供了 I^2C 主接口，可以同时连接多个 I^2C 器件，只要各器件具有不同的设备地址。I^2C 接口包括两根信号线：串行时钟线 SCL、串行数据线 SDA。

SCL 总是由 CH371 驱动，空闲时为低电平。SDA 是开漏输出引脚，并且在 CH371 芯片中带有弱上拉电阻，可以由 CH371 芯片或者 I^2C 目标设备驱动，空闲时为高电平，用于双向数据传输。具体的数据传输过程可以参考相关协议。

CH371 芯片提供的 I^2C 接口可以方便地连接各种具有 I^2C 接口的外围芯片，例如，外接串行 EEPROM 器件 24CXX，存储一些希望断电后能够继续保持的数据。

6.3.3. 主控方式接口

CH371 芯片可以工作于一种主控方式，即使没有连接单片机、DSP、MCU 等控制器，CH371 芯片仍然可以输入数据和状态信号并且输出控制信号。在主控方式下，被动并行接口的所有引脚都可以得到充分的利用。

CH371 芯片的 D7～D0 和 A3～A0 引脚都是双向引脚，可以被分为 D7～D0、A3～A0 两组，分别设置用于输入或者输出。当设置双向引脚用于输出时，每根输出信号线都具有

对应的数据寄存器，从而可以独立控制各输出信号线的状态，当数据寄存器为 1 时输出高电平，当数据寄存器为 0 时输出低电平。如果将 D7～D0 和 A3～A0 引脚都设置用于输出，则 CH371 芯片可以有 12 根输出信号线，这些信号线可以产生多种组合的控制信号，而且都具有 10mA 的电流驱动能力，完全可以直接驱动 LED 指示灯等。

CH371 芯片的 RD#、WR#、ALE、CS#引脚都是输入引脚，如果将双向引脚 D7～D0 和 A3～A0 也设置用于输入，则 CH371 芯片可以有 16 根输入信号线，这些信号线的输入电平能够与 TTL/CMOS 兼容，可以直接输入数据和状态信息。

在主控方式下，CH371 芯片不需要任何单片机、MCU，就可以完成不是特别复杂的数据输入与输出，例如，直接控制模数转换芯片和通道选择器进行多通道数据采集。

6.3.4. USB 总线接口

CH371 芯片支持 USB 规范 V1.1，相关的引脚包括：D+引脚、D-引脚、OFF 引脚。

D+和 D-是用于 USB 通讯的双向引脚，应该在分别串接一个匹配电阻后连接到 USB 总线上。为了简化外围电路，在 CH371 芯片中还为 D+引脚内置了一个可以控制的上拉电阻 RD+，该上拉电阻主要用于向 USB 总线提供符合 USB 规范 V1.1 的上电检测信号。芯片上电复位期间，上拉电阻 RD+被关闭，当上电复位完成后，上拉电阻 RD+被启用，从而使 USB 总线能够检测到 CH371 芯片的存在，并完成 USB 设备的自动配置。

OFF 引脚用于在必要时，强行关闭 D+引脚的上拉电阻 RD+，模拟 CH371 芯片从 USB 总线上断开连接。OFF 引脚具有内部下拉电阻，不用时可以悬空或者将其接地，OFF 引脚的输入控制信号是高电平有效。当控制器将 OFF 引脚置为高电平时，CH371 芯片的 D+上拉电阻被关闭，从而模拟将 USB 产品从 USB 插槽中拔出的效果，而不必真正拔出 USB 产品；当 OFF 引脚恢复低电平后，又可以模拟 USB 产品刚刚接入 USB 总线的效果。

6.3.5. μP 监控接口

CH371 芯片提供的μP 监控包括上电复位和看门狗 Watch-Dog。单片机、DSP、MCU 等控制器的复位输入引脚可以根据需要直接连接到 CH371 芯片的 RST 引脚或 RST#引脚，当 CH371 芯片通电或者看门狗溢出时，RST 引脚输出高电平有效的复位脉冲信号，RST#引脚输出低电平有效的复位脉冲信号。CH371 芯片的上电复位脉冲信号同时作用于 CH371 芯片的内部电路，而看门狗复位脉冲信号不会对 CH371 芯片的内部电路起作用。

CH371 芯片的上电复位是指上电过程（从断电状态变为正常供电状态的过程）中产生的复位脉冲。CH371 芯片的上电复位支持短至数微秒的快速上电过程和长达数秒的慢速上电过程，以及正常供电状态下电压偶尔降低到复位门限以下的情况，但对于正常供电状态下电源电压的小幅度波动以及各种短于数微秒的尖锋干扰，CH371 芯片不会产生复位信号。为了减少电源干扰，在设计印制电路板 PCB 时，应该紧靠 CH371 芯片，在正负电源之间并联一个容量为 0.1uF 的独石或者瓷片电容进行电源退耦。

CH371 芯片在启用看门狗功能后，只要清除输入引脚 RD#的电平没有变化，看门狗计时器就会计时，当计满溢出周期时，就会产生看门狗复位脉冲信号。为了避免计时溢出而产生复位信号，控制器应该定期变化 RD#的电平，及时清除看门狗的计时。CH371 芯片的看门狗计时可以被下述的任何一个操作清除：上电复位、RD#从低电平变为高电平、RD#从高电平变为低电平、RD#从高电平变为低电平再立即恢复为高电平（读选通脉冲）等。启用看门狗功能后，当控制器的程序失控而使 RD#的电平长时间保持不变时，CH371 芯片就会输出看门狗复位脉冲信号，从而使控制器复位并重新进入正常工作状态。

6.3.6. 其它辅助电路

CH371 芯片正常工作时需要 12MHz 的时钟信号，一般情况下，时钟信号由 CH371 芯

片内置的反相器通过晶体稳频振荡产生。

CH371 芯片的 XI 引脚是反相器的输入端，XO 引脚是反相器的输出端，CH371 芯片内部已经在 XI 和 XO 之间连接了一个大阻值的电阻 RB，用于为反相器输入端提供必要的偏置电压。外围电路只需要在 XI 和 XO 引脚之间连接一个频率为 12MHz 的晶体，并分别为 XI 和 XO 引脚对地连接一个 20pF 的振荡电容，另外，在设计印制电路板 PCB 时，应该尽量缩短走线长度，并在这些元器件周围提供良好的覆铜和接地措施。

6.4. 计算机端的软件接口

CH371 在计算机端可以提供三个层次的软件接口：最底层是驱动程序级接口；中间层是设备级接口；最高层是应用层接口，后两层都可以由应用层直接访问。

6.4.1. 驱动程序级接口

驱动程序级接口是指 USB 产品制造商定制自己的驱动程序。编写驱动程序，需要了解驱动程序方面的一般知识，USB 总线的结构，USB 设备的结构，USB 的各种描述符，部分 USB 通讯协议，USB 设备的配置过程，USB 数据的传输过程，以及 CH371 芯片的各种特性等。对于具有相关设计能力的 USB 产品制造商而言，编写专用驱动程序去代替 CH371 通用驱动程序，能够增强技术保密性、提高程序运行效率、实现某些特殊功能，例如，使用 CH371 芯片和单片机构成一个符合智能卡规范的读卡器（规范对驱动程序有特定要求）。有关驱动程序级接口以及 CH371 芯片的特性在手册（二）中还有进一步的说明。

6.4.2. 设备级接口

设备级接口是由 CH371 通用驱动程序提供的面向设备操作的接口。CH371 驱动程序与 CH371 芯片配合后完成 USB 设备的自动配置，在设备级接口，CH371 提供的 USB 通讯已经被分解到多个管道，每个管道都具有自身的设备特性，所有的管道都有操作状态返回（指来自驱动程序的操作成功或者失败信息），但不一定有应答数据（指来自 CH371 芯片的数据）。USB 产品制造商不必了解 USB 设备的配置过程以及各种描述符，只需要选择特定的管道进行数据上传或者下传，例如，选择只读管道进行数据块上传，选择只写管道进行数据块下传等。如果将管道看成设备，那么操作设备级接口，仍然需要了解各个管道的特性，有些管道的功能是特定的，有些管道则是多功能的。

CH371 通用驱动程序提供的设备级接口主要包括 4 个管道：综合控制管道、中断数据上传管道、数据块上传管道、数据块下传管道。

6.4.2.1. 综合控制管道

综合控制是多功能的交互管道，对应于 CH371 提供的控制传输。综合控制管道的功能包括：执行 USB 标准设备请求、发出 I^2C 接口的操作命令、输出控制信号、输入状态信号。应用层软件向综合控制管道所提交的每个操作请求，都至少包括 8 个字节数据，以及可选的附加数据；综合控制管道根据请求的不同有可能返回应答数据。

6.4.2.1.1. 执行 USB 标准设备请求

USB 标准设备请求至少包括 8 个字节的请求和请求参数，以及可选的附加数据，具体格式和数据以及应答可以参考 USB 规范。CH371 支持下列标准设备请求：清除特性、设置特性、获取状态、获取描述符等。例如，获取设备描述符的请求数据是“80H、06H、00H、01H、00H、00H、12H、00H”，将其写入综合控制管道，则正常情况下的应答数据是 CH371 芯片的设备描述符，长度是 18 字节。

6.4.2.1.2. 发出 I²C 接口的操作命令

I²C 接口操作用于直接存取 CH371 芯片所挂接的 I²C 器件。CH371 芯片可以同时连接多个 I²C 器件，各器件都具有不同的设备地址，从而便于应用层区分和选择目标设备。发出 I²C 操作命令的请求数据是 8 个字节，不需要附加数据，只有操作状态返回。

字节顺序	请求数据	请求：发出 I²C 接口的操作命令
字节 0	40H	请求类型
字节 1	53H	请求代码（专用于发出 I²C 接口的操作命令）
字节 2	xxH	对于写操作是待写入数据，对于读操作则为任意值
字节 3	00H	（保留）
字节 4	xxH	指定存取地址，即 I²C 目标设备内的单元地址
字节 5	xxH	位 7～1 是目标设备地址，位 0 为 0 则写，位 0 为 1 则读
字节 6	00H	附加发送的数据长度，因为没有附加数据所以为 0， 对于 I²C 读操作，读出的数据请参考第 6.4.2.1.4.节
字节 7	00H	

例如，CH371 芯片连接了多个 I²C 接口器件，其中有 24C02 芯片，则请求数据“40H、53H、A5H、00H、36H、A0H、00H、00H”表示向 24C02 芯片中的地址为 36H 的单元写入数据 A5H，其中，第 6 字节 A0H 的位 7～位 1 指定目标设备地址为 1010000B，对应于页地址为 000B 的 24C02 芯片，位 0 为 0，指定发出 I²C 接口的写操作命令。

6.4.2.1.3. 输出控制信号

输出控制信号主要用于在主控方式下由 CH371 芯片直接产生控制信号。CH371 芯片的 D7～D0 和 A3～A0 引脚可以被设置用于输出，例如，驱动 LED 指示灯、控制继电器等。输出控制信号的请求数据是 8 个字节，不需要附加数据，只有操作状态返回。

字节顺序	请求数据	请求：输出控制信号
字节 0	40H	请求类型
字节 1	51H	请求代码（专用于输出控制信号）
字节 2	xxH	控制 D7～D0 引脚，0 对应低电平，1 对应高电平
字节 3	0xH	位 3～0 控制 A3～A0 引脚，0 对应低电平，1 对应高电平
字节 4	0xH	位 0 控制 D7～D0 三态输出，为 0 则输入，为 1 则输出， 位 1 控制 A3～A0 三态输出，为 0 则输入，为 1 则输出， 位 2 反相后控制 INT#引脚，0 则高电平，1 则低电平
字节 5	00H	（保留）
字节 6	00H	附加发送的数据长度，因为没有附加数据所以为 0
字节 7	00H	

例如，基于 CH371 芯片设计一个 8 通道数据采集器，用 CH371 芯片的 A2～A0 引脚控制通道选择器，用 D7～D0 引脚以及 CS#、RD#、WR#、ALE 引脚输入模数转换芯片的并行数据和转换状态，用 A3 引脚启动模数转换，不需要单片机、MCU 等控制器，则请求数据“40H、51H、00H、0BH、02H、00H、00H、00H”表示将 CH371 芯片的 D7～D0 引脚设置为输入，INT#引脚保持高电平，A3～A0 引脚设置为输出，其中 A2～A0 引脚输出 011B，用于选择采集通道，A3 引脚输出高电平，用于启动模数转换。

6.4.2.1.4. 输入状态信号

输入状态信号可以用于获取 I²C 接口读操作的结果，或者利用 CH371 芯片的多余引脚获取 USB 产品的一些状态信息，或者在主控方式下直接输入数据和状态信息。CH371 芯片的 RD#、WR#、ALE、CS#引脚以及 D7～D0 和 A3～A0 引脚都可以用于输入数据和状态

信息。输入状态信号的请求数据是 8 个字节，不需要附加数据，但具有应答数据。

字节顺序	请求数据	请求：输入状态信号
字节 0	C0H	请求类型
字节 1	52H	请求代码（专用于输入状态信号）
字节 2	00H	（保留）
字节 3	00H	（保留）
字节 4	00H	（保留）
字节 5	00H	（保留）
字节 6	08H	希望返回的应答数据的长度，根据需要可以设定为 0 至 8 个字节，该数值决定了应答数据的长度
字节 7	00H	

正常情况下，输入状态信号的请求具有应答数据，长度不超过 8 个字节。

字节顺序	应答数据	输入状态信号请求的应答数据
字节 0	xxH	D7～D0 引脚的输入状态
字节 1	xxH	RD#、WR#、ALE、CS#、A3～A0 引脚的输入状态
字节 2	xxH	D7～D0 引脚对应的数据寄存器
字节 3	0xH	A3～A0 引脚对应的数据寄存器，只有位 3～位 0 有效
字节 4	xxH	如果字节 5 位 0 为 0 即 I^2C 接口空闲，则该单元是最近从 I^2C 目标设备读出的数据。在发出 I^2C 接口读操作命令后，应该请求输入状态信号从应答数据字节 4 获得读操作结果
字节 5	000000xxB	位 1 是 SDA 引脚的输入状态，位 0 是 I^2C 接口的忙标志
字节 6 或 7	00H	（保留）

例如，CH371 芯片连接了 24C02 芯片，如果提交请求“40H、53H、00H、00H、49H、A1H、00H、00H”发出 I^2C 操作命令，则表示从 24C02 芯片中的地址为 49H 的单元读取数据，然后提交请求“C0H、52H、00H、00H、00H、00H、08H、00H”输入状态信号，则应答数据的第 5 个字节就是从 24C02 芯片的 49H 地址读取的数据。如果是第 6.4.2.1.3.节的数据采集器，输入状态信号所返回的应答数据中将包含 D7～D0、CS#、RD#、WR#、ALE 等引脚的输入状态，也就是模数转换的结果。

6.4.2.2. 中断数据上传管道

中断数据上传是只读管道，对应于 CH371 提供的中断传输的上传，用于在可以预见的等待时间内查询 USB 产品控制器提交给 CH371 芯片的中断数据，该管道具有相对的实时性，其响应时间不超过 2 毫秒。

中断数据上传是单向上传管道，所以应用层不需要提交任何数据就可以直接读取该管道，根据中断数据设定寄存器的设定值，返回的应答数据会有所不同。

中断数据设定寄存器	从中断数据上传管道返回的应答数据
00000111B 或者 07H	与数据块上传管道一样，数据来自数据块上传缓冲区
00000000B 或者 00H	中断数据未准备好，不上传数据，稍后计算机将自动重试
00000001B～00000110B	数据来自中断数据设定寄存器，长度为 1 字节，数值是 01H～06H，作为中断特征数据选择伪中断服务程序

6.4.2.3. 数据块上传管道

数据块上传是只读管道，对应于 CH371 提供的批量传输的上传，用于将本地端控制器提交的数据块上传给计算机的应用层。由于是单向上传管道，所以应用层不需要提交任何数据就可以直接读取该管道，返回的应答数据来自 CH371 芯片的数据块上传缓冲区，

应用层可以在 0 至 250 个字节之间指定需要上传的数据块的长度，但应答数据的实际长度同时受到上传数据长度寄存器的影响。如果上传数据长度寄存器的值为 15，则说明数据未准备好，CH371 芯片将不会上传任何数据，计算机将会在稍后的时间内重试。

6.4.2.4. 数据块下传管道

数据块下传是只写管道，对应于 CH371 提供的批量传输的下传，用于将计算机应用层提交的数据块下传给本地端的控制器。由于是单向下传管道，所以应用层必须提交数据才能写入该管道，并且只有操作状态返回而没有应答数据，应用层必须将所提交的数据块的长度指定在下传数据长度寄存器中，长度可以是 0 至 250 个字节，如果长度为 0 则 CH371 芯片仍然会通知本地端控制器，只是数据块下传缓冲区中没有数据。

6.4.3. 应用层接口

应用层接口是由 CH371 动态链接库 DLL 提供的面向功能应用的 API。在应用层接口，动态链接库将 CH371 驱动程序提供的管道进一步分解为多个功能应用 API，每个 API 都实现一个具体的功能，并用简便易用的 API 参数代替设备级接口中所要求的数据格式，所有 API 在调用后都有操作状态返回，但不一定有应答数据。USB 产品制造商不必了解 USB 的知识，也不必了解 CH371 管道的数据格式，只需要选择特定 API 实现所需功能，就像通过并口、串口传输数据或者读写计算机硬盘中的文件一样。例如，调用数据传输 API 就可以与本地端控制器互传数据，调用 I^2C 操作 API 就可以读写 I^2C 接口的目标设备。

CH371 动态链接库提供的 API 主要包括：设备管理 API、数据传输 API、中断查询 API、I^2C 操作 API、直接控制 API。

6.4.3.1. 设备管理 API

设备管理 API 一般不会用到，该 API 包括多个具体子功能：获取 USB 设备描述符、获取 USB 配置描述符、复位 USB 设备等。以获取 USB 设备描述符为例，只要提供一个数据缓冲区，调用设备管理 API 后，就可以在缓冲区中获得 USB 设备描述符，完全不需要考虑 USB 结构、USB 协议、USB 配置、数据格式、以及 CH371 芯片的各种特性等。

6.4.3.2. 数据传输 API

数据传输 API 用于计算机应用层与 USB 产品的控制器互传数据，所以使用较频繁，该 API 包括三个具体子功能：数据上传、数据下传、数据先下传再上传，其中的数据上传还可以选用高优先级（与 CH371 芯片的设定有关）。只要提供一个数据缓冲区，调用数据传输 API 就可以上传或者下传数据。例如，指定一个数据缓冲区，里面存放准备下传的数据，调用数据传输 API 后，就可以将缓冲区中的数据下传给 USB 产品的控制器。

6.4.3.3. 中断查询 API

中断查询 API 包括多个具体子功能：查询中断数据、设置伪中断等。查询中断数据是指及时地查询 USB 产品控制器是否提交了中断数据，调用 API 时需要提供一个数据缓冲区；设置伪中断用于在 USB 产品控制器提交中断数据后及时通知指定的伪中断服务程序，因为 CH371 芯片支持 6 个中断特征数据，每个中断特征数据都可以对应一个伪中断服务程序，所以调用 API 时需要提供一个伪中断服务程序和对应的中断特征数据。

6.4.3.4. I^2C 操作 API

I^2C 操作 API 用于读写 I^2C 接口的目标设备，包括两个子功能：读目标设备、写目标设备。指定目标设备地址、单元地址，然后调用该 API 就可以从目标设备读取数据，指

定目标设备地址、单元地址和待写数据，然后调用该 API 就可以向目标设备写入数据，完全不需要考虑设备级接口中所要求的数据格式。

6.4.3.5. 直接控制 API

直接控制 API 用于 CH371 芯片直接输入数据和状态信息，以及直接输出控制信号，包括三个子功能：设置模式、输入状态、输出控制。通过该 API 实现直接输入输出，完全不需要考虑设备级接口中所要求的数据格式。

6.5. 端对端的数据传输

CH371 在计算机应用层与本地端控制器之间提供了端对端的连接，在这个基础上，USB 产品的设计人员可以选用两种通讯模式：单向数据流模式、请求加应答模式。前者使用两个方向相反的单向数据流进行通讯，具有相对较高的数据传输速率，但是数据不容易同步，适用于数据量较大、交互操作少的应用；后者使用主动请求和被动应答的查询方式进行通讯，数据能够自动同步，具有较好的交互性和可控性，但是数据传输速率相对较低，适用于交互操作多、数据量较小的应用。

CH371 芯片的传输状态信息寄存器中提供了多个标志位：数据传输方向标志位、传输成功标志位、以及多个传输同步标志位。数据传输方向标志与传输成功标志配合后，可以确定是上传成功还是下传成功，上传成功是指本地端控制器提交的数据已经被计算机成功接收到，下传成功是指 CH371 芯片成功接收到计算机发送的数据，但有可能还没有被本地端控制器读出。传输成功标志反相后直接作为中断输出，CH371 芯片在传输成功后通知本地端的控制器，为了配合各种速度的控制器，同时为了防止中断输出引脚长时间保持在低电平，传输成功状态被设定为一个时间比较短暂的脉冲信号，也就是说，一次传输成功后，传输成功标志变为 1 并保持一段时间，然后自动恢复为默认值 0。对于速度较高的控制器，完成可以根据传输成功标志确定传输是否成功，但对于速度较低的控制器，有可能检测不到或者漏掉传输成功状态，所以 CH371 芯片提供了传输同步标志以确定传输是否成功。上传、下传、中断、控制管道都有各自的传输同步标志，上电复位后的默认值是 0，每当成功地上传或者下传一组数据后，相应的同步标志位就会取反，从 0 变为 1，或者从 1 变为 0，本地端控制器可以查看该标志位以确定传输是否成功。

6.5.1. 单向数据流模式

单向数据流模式使用一个上传数据流和一个下传数据流进行双向数据通讯，两个数据流之间完全独立。

下传数据流是由计算机应用层发起的，CH371 芯片在收到下传数据后以中断方式通知本地端控制器，控制器首先查看传输状态信息寄存器中的数据传输方向标志位，如果是下传，则根据下传数据长度寄存器中的长度，从数据下传缓冲区中读取数据块。虽然数据流具有连续、无首尾、无长度的特性，但在实际通讯过程中，仍然需要将其人为地断开成多个数据块。在计算机应用层，数据块的大小只受缓冲区大小的限制；在本地端，受控制器处理速度的限制，CH371 芯片以 8 个字节为一组，将一个较大的数据块分成多组不超过 8 个字节的小数据块提交给本地端控制器；如果计算机应用层发送的数据块是 20 个字节，则本地端控制器将会被中断 3 次，前两次各获取 8 个字节，最后一次获取 4 个字节；如果本地端控制器预先知道数据块的总长度，也可以在第一次被中断并读取 8 个字节后，不退出中断程序，而是查询传输状态信息寄存器中的传输同步标志位（或者传输成功标志位），当其变化后就可以继续读取 8 个字节，然后再次查看缓冲区传输同步标志位是否变化以便再次读取后续数据，直到取得所有数据后再退出中断程序。

上传数据流的发起方式有两种：一种是查询方式，指计算机应用层定期以查询方式

发起；另一种是伪中断方式，指本地端控制器以中断数据通知计算机应用层，再由计算机应用层发起。因为 USB 总线是主从式结构，只有在主控制器将令牌分配给 USB 设备的期间，USB 设备才能向总线上发送数据，所以即使是上传数据流，仍然需要由计算机应用层发起，但数据流的方向是从本地端到计算机端。

上传数据流以查询方式发起的系统中，计算机应用层可以通过一个定时事件发起查询，定时的间隔时间可以根据需要而定，但不宜少于 20 毫秒，否则可能会占用 CPU 时间和 USB 带宽。如果设定 50 毫秒定时，则本地端有数据需要上传时，计算机应用层可以在 50 毫秒时间内响应；设定 1 秒定时则响应时间也为 1 秒。在本地端，CH371 芯片上电复位后，由于此时没有数据需要上传，所以控制器应该设置上传数据长度寄存器，使长度为 0（上电复位后的默认值）或者 15。设置长度为 0 则说明需要上传的数据块长度为 0，也就是说，计算机应用层将会得到长度为 0 的上传数据块；设置长度为 15 则说明数据尚未准备好，暂时不上传，所以计算机将会每隔一段时间再重试，在重试期间，应用层不会得到任何上传数据，重试也不会对本地端控制器产生影响。当本地端控制器需要上传数据时，只需将数据写入数据上传缓冲区中，然后将数据长度写入上传数据长度寄存器，计算机应用层将在稍后的定时查询或者重试过程中获得数据。当数据上传成功后，CH371 芯片将会以中断方式通知本地端控制器，控制器首先查看传输状态信息寄存器中的数据传输方向标志位，是上传则说明数据上传成功，缓冲区空闲，如果还有后续数据则可以继续发送，或者在没有后续数据时，设置上传数据长度寄存器中的长度为 0 或者 15。

上传数据流以伪中断方式发起的系统中，计算机应用层初始化时将一个伪中断服务程序和对应的中断特征数据提交给中断查询 API，然后应用层就不需要再涉及到上传数据流，伪中断服务程序只需要被动地等待伪中断，这里的伪中断区别于纯粹的硬件中断。在本地端，CH371 芯片上电复位后，控制器应该设置中断数据设定寄存器，使数据为 0，说明此时上传数据尚未准备好，暂时不上传。当本地端控制器需要上传数据时，首先将数据写入数据上传缓冲区中，接着将数据长度写入上传数据长度寄存器，然后将中断特征数据写入中断数据设定寄存器。在 2 毫秒之内，与中断特征数据对应的伪中断服务程序被激活，伪中断服务程序指定一个数据缓冲区并调用数据传输 API 获得上传数据块。当数据上传成功后，CH371 芯片将会以中断方式通知本地端控制器，控制器首先查看传输状态信息寄存器中的数据传输方向标志位，如果是上传，则继续发送后续数据，或者在没有后续数据时，设置中断数据设定寄存器中的数据为 0。

6.5.2. 请求加应答模式

请求加应答模式使用一个下传的主动请求和一个上传的被动应答进行交互式的双向数据通讯，下传与上传一一对应，相互关联。

主动请求是指由计算机应用层下传给本地端控制器的数据请求，被动应答是指在本地端控制器收到数据请求后，上传给计算机应用层的应答数据。所有的通讯都由计算机应用层发起，然后以接收到本地端控制器的应答结束，完整的过程包括：

① 计算机应用层将数据请求发送给 CH371 芯片；
② CH371 芯片以中断方式通知本地端控制器；
③ 控制器进入中断程序，查看传输状态信息寄存器中的数据传输方向标志位；
④ 如果是上传，则将上传数据长度寄存器置为 15，然后退出中断程序；
⑤ 如果是下传，则查看下传数据长度寄存器，从数据下传缓冲区中读取数据块；
⑥ 分析接收到的数据块，准备应答数据；
⑦ 将应答数据写入数据上传缓冲区中；
⑧ 将应答数据的长度写入上传数据长度寄存器中，然后退出中断程序；
⑨ CH371 芯片将应答数据返回给计算机；

⑩ 计算机应用层接收到应答数据。

选用请求加应答模式，一般应该在计算机应用层与本地端控制器之间做一些事先约定，以支持相对复杂的数据传输。例如，约定请求数据块或者应答数据块的首字节是后续数据的长度，从而支持大数据块的传输，CH371 芯片将请求数据分为多组不超过 8 个字节的小数据块提交给本地端控制器，而控制器从第一组的首字节得知后续数据的长度，所以也不会提前应答。另外，也可以事先约定一些命令和状态，例如，约定请求数据第二字节为 5AH 时，用于下传后续数据，本地端控制器只需要返回简单的应答（但不能没有应答）；约定请求数据第二字节为 A5H 时，用于上传数据，计算机应用层只需要发送两个字节的请求，而本地端控制器则上传整个数据块。

7、参数

7.1. 绝对最大值（临界或者超过绝对最大值将可能导致芯片工作不正常甚至损坏）

名称	参数说明	最小值	最大值	单位
TA	工作时的环境温度	0	70	℃
TS	储存时的环境温度	-55	125	℃
VCC	电源电压（VCC 接电源，GND 接地）	-0.5	6.5	V
VIO	输入或者输出引脚上的电压	-0.5	VCC+0.5	V
IMAX	单个引脚的驱动电流（输出为正，吸入为负）	-30	30	mA

7.2. 电气参数（测试条件：TA=25℃，VCC=5V，不包括连接 USB 总线的引脚）

名称	参数说明	最小值	典型值	最大值	单位
VCC	电源电压	4.5	5	5.5	V
ICC	电源电流	1	5	50	mA
VIL	低电平输入电压	-0.5		0.8	V
VIH	高电平输入电压	2.0		5.5	V
VOL	低电平输出电压（-4mA）			0.5	V
VOH	高电平输出电压（4mA）	4.5			V
IDN	带下拉的输入端的输入电流		-30	-60	uA
IUPad	带上拉的 A0-A3 和 D0-D7 的输入电流		30	60	uA
IUP	带上拉的其它输入端的输入电流		50	100	uA
IUPsda	带上拉的 SDA 的输入电流	100	200	300	uA
IIN	无上拉/下拉的输入端的输入电流			10	uA
RDN	下拉电阻的阻值（非线性等同值）	40	80	160	KΩ
RUP	上拉电阻的阻值（非线性等同值）	25	65	160	KΩ
RSDA	SDA 引脚的上拉电阻（等同值）	8	16	32	KΩ
RB	XI 引脚与 XO 引脚之间的偏置电阻	500	1000	2000	KΩ
VR	上电复位的默认电压门限	2.5	2.7	3	V

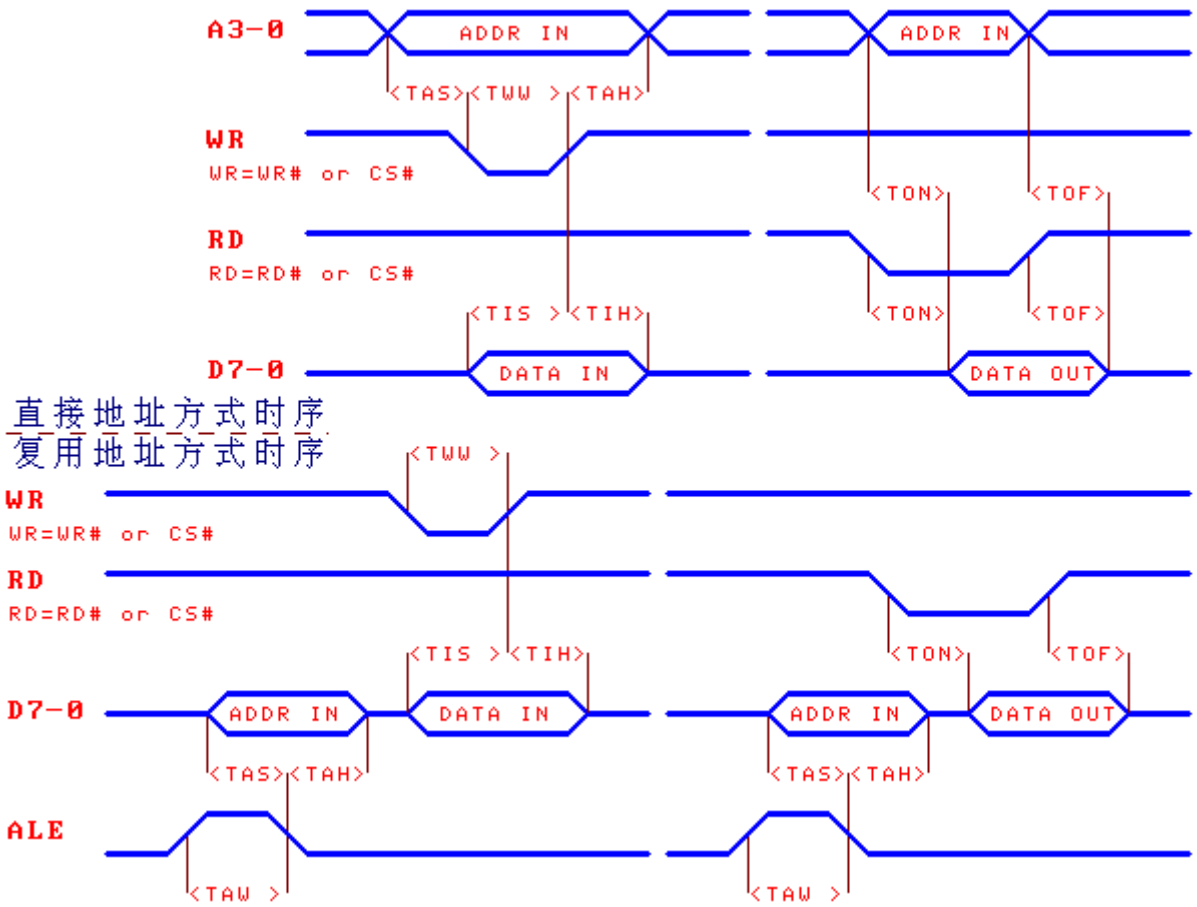
7.3. 被动并行接口的时序参数（测试条件：TA=25℃，VCC=5V，参考附图）

（表中和图中的 RD 是指 RD#信号与 CS#信号同时有效，即 RD#与 CS#的逻辑或）

（表中和图中的 WR 是指 WR#信号与 CS#信号同时有效，即 WR#与 CS#的逻辑或）

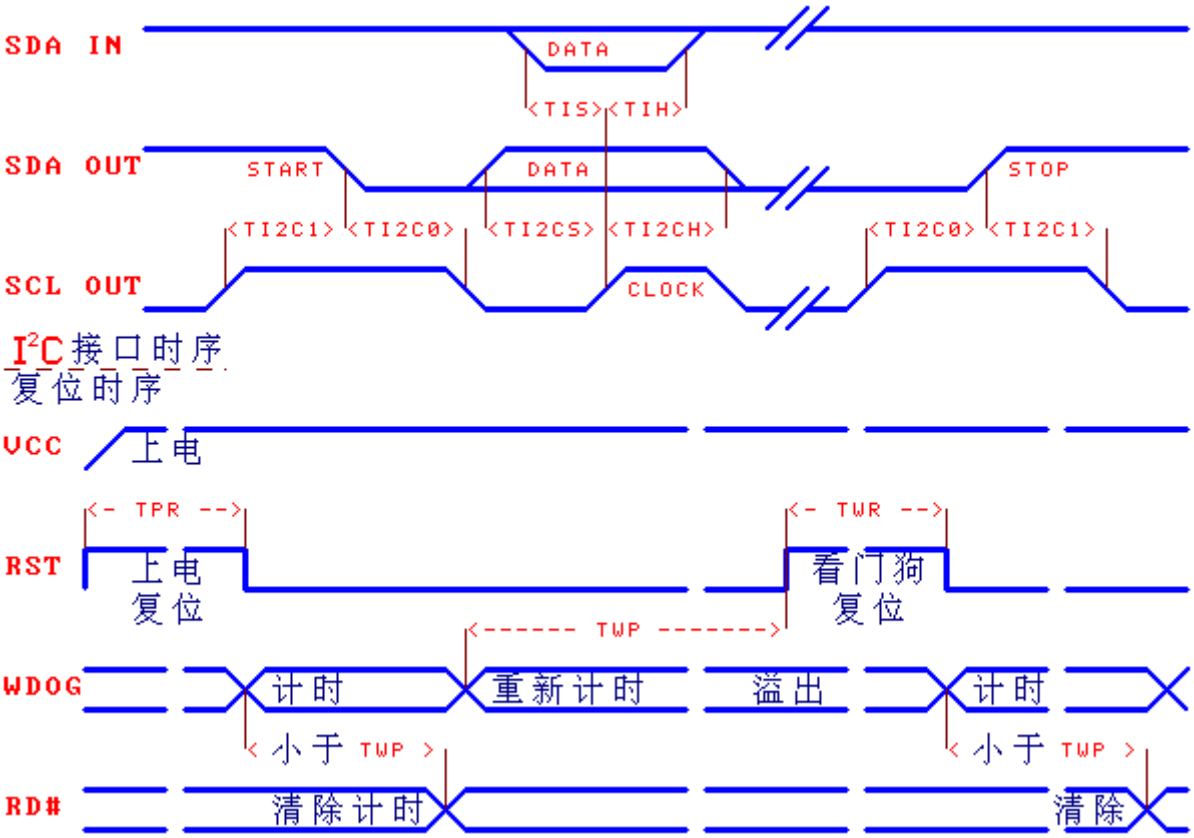
名称	参数说明	最小值	典型值	最大值	单位
TWW	写选通脉冲 WR 的宽度	40			nS

TAW	地址锁存使能脉冲 ALE 的宽度	40			nS
TAS	地址输入建立时间	20			nS
TAH	地址输入保持时间	0			nS
TIS	数据输入建立时间	20			nS
TIH	数据输入保持时间	0			nS
TON	地址或读选通有效到数据输出有效	0		20	nS
TOF	地址或读选通无效到数据输出无效	0		10	nS



7.4. 其余时序参数（测试条件：TA=25℃，VCC=5V，FCLK=12MHz，参考附图）

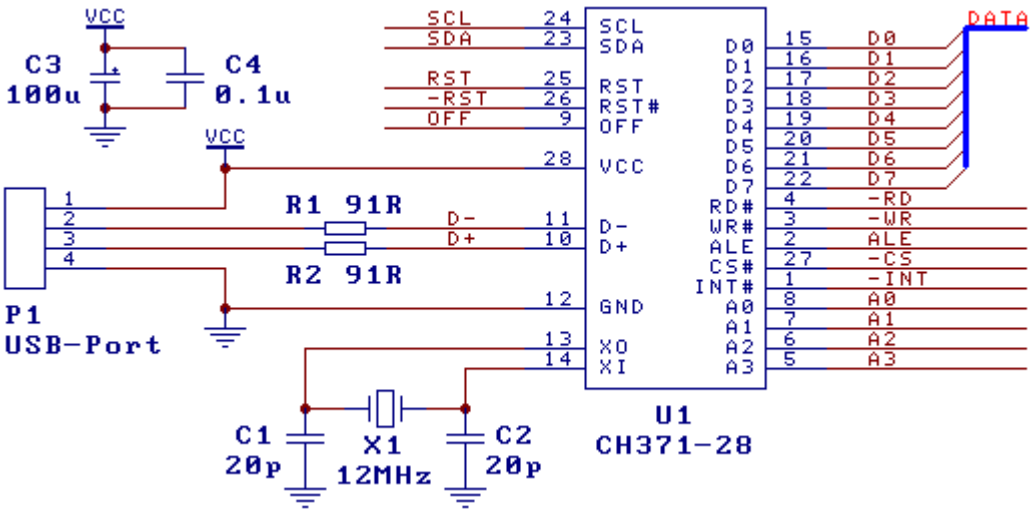
名称	参数说明	最小值	典型值	最大值	单位
FCLK	XI 引脚的系统时钟信号的频率		12		MHz
TPR	上电检测产生的复位脉冲宽度	87.4		175	mS
TWR	看门狗溢出产生的复位脉冲宽度		87.4		mS
TWP	看门狗溢出的周期	524		612	mS
FSCL	SCL 输出频率（I^2C 接口的主频）		375		KHz
TI2C0	开始和终止操作时 SDA 低电平时间		2.67		uS
TI2C1	开始和终止操作时 SDA 高电平时间		2.67		uS
TI2CS	SDA 数据输出建立时间		1.33		uS
TI2CH	SDA 数据输出保持时间		1.33		uS



8、应用

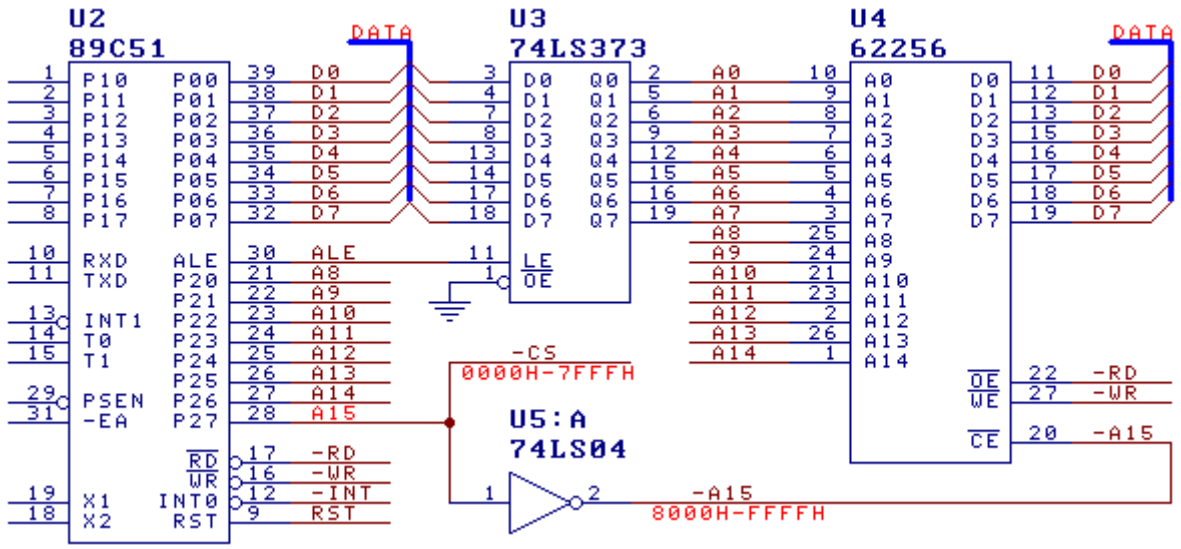
8.1. 连接 USB 总线（下图）

这是 CH371 芯片连接 USB 总线的标准电路，CH371 芯片可以直接使用 USB 总线的 5V 电源。电容 C3 和 C4 用于电源退耦；电阻 R1 和 R2 串接于 CH371 芯片与 USB 总线之间，用于阻抗匹配；晶振 X1、电容 C1 和 C2 用于 CH371 芯片的时钟振荡电路。CH371 芯片的 SCL 和 SDA 信号线可以直接连接 I²C 接口的从设备。例如，连接 24COX 器件，用于存储在系统断电后不能丢失的重要数据，或者存储身份识别数据、记费数据等，CH371 的 I²C 接口与并行接口之间独立，所以 24COX 中的数据只有计算机应用层能够存取。



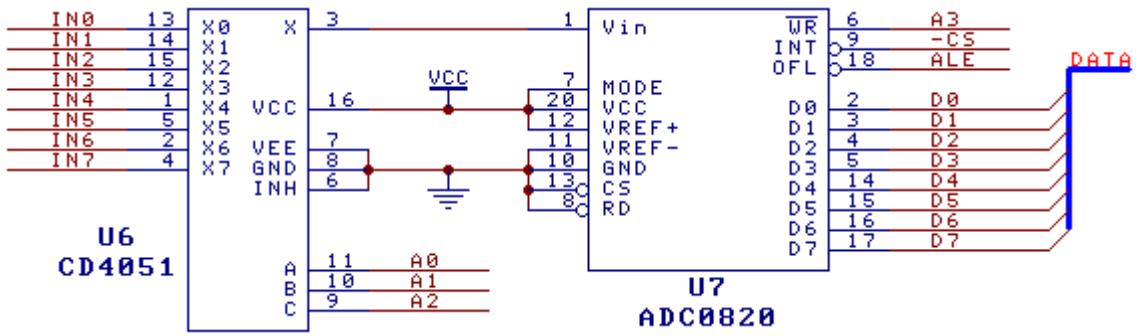
8.2. 连接单片机（下图）

CH371 芯片具有通用的被动并行接口，可以直接连接多种单片机、DSP、MCU 等。下图是常用的 MCS-51 系列的应用电路，CH371 芯片可以通过 D7～D0、-RD、-WR、-CS、ALE 直接挂接到 U2（AT89C51）的系统总线上。连接 ALE 而不连接 A3～A0 则采用复用地址方式，连接 A3～A0 而不连接 ALE 则采用直接地址方式。当采用复用地址方式时，CH371 芯片的 A3～A0 引脚空闲，可以参考主控方式的说明，将 A3～A0 用于 LED 驱动、状态输入等；当采用直接地址方式时，应该将 CH371 芯片的 ALE 引脚悬空或者接高电平，然后通过 A3～A0 直接输入地址。该电路中，CH371 芯片向 U2 提供了上电复位信号 RST，中断输出-INT 连接到 U2 的外部中断 INT0，反相器 U5 用于简单的地址译码，SRAM 器件 62256 的地址为 8000H-FFFFH，CH371 芯片的地址为 0000H-7FFFH（实际只需要 16 个地址）。



8.3. 主控方式应用（下图）

这是基于 CH371 芯片的主控方式设计的 8 通道数据采集器，包括 CH371 在内，只用到了三个集成电路，完全不需要任何单片机、DSP、MCU 等控制器。CH371 芯片的双向引脚 A3～A0 应该设置为输出，A3 用于以低电平启动模数转换，A2～A0 用于选择采集通道，双向引脚 D7～D0 应该设置为输入，用于输入采集结果，-CS、ALE 用于输入采集状态。实际电路应该采用精确的参考电压，并添加辅助电路，例如电源退耦电容等，另外，也可以用自带多通道选择器的模数转换芯片代替 U6 和 U7，例如，8 通道 AD 芯片 MAX158。



8.4. 单片机接口程序

下面提供了 U2（MCS-51 单片机）与 U1（CH371）的接口程序，供参考。

```
;*****************************************************************************
```

```
;需要主程序定义的参数
;CH371_PAGE     EQU    00H             ;CH371 所在的页面地址,地址译码后自动片选
;CH371_SYSTEM   EQU    02H             ;CH371 系统功能设定寄存器的地址偏移
;CH371_CONFIG   EQU    02H             ;CH371 设备配置信息寄存器的地址偏移
;CH371_INT_SET  EQU    06H             ;CH371 中断数据设定寄存器的地址偏移
;CH371_STATUS   EQU    06H             ;CH371 传输状态信息寄存器的地址偏移
;CH371_LENGTH   EQU    07H             ;CH371 数据长度寄存器的地址偏移
;CH371_BUFFER   EQU    08H             ;CH371 数据缓冲区的起始地址偏移
;SAVE_STATUS    DATA   29H             ;保存传输状态信息,根据需要可选
;SAVE_LENGTH    DATA   2AH             ;当前数据缓冲区中的长度,用于保存下传长度
;SAVE_BUFFER    DATA   30H             ;数据缓冲区,用于保存接收到的下传数据
;*******************************************************************************
;
; 初始化子程序
; USE:   ACC, DPTR
CH371_INIT:     MOV    DPH,#CH371_PAGE   ;CH371 所在的页面地址,地址译码后自动片选
                MOV    DPL,#CH371_LENGTH ;CH371 数据长度寄存器的地址偏移
                MOV    A,#0FH
                MOVX   @DPTR,A           ;置上传数据长度寄存器为 15,暂时没有数据上传
                CLR    A                 ;尚未有数据下传
                MOV    SAVE_LENGTH,A     ;保存下传数据长度
                SETB   IT0               ;置外部信号为下降沿触发
                CLR    IE0               ;清中断标志
                SETB   PX0               ;置高优先级
                SETB   EX0               ;允许中断
                RET
;
; 上传数据子程序
; ENTRY: R0 指向存放了准备上传数据的缓冲区, R7 准备上传的数据长度 0 至 8
; USE:   ACC, B, R0, R7, DPTR
CH371_UPLOAD:   MOV    B,R7              ;将数据长度暂存到 B 中
                MOV    DPH,#CH371_PAGE   ;CH371 所在的页面地址,地址译码后自动片选
                MOV    DPL,#CH371_BUFFER ;CH371 数据缓冲区的起始地址偏移
                MOV    A,R7              ;上传数据长度
                JZ     CH371_UPLOAD_0    ;数据长度为 0 则不必写入
CH371_UPLOAD_1: MOV    A,@R0             ;读取一字节的数据
                INC    R0                ;指向下一个数据的地址
                MOVX   @DPTR,A           ;写到 CH371 的上传数据缓冲区
                INC    DPL
                DJNZ   R7,CH371_UPLOAD_1 ;继续读取上传数据直至结束
CH371_UPLOAD_0: MOV    DPL,#CH371_LENGTH ;CH371 数据长度寄存器的地址偏移
                MOV    A,B
                MOVX   @DPTR,A           ;将本次数据的长度置入上传数据长度寄存器
                RET
;
; 中断服务子程序
```

```
; USE:    堆栈 6 字节, 工作寄存器组 1 的 R0,R7
CH371_INTER:    PUSH  PSW                ;现场保护
                CLR   IE0                ;清中断标志,防止重复执行,对应于 INT0 中断
                PUSH  ACC
                PUSH  DPL
                PUSH  DPH
                SETB  RS0                ;PSW.3,切换至工作寄存器组 1
                MOV   DPH,#CH371_PAGE    ;CH371 所在的页面地址,地址译码后自动片选
                MOV   DPL,#CH371_STATUS  ;CH371 传输状态信息寄存器的地址偏移
                MOVX  A,@DPTR            ;读取传输状态信息寄存器
                MOV   SAVE_STATUS,A      ;保存传输状态
                MOV   DPL,#CH371_LENGTH  ;CH371 数据长度寄存器的地址偏移
                JB    ACC.0,CH371_INT_UP ;传输状态信息寄存器位 0 为 1,则指示上传完成
; 是数据下传完成中断
                MOVX  A,@DPTR            ;读取下传数据长度寄存器
                MOV   SAVE_LENGTH,A      ;保持下传数据长度
                JZ    CH371_INT_RET      ;下传数据长度为 0,则直接退出中断
                MOV   DPL,#CH371_BUFFER  ;CH371 数据缓冲区的起始地址偏移
                MOV   R0,#SAVE_BUFFER    ;单片机内部的数据缓冲区,用于存放下传数据
                MOV   R7,A               ;用于读取数据的计数
CH371_INT_DOWN: MOVX  A,@DPTR            ;读取一字节的下传数据
                INC   DPL                ;指向下一个数据的地址
                MOV   @R0,A              ;保存到数据缓冲区
                INC   R0
                DJNZ  R7,CH371_INT_DOWN  ;继续读取下传数据直至结束
                SJMP  CH371_INT_RET      ;接收完下传数据,退出中断
; 是数据上传完成中断
CH371_INT_UP:   MOV   A,#0FH             ;15
                MOVX  @DPTR,A            ;置上传数据长度寄存器为 15,暂时没有后续数据
CH371_INT_RET:                           ;中断返回
                POP   DPH
                POP   DPL
                POP   ACC
                POP   PSW                ;恢复寄存器并选择工作寄存器组 0
                RETI                     ;中断返回
;
;*******************************************************************************
```