



第 2 章 驱动设计的硬件基础

本章讲解底层驱动工程师必备的硬件基础，给出了嵌入式系统硬件原理及分析方法的全景视图。

2.1 节讲解微控制器、微处理器、数字信号处理器以及应用于特定领域的处理器各自的特点。

2.2 节对嵌入式系统中所使用的各类存储器与 CPU 的接口、应用领域及特点进行了详细讲解。

2.3 节讲解常见的外设接口与总线的工作方式，包括串口、I²C、USB、以太网接口、ISA、PCI 和 cPCI 等。

嵌入式系统硬件电路中经常会使用 CPLD 和 FPGA，2.4 节讲解了 CPLD 和 FPGA 在电路中的作用。

2.5~2.7 节讲解实际项目开发过程中硬件分析的方法，包括如何进行原理图分析、时序分析以及如何快速地从芯片手册获取有效信息。

2.8 节讲解了调试过程中常用仪器、仪表的使用方法，涉及万用表、示波器和逻辑分析仪。

2.1

处理器

2.1.1 通用处理器

通用处理器（GPP）并不针对特定的应用领域进行体系结构和指令集的优化，它们具有一般化的通用体系结构和指令集，以支持复杂的运算并易于新开发功能的添加。一般而言，在嵌入式微控制器（MCU）和微处理器（MPU）中会包含一个通用处理器核。

MPU 通常代表一个 CPU（中央处理器），而 MCU 则强调把中央处理器、存储器和外围电路集成在一个芯片中。早期，微控制器被称为单片机，意思是把计算机集成在一个芯片内。嵌入式微控制器也常被称作片上系统（SoC），含义是在一个芯片上设计了整个系统。芯片厂商在推出 MCU 时，往往会有明确的市场定位，如定位于 PDA、MP3、ADSL 等。定位不同的产品可能包含共同的 CPU 核，但是集成的扩展电路则不一样。图 2.1 所示为一个典型的集成了外围电路的 MCU 的结构。

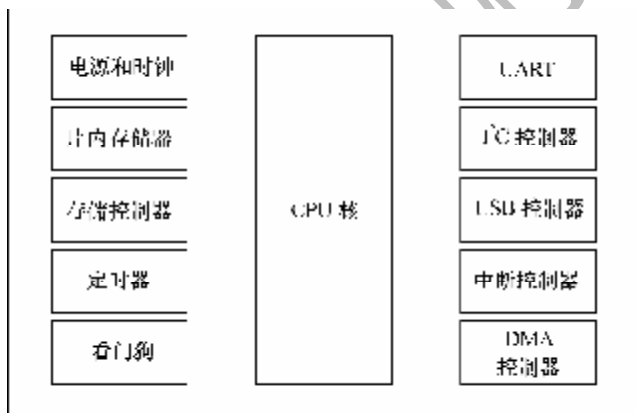


图 2.1 典型的 MCU 内部结构

举个例子，Intel 的 80386 属于微处理器，而内部集成了 80386 处理器、片选单元、中断控制、定时器、看门狗、定时器、串行 I/O、DMA、总线仲裁、DRAM 控制器等的 386EX 则是 80386 微处理器的微控制器版本。

嵌入式微控制器一般由一个 CPU 核和多个外围电路集成，目前主流的 CPU 核如下。

1. Advanced RISC Machines 公司的 ARM

ARM 内核的设计技术被授权给数百家半导体厂商，做成不同的 SoC 芯片。ARM 的功耗很低，在当今最活跃的无线局域网、3G、手机终端、手持设备、有线网络通信设备等中应用非常广泛。

2. Silicon Graphics 公司的 MIPS

两个最重要的 MIPS 芯片厂商为 PMC 和 IDT。PMC-Sierra 公司的 MIPS 处理器被 CISCO 公司大量采用在高端路由器上。IDT 公司在 MIPS 核上集成 PCI 接口，广泛用于以太网交换，另外也尝试增加了 HDLC、Ethernet、串口、SDRAM 控制器、片选、DMA 控制器等外设接口，以用于低端通信产品。

3. IBM 和 Motorola 的 PowerPC

PowerPC 处理器是通信和工控领域应用最广泛的处理器，国内包括华为、中兴在内的通信公司都大量使用 PowerPC，MPC860 和 MPC8260 是其最经典的两款。

4. Motorola 公司独有的内核 68K/COLDFIRE

68K 内核是最早在嵌入式领域广泛应用的内核，其最著名的代表芯片是 68360。Coldfire 则继承了 68K 的特点并对其保持了兼容。Coldfire 内核被用于 DSP 模块、CAN 总线模块以及一般嵌入式处理器所集成的外设模块，在工业控制、机器人研究、家电控制等领域被广泛采用。



Motorola 的半导体部已经独立为飞思卡尔半导体公司(Freescale Semiconductor Inc.)，因为历史原因，上文仍然使用 Motorola。

中央处理器的体系架构可以分为两类，一类为冯·诺伊曼结构，一类为哈佛结构。冯·诺伊曼结构也称普林斯顿结构，是一种将程序指令存储器和数据存储器合并在一起的存储器结构。程序指令存储地址和数据存储地址指向同一个存储器的不同物理位置，因此程序指令和数据的宽度相同。而哈佛结构将程序指令和数据分开存储，指令和数据可以有不同的数据宽度。此外，哈佛结构还采用了独立的程序总线 and 数据总线，分别作为 CPU 与每个存储器之间的专用通信路径，具有较高的执行效率。图 2.2 显示了冯·诺伊曼结构和哈佛结构的区别。

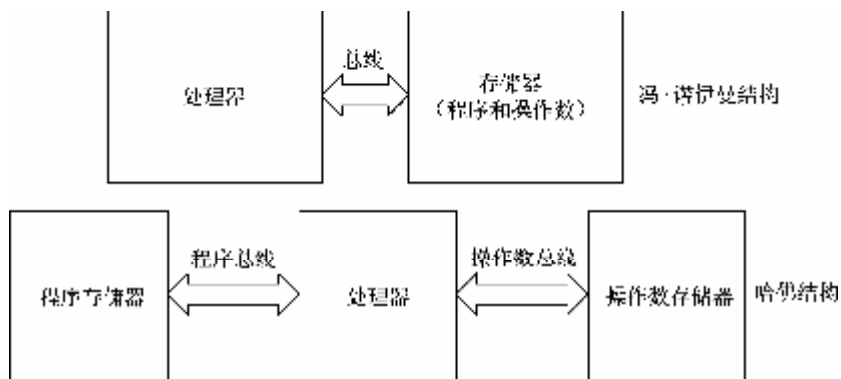


图 2.2 冯·诺伊曼结构与哈佛结构

从指令集的角度来讲，中央处理器也可以分为两类，即 RISC（精简指令集计算机）和 CISC（复杂指令集计算机）。CSIC 强调增强指令的能力、减少目标代码的数

量，但是指令复杂，指令周期长；而 RISC 强调尽可能减少指令集、指令单周期执行，但是目标代码会更大。ARM、MIPS、PowerPC 等 CPU 内核都采用了 RISC 指令集。目前，RISC 和 CSIC 的融合非常明显。

2.1.2 数字信号处理器

数字信号处理器 (DSP) 针对通信、图像、语音和视频处理等领域的算法而设计。它包含独立的硬件乘法器。DSP 的乘法指令一般在单周期内完成，且优化了卷积、数字滤波、FFT (快速傅立叶变换)、相关、矩阵运算等算法中的大量重复乘法。

DSP 一般采用如图 2.3 所示的改进的哈佛架构，它具有独立的地址总线 and 数据总线，两条总线由程序存储器和数据存储器分时共用。

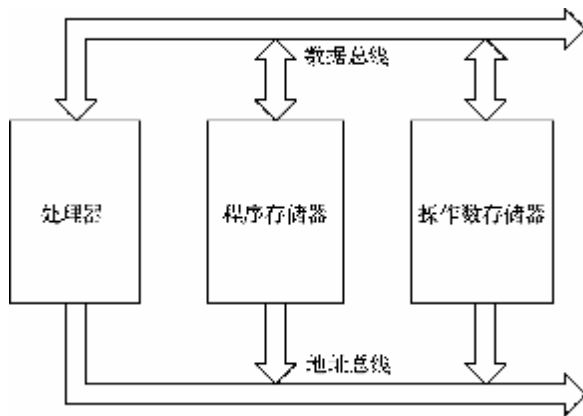


图 2.3 改进的哈佛结构

DSP 分为两类，一类是定点 DSP，一类是浮点 DSP。浮点 DSP 的浮点运算用硬件来实现，可以在单周期内完成，因而其浮点运算处理速度高于定点 DSP。而定点 DSP 只能用定点运算模拟浮点运算。

德州仪器 (TI)、美国模拟器件公司 (ADI) 是全球 DSP 的两大主要厂商。

TI 的 TMS320[™]DSP 平台包含了功能不同的多个系列：2000 系列、3000 系列、4000 系列、5000 系列、6000 系列，工程师也习惯称其为 2X、3X、4X、5X、6X。除了 2000 与 5000 系列是定点 DSP 外，其余的均为浮点 DSP。

ADI 主要有 16 位定点的 21xx 系列、32 位浮点的 SHARC 系列、从 SHARC 系列发展而来的 TigerSHARC 系列及高性能 16 位 DSP 信号处理能力与通用微控制器方便性相结合的 blackfin 系列等。

通用处理器和数字信号处理器也有相互融合以取长补短的趋势，如数字信号控制器 (DSC) 即为 MCU+DSP，blackfin 系列属于 DSC。目前，芯片厂商推出了许多 ARM+DSP 的双核处理器。

除了上述的通用微控制器和数字信号处理器外，还有一些针对特定领域而设计的专用处理器 (ASP)，它们都是针对一些特定应用而设计的，如用于 HDTV、ADSL、Cable Modem 等的专用处理器。

网络处理器是一种可编程器件，它应用于电信领域的各种任务，如包处理、协议分析、路由查找、声音/数据的汇聚、防火墙、QoS 等。网络处理器器件内部通常由若干

微码处理器和若干硬件协处理器组成，多个微码处理器在网络处理器内部并行处理，通过预先编制的微码来控制处理流程。而对于一些复杂的标准操作（如内存操作、路由表查找算法、QoS 的拥塞控制算法、流量调度算法等）则采用硬件协处理器来进一步提高处理性能，从而实现了业务灵活性和高性能的有机结合。

对于某些应用场合，使用 ASIC（专用集成电路）往往是低成本且高性能的方案。ASIC 专门针对特定应用而设计，不具备也不需要灵活的编程能力。使用 ASIC 完成同样地功能往往比直接使用 CPU 资源或 CPLD（复杂可编程逻辑器件）/FPGA（现场可编程门阵列）更廉价、高效。

在实际项目的硬件方案中，往往会根据应用的需求选择通用处理器、数字信号处理器、特定领域处理器、CPLD/FPGA 或 ASIC 之一的解决方案，在复杂的系统中，这些芯片可能会同时存在，协同合作，各自发挥自己的长处。如在一款智能手机中，可使用 MCU 处理图形用户界面和用户的按键输入并运行多任务操作系统，使用 DSP 进行音视频编解码，而在射频方面则采用 ASIC。

综合 2.1 节的内容，我们可得出如图 2.4 所示的处理器分类。

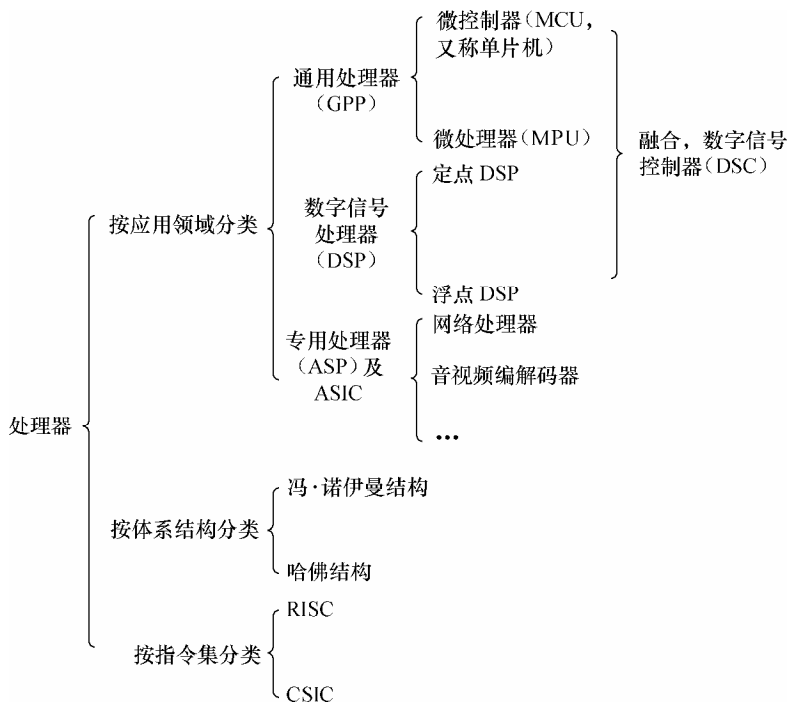


图 2.4 处理器分类

2.2

存储器

存储器可分为只读存储器（ROM）、闪存（Flash）、随机存取存储器（RAM）、光介质存储器和磁介质存储器。

ROM 还可再细分为不可编程 ROM、可编程 ROM (PROM)、可擦除可编程 ROM (EPROM) 和电可擦除可编程 ROM (E²PROM), E²PROM 完全可以用软件来擦写, 已经非常方便了。

目前 ROM 有被 Flash 替代的趋势, NOR (或非) 和 NAND (与非) 是市场上两种主要的 Flash 闪存技术。Intel 公司于 1988 年首先开发出 NOR Flash, 彻底改变了原先由 EPROM 和 E²PROM 一统天下的局面。紧接着, 1989 年, 东芝公司开发出 NAND Flash, 每比特的成本被大大降低。

NOR Flash 和 CPU 的接口属于典型的类 SRAM 接口 (如图 2.5 所示), 不需要增加额外的控制电路。NOR Flash 的特点是可在芯片内执行(XIP, eXecute In Place), 程序可以直接在 NOR 内运行。而 NAND Flash 和 CPU 的接口必须由相应的控制电路进行转换, 当然也可以通过地址线或 GPIO 产生 NAND Flash 接口的信号。NAND Flash 以块方式进行访问, 不支持芯片内执行。

公共闪存接口 (Common Flash Interface, 简称 CFI) 是一个公开的、标准的从 NOR Flash 器件中读取数据的接口。它可以使系统软件查询已安装的 Flash 器件的各种参数, 包括器件阵列结构参数、电气和时间参数以及器件支持的功能等。利用 CFI, 在不修改系统软件的情况下, 就可以用新型的和改进的产品代替旧版本的产品。

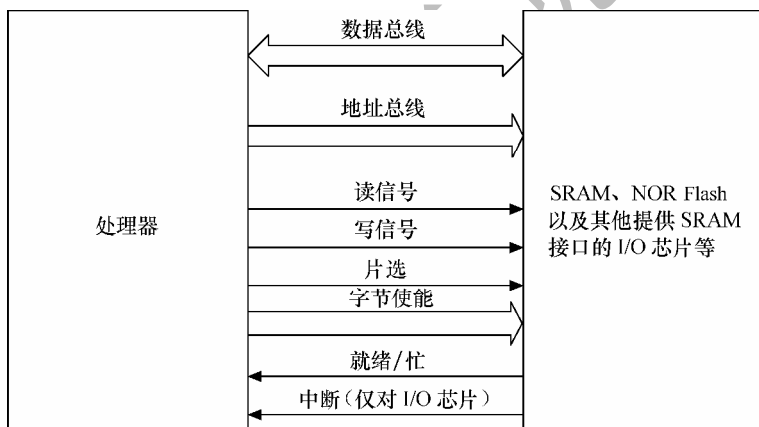


图 2.5 典型的类 SRAM 接口

一个 NAND Flash 的接口主要包含如下信号。

- l I/O 总线: 地址、指令和数据通过这组总线传输, 一般为 8 位或 16 位。
- l 芯片启动 (Chip Enable, CE#): 如果没有检测到 CE# 信号, 那么, NAND 器件就保持待机模式, 不对任何控制信号做出响应。
- l 写使能 (Write Enable, WE#): WE# 负责将数据、地址或指令写入 NAND 之中。
- l 读使能 (Read Enable, RE#): RE# 允许数据输出。
- l 指令锁存使能 (Command Latch Enable, CLE): 当 CLE 为高时, 在 WE# 信号的上升沿, 指令将被锁存到 NAND 指令寄存器中。
- l 地址锁存使能 (Address Latch Enable, ALE): 当 ALE 为高时, 在 WE# 信号的上升沿, 地址将被锁存到 NAND 地址寄存器中。

I 就绪/忙 (Ready/Busy, R/B#): 如果 NAND 器件忙, R/B#信号将变低。该信号是漏极开路, 需要采用上拉电阻。

NAND Flash 与 NOR Flash 相比容量大、价格低; NAND Flash 中每个块的最大擦写次数是 100 万次, 而 NOR 的擦写次数是 10 万次; NAND Flash 的擦除、编程速度远超过 NOR Flash。

由于 Flash 固有的电器特性, 在读写数据过程中偶尔会产生 1 位或几位数据错误, 即位反转, NAND Flash 发生位反转的几率要远大于 NOR Flash。位反转无法避免, 因此, 使用 NAND Flash 的同时, 应采用错误探测/错误更正 (EDC/ECC) 算法。

Flash 的编程原理都是只能将 1 写为 0, 而不能将 0 写为 1。所以在 Flash 编程之前, 必须将对应的块擦除, 而擦除的过程就是把所有位都写为 1 的过程, 块内的所有字节变为 0xFF。

许多嵌入式系统都提供了 IDE (Integrated Drive Electronics) 接口, 以供连接硬盘控制器或光驱, IDE 接口的信号与 SRAM 类似。人们通常也把 IDE 接口称为 ATA (Advanced Technology Attachment) 接口, 技术角度而言并不准确。其实, ATA 接口发展至今, 已经经历了 ATA-1 (IDE)、ATA-2 (EIDE Enhanced IDE/Fast ATA)、ATA-3 (FastATA-2)、Ultra ATA、Ultra ATA/33、Ultra ATA/66、Ultra ATA/100 及 Serial ATA 的发展过程。

以上所述的各种 ROM、Flash 和磁介质存储器都属于非易失性存储器 (NVM), 掉电信息不会丢失, 而 RAM 则与此相反。

RAM 也可再分为静态 RAM (SRAM) 和动态 RAM (DRAM)。DRAM 以电荷形式进行存储, 数据存储在电容器中。由于电容器会由于漏电而导致电荷丢失, 因而 DRAM 器件需要定期被刷新。SRAM 是静态的, 只要供电它就会保持一个值, SRAM 没有刷新周期。每个 SRAM 存储单元由 6 个晶体管组成, 而 DRAM 存储单元由 1 个晶体管和 1 个电容器组成。

通常所说的 SDRAM、DDR SDRAM 皆属于 DRAM 的范畴, 它们采用与 CPU 外存控制器同步的时钟工作 (注意, 不是 CPU 的工作频率)。与 SDRAM 相比, DDR SDRAM 同时利用了时钟脉冲的上升沿和下降沿传输数据, 因此在时钟频率不变的情况下, 数据传输频率加倍。此外, 还存在使用 RSL (Rambus 发信电平) 技术的 RDRAM (Rambus DRAM) 和 Direct RDRAM。

针对许多特定场合的应用, 嵌入式系统中往往还使用了一些特定类型的 RAM。

1. NVRAM (非易失性 RAM)

既然是 RAM, 就是易失性的, 为什么会有一类非易失性的 RAM 呢?

实际上, NVRAM 借助带有备用电源的 SRAM 或借助 NVM (如 E²PROM) 存储 SRAM 的信息并恢复来实现非易失性。NVRAM 的特点是完全像 SRAM 一样读写, 而且写入的信息掉电不丢失, 不需要 E²PROM 和 Flash 的特定擦除和编程操作。NVRAM 多用于存放系统中的参数信息。

2. DPRAM (双端口 RAM)

DPRAM 的特点是可以两个端口同时访问, 具有两套完全独立的数据总线、

地址总线线和读写控制线，通常用于两个处理器之间交互数据，如图 2.6 所示。当一端被写入数据后，另一端可以通过轮询或中断获知，并读取其写入的数据。由于双 CPU 同时访问 DPRAM 时的仲裁逻辑电路集成在 DPRAM 内部，因而硬件工程师设计电路的原理比较简单。

DPRAM 的优点是通信速度快、实时性强、接口简单，而且两边 CPU 都可主动进行数据传输。除了双端口 RAM 以外，目前 IDT 等芯片厂商还推出了多端口 RAM，可以供 3 个以上的 CPU 互通数据。

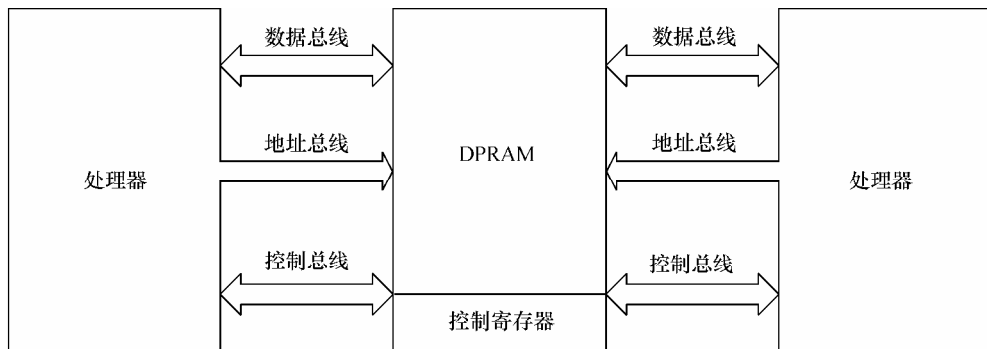


图 2.6 双端口 RAM

3. CAM (内容寻址 RAM)

CAM 是按内容进行寻址的存储器，是一种特殊的存储阵列 RAM，它的主要工作机制就是将一个输入数据项与存储在 CAM 中的所有数据项自动同时进行比较，判别该输入数据项与 CAM 中存储的数据项是否相匹配，并输出该数据项对应的匹配信息。

如图 2.7 所示，在 CAM 中，输入的是所要查询的数据，输出的是数据地址和匹配标志。若匹配（即搜寻到数据），则输出数据地址。CAM 用于数据检索的优势是软件无法比拟的，可以极大地提高系统性能。

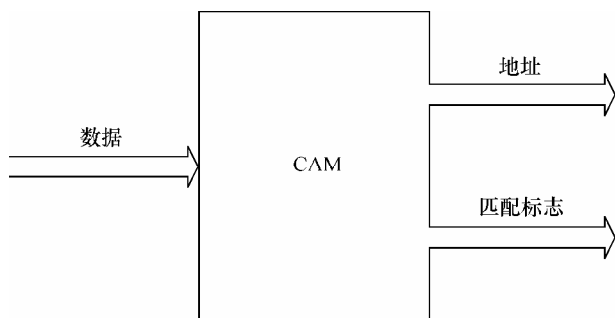


图 2.7 CAM 的输入与输出

4. FIFO (先进先出队列)

FIFO 存储器的特点是先进先出，进出有序，FIFO 多用于数据缓冲。FIFO 和 DPRAM 类似，具有两个访问端口，但是 FIFO 两边的端口并不对等，某一时刻只能被设置为一边作为输入，一边作为输出。

如果 FIFO 的区域共为 n 个字节，我们只能通过循环 n 次读取同一个地址才能将该片区域读出，不能指定偏移地址。对于有 n 个数据的 FIFO，当循环读取 m 次，下一次读会自动读取到第 $m+1$ 个数据，这是由 FIFO 本身的特性决定的。

总结 2.2 节的内容，我们可得出如图 2.8 所示的存储器分类。

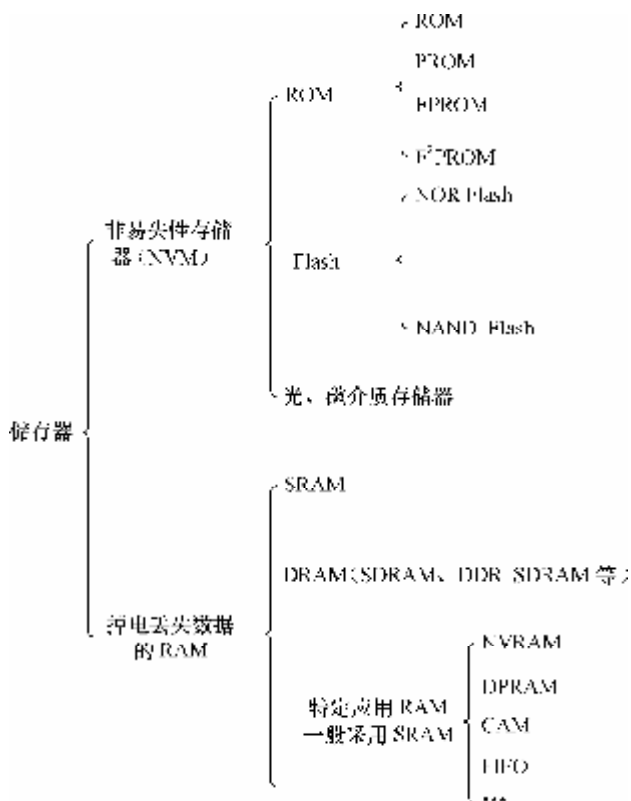


图 2.8 存储器分类

2.3

接口与总线

2.3.1 串口

RS-232、RS-422 与 RS-485 都是串行数据接口标准，最初都是由电子工业协会 (EIA) 制定并发布的。

RS-232 在 1962 年发布，命名为 EIA-232-E。之后发布的 RS-422 定义了一种平衡通信接口，它是一种单机发送、多机接收的单向、平衡传输规范，被命名为 TIA/EIA-422-A 标准。RS-422 改进了 RS-232 通信距离短、速率低的缺点。为进一步扩展应用范围，EIA 又于 1983 年在 RS-422 的基础上制定了 RS-485 标准，增加了多点、双向通信能力，即允许多个发送器连接到同一条总线上，同时增加了发送器的驱动能力和冲突保护特性，并扩展了总线共模范围，被命名为 TIA/EIA-485-A 标准。

1969 年发布的 RS-232 修改版 RS-232C 是嵌入式系统应用最广泛的串行接口，它为连接 DTE（数据终端设备）与 DCE（数据通信设备）而制定。RS-232C 规标准接口有 25 条线（4 条数据线、11 条控制线、3 条定时线、7 条备用和未定义线），常用的只有 9 根，它们是 RTS/CTS（请求发送/清除发送流控制）、RxD/TxD（数据收发）、DSR/DTR（数据终端就绪/数据设置就绪流控制）、DCD（数据载波检测，也称 RLSD）、Ringin-Ri（振铃指示）、SG（信号地）信号。RTS/CTS、TxD/RxD、DSR/DTR 等信号的定义如下。

- l RTS：用来表示 DTE 请求 DCE 发送数据，当终端要发送数据时，使该信号有效。
- l CTS：用来表示 DCE 准备好接收 DTE 发来的数据，是对 RTS 的响应信号。
- l TxD：DTE 通过 TxD 将串行数据发送到 DCE。
- l RxD：DTE 通过 RxD 接收从 DCE 发来的串行数据。
- l DSR：有效（ON 状态）则表明 DCE 可以使用。
- l DTR：有效（ON 状态）则表明 DTE 可以使用。
- l DCD：当本地 DCE 设备收到对方 DCE 设备送来的载波信号时，使 DCD 有效，通知 DTE 准备接收，并且由 DCE 将接收到的载波信号解调为数字信号，经 RxD 线送给 DTE。
- l Ringin-Ri：当调制解调器收到交换台送来的振铃呼叫信号时，使该信号有效（ON 状态），通知终端，已被呼叫。

在嵌入式系统中，并不太注重 DTE 和 DCE 的概念，而 RS-232C 也很少用来连接调制解调器，多使用 RS-232C 进行对等通信，如 Windows 超级终端、Linux minicom 用来连接电路板控制台等。最简单的 RS-232C 串口只需要连接 RxD、TxD、SG 这 3 个信号，使用 XON/XOFF 软件流控。

组成一个 RS-232C 串口的硬件原理如图 2.9 所示，从 CPU 到连接器依次为 CPU、UART（通用异步接收器发送器，作用是完成并/串转换）、CMOS/TTL 电平与 RS-232C 电平转换器、DB9/DB25 或自定义的连接器。

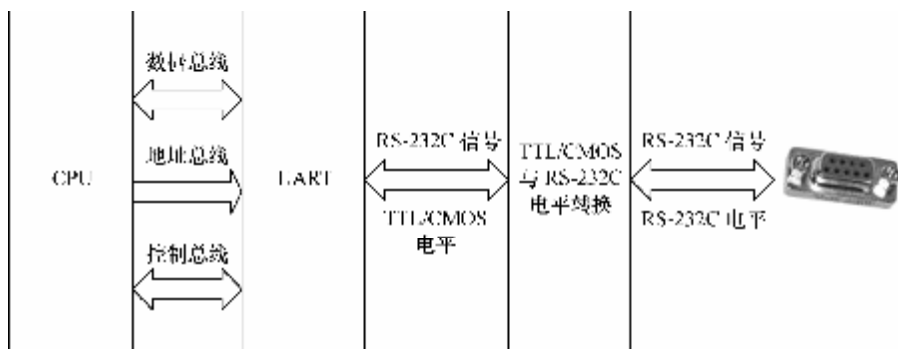


图 2.9 RS-232C 串口电路原理

2.3.2 I²C 总线

I²C（内置集成电路）总线是由 Philips 公司开发的两线式串行总线，产生于 20 世纪 80 年代，用于连接微控制器及其外围设备。I²C 总线简单而有效，占用很少的 PCB（印

刷电路板)空间,芯片管脚数量少,设计成本低。 I^2C 总线支持多主控 (multi-mastering) 模式,任何能够进行发送和接收的设备都可以成为主设备。主控能够控制数据的传输和时钟频率,在任意时刻只能有一个主控。

组成 I^2C 总线的两个信号为数据线 SDA 和时钟线 SCL。为了避免总线信号的混乱,要求各设备连接到总线的输出端必须是开漏输出或集电极开路输出的结构。总线空闲时,上拉电阻使 SDA 和 SCL 线都保持高电平。根据开漏输出或集电极开路输出信号的“线与”逻辑, I^2C 总线上任意器件输出低电平都会使相应总线上的信号线变低。



“线与”逻辑指的是两个或两个以上的输出直接互连就可以实现“AND”的逻辑功能,只有输出端是开漏(对于 CMOS 器件)输出或集电极开路(对于 TTL 器件)输出时才满足此条件。工程师一般以“OC 门”简称开漏或集电极开路。

I^2C 设备上的串行数据线 SDA 接口电路是双向的,输出电路用于向总线发送数据,输入电路用于接收总线上的数据。同样地,串行时钟线 SCL 也是双向的,作为控制总线数据传送的主机要通过 SCL 输出电路发送时钟信号,并检测总线上 SCL 上的电平以决定什么时候发下一个时钟脉冲电平;作为接收主机命令的从设备需按总线上 SCL 的信号发送或接收 SDA 上的信号,它也可以向 SCL 线发出低电平信号以延长总线时钟信号周期。

当 SCL 稳定在高电平时,SDA 由高到低的变化将产生一个开始位,而由低到高的变化则产生一个停止位,如图 2.10 所示。

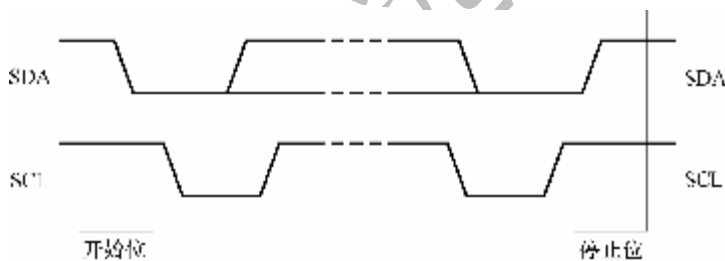
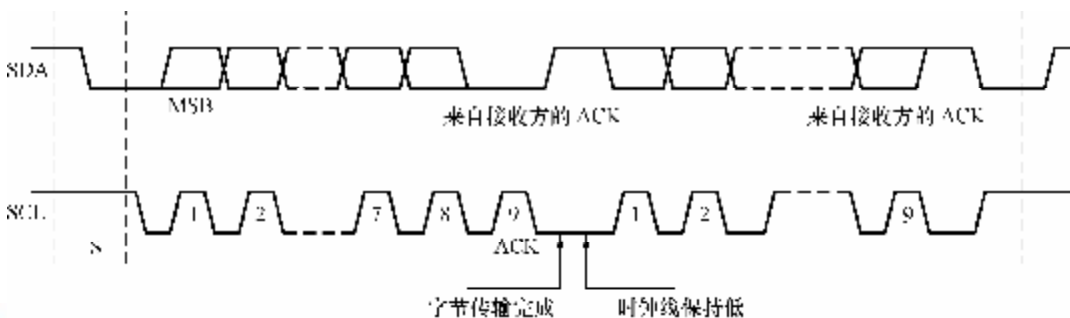


图 2.10 I^2C 总线开始位和停止位

开始位和停止位都由 I^2C 主设备产生。在选择从设备时,如果从设备采用 7 位地址,则主设备在发起传输过程前,需先发送 1 字节的地址信息,前 7 位为设备地址,最后 1 位为读写标志。之后,每次传输的数据也是 1 个字节,从 MSB 位开始传输。每个字节传完后,在 SCL 的第 9 个上升沿到来之前,接收方应该发出 1 个 ACK 位。SCL 上的时钟脉冲由 I^2C 主控方发出,在第 8 个时钟周期之后,主控方应该释放 SDA, I^2C 总线的时序如图 2.11 所示。



2.3.3 USB

USB（通用串行总线）是 Intel、Microsoft 等厂商为解决计算机外设种类的日益增加与有限的主板插槽和端口之间的矛盾而于 1995 年提出的，它具有数据传输率高、易扩展、支持即插即用和热插拔的优点，目前已得到广泛的应用。

USB1.1 包含全速和低速两种模式，低速方式的速率为 1.5Mbit/s，支持一些不需要很大数据吞吐量和很高实时性的设备，如鼠标等。全速模式为 12Mbit/s，可以外接速率更高的外设。在 USB 2.0 中，增加了一种高速方式，数据传输率达到 480Mbit/s，可以满足更高速外设的需要。

USB 的物理拓扑结构如图 2.12 所示，在 USB 2.0 中，高速方式下 Hub 使全速和低速方式的信令环境独立出来，图 2.13 所示为高速方式下 Hub 的作用。

在嵌入式系统中，电路板若需要挂接 USB 设备(device)，则需提供 USB 主机(host)控制器和连接器；若电路板需要作为 USB 设备，则需提供 USB 设备适配器和连接器。有的 MCU 集成了 USB 主机控制器和设备适配器。

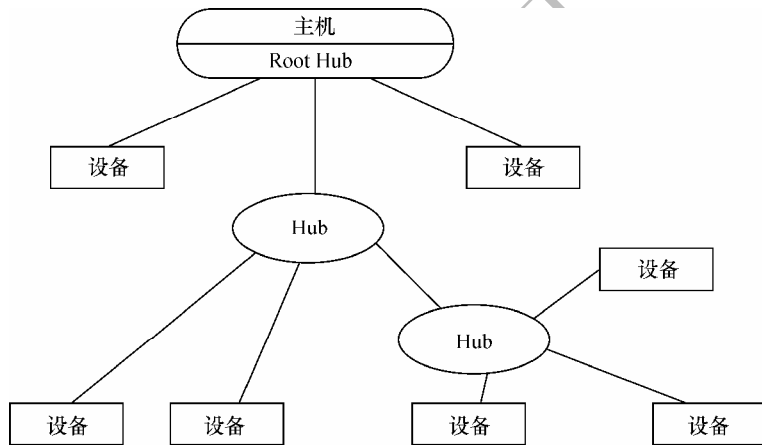


图 2.12 USB 的物理拓扑

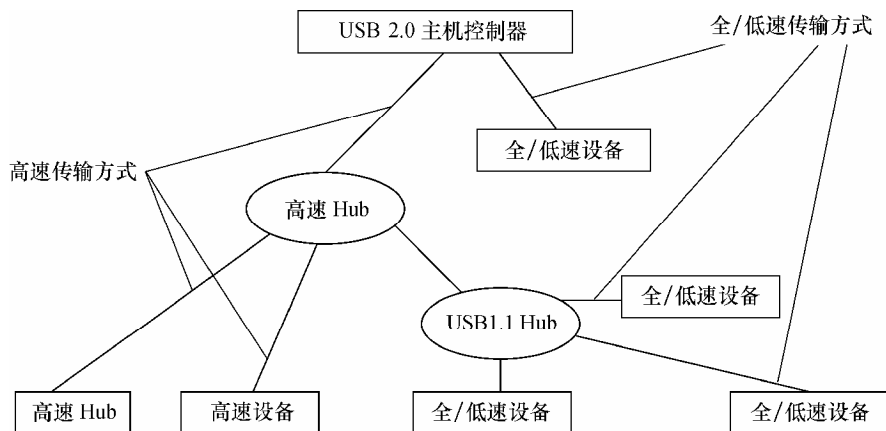


图 2.13 USB 2.0 连接高速、全/低速设备

USB 总线的机械连接非常简单，采用 4 芯的屏蔽线，一对差分线（D+，D-）传

送信号，另一对（VBUS，电源地）传送+5V 的直流电。一个 USB 主控制器端口最多可连接 127 个器件，各器件之间的距离不超过 5 米。

USB 提供了 4 种传输方式以适应各种设备的需要，说明如下。

(1) 控制（Control）传输方式。

控制传输是双向传输，数据量通常较小，主要用来进行查询、配置和给 USB 设备发送通用的命令。

(2) 同步（Synchronization）传输方式。

同步传输提供了确定的带宽和间隔时间，它被用于时间严格并具有较强容错性的流数据传输，或者用于要求恒定的数据传送率的即时应用。例如进行语音业务传输时，使用同步传输方式是很好的选择。

(3) 中断（Interrupt）传输方式。

中断方式传送是单向的，对于 USB 主机而言，只有输入。中断传输方式主要用于定时查询设备是否有中断数据要传送，该传输方式应用在少量的、分散的、不可预测的数据传输场合，键盘、游戏杆和鼠标属于这一类型。

(4) 批量（Bulk）传输方式。

批量传输主要应用在没有带宽和间隔时间要求的大量数据的传送和接收，它进行有保证的传输。打印机和扫描仪均采用这种传输类型。

2.3.4 以太网接口

以太网接口由 MAC（以太网媒体接入控制器）和 PHY（物理接口收发器）组成。以太网 MAC 由 IEEE-802.3 以太网标准定义，实现了数据链路层。常用的 MAC 支持 10Mbit/s 或 100Mbit/s 两种速率。PHY 则实现物理层功能，IEEE-802.3 标准定义了以太网 PHY，它符合 IEEE-802.3k 中用于 10BaseT（第 14 条）和 100BaseTX（第 24 条和第 25 条）的规范。10BaseT 和 100BaseTX PHY 两种实现的帧格式是一样的，但信令机制不同，而且 10BaseT 采用曼彻斯特编码，100BaseTX 采用 4B/5B 编码。

MAC 和 PHY 之间采用 MII（媒体独立接口）连接，它是 IEEE-802.3 定义的以太网行业标准，包括 1 个数据接口和 1 个 MAC 和 PHY 之间的管理接口。数据接口包括分别用于发送和接收的两条独立信道，每条信道都有自己的数据、时钟和控制信号，MII 数据接口总共需要 16 个信号。MII 管理接口包含两个信号，一个是时钟信号，另一个是数据信号。通过管理接口，上层能监视和控制 PHY。

以太网接口电路原理如图 2.14 所示，从 CPU 到最终接口依次为 CPU、MAC、PHY、以太网隔离变压器、RJ45 插座。以太网隔离变压器是以太网收发芯片与连接器之间的磁性组件，在其两者之间起着信号传输、阻抗匹配、波形修复、信号杂波抑制和高电压隔离作用。

许多处理器内部集成了 MAC 或同时集成了 MAC 和 PHY，另有许多以太网控制芯片也集成了 MAC 和 PHY。

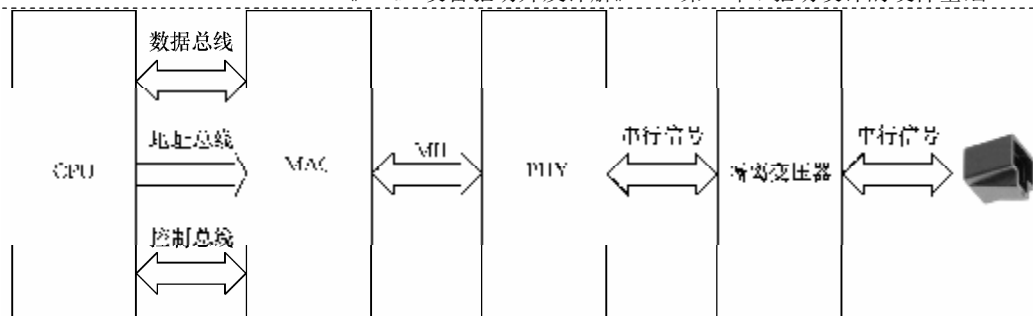


图 2.14 以太网接口电路原理

2.3.5 ISA 总线

ISA（工业标准结构总线）总线起源于 1981 年 IBM 公司生产的以 Intel 8088 为 CPU 的 IBM-PC 微计算机，开始时总线宽度为 8 位。1984 年推出的 IBM-PC/AT 系统将 ISA 总线扩充为 16 位数据总线宽度，同时地址总线宽度也由 20 位扩充到了 24 位。其后推出的 EISA（扩展的 ISA）采用 32 位地址线，数据总线也扩展为 32 位，但仍保持了与 ISA 的兼容。

图 2.15 所示为 ISA 总线的信号，这些信号可分为 3 组。

- ┆ 总线基本信号：ISA 总线工作所需要的最基本信号，含复位、时钟、电源、地等。
- ┆ 总线访问信号：用于访问 ISA 总线设备的地址线、数据线以及相应的应答信号。
- ┆ 总线控制信号：中断和 DMA 请求。

图 2.15 中各信号的详细定义如下。

- ┆ RESET、BCLK：复位及总线基本时钟，BCLK 为 8MHz。
- ┆ SA19~SA0：存储器及 I/O 空间 20 位地址，带锁存。
- ┆ LA23~LA17：存储器及 I/O 空间 20 位地址，不带锁存。
- ┆ BALE：总线地址锁存，外部锁存器的选通。
- ┆ AEN：地址允许，表明 CPU 让出总线，DMA 开始。
- ┆ SMEMR#、SMEMW#：8 位 ISA 存储器读写控制。
- ┆ MEMR#、MEMW#：16 位 ISA 存储器读写控制。
- ┆ SD15~SD0：数据总线，访问 8 位 ISA 卡时高 8 位自动传送到 SD7~SD0。
- ┆ SBHE#：高字节允许，打开 SD15~SD8 数据通路。
- ┆ MEMCS16#、IOCS16#：ISA 卡发出此信号确认可以进行 16 位传送。
- ┆ I/OCHRDY：ISA 卡准备信号，可控制插入等待周期。
- ┆ NOWS#：有效则暗示不用插入等待周期。
- ┆ I/OCHCK#：ISA 卡奇偶校验错。
- ┆ IRQ15、IRQ14、IRQ12~IRQ9、IRQ7~IRQ3：中断请求。
- ┆ DRQ7~DRQ5、DRQ3~DRQ0：ISA 卡 DMA 请求信号。
- ┆ DACK7#~DACK5#、DACK3#~DACK0#：DMA 请求响应信号。
- ┆ MASTER#：ISA 主模块确立信号，ISA 发出此信号，与主机内 DMAC（DMA

控制器) 配合使 ISA 卡成为主模块。

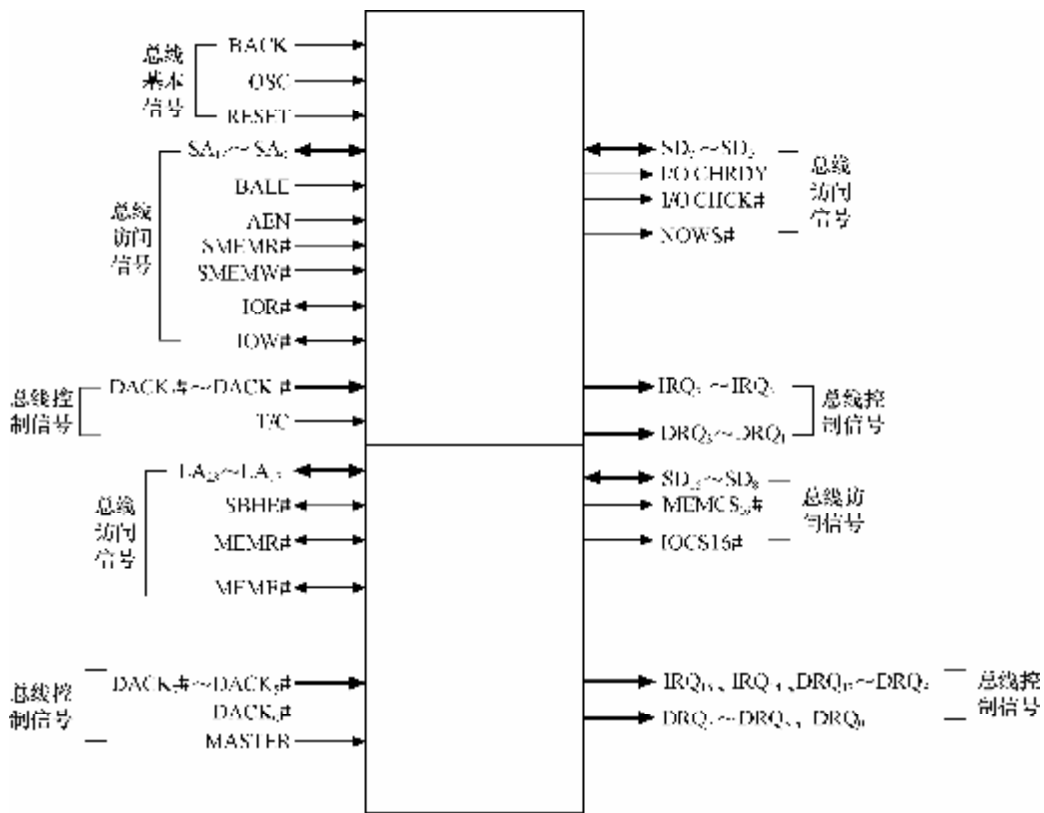


图 2.15 ISA 总线信号

2.3.6 PCI 和 cPCI

PCI (外围部件互连) 是 Intel 公司于 1991 年推出的一种局部总线, 作为一种通用的总线接口标准, 它在目前的计算机系统中得到了非常广泛的应用。PCI 提供了一组完整的总线接口规范, 其目的是描述如何将计算机系统的外围设备以一种结构化和可控化的方式连接在一起, 给出了外围设备在连接时的电气特性和行为规约, 并且详细定义了计算机系统各个不同部件之间应该如何正确地进行交互。PCI 总线具有如下特点。

- l 数据总线 32 位, 可扩充到 64 位。
- l 可进行突发 (burst) 模式传输。



突发方式传输是指取得总线控制权后连续进行多个数据的传输。突发传输时, 只需要给出目的地的首地址, 访问第 1 个数据后, 第 2~n 个数据会在首地址基础上按一定规则自动被寻址和传输。与突发方式对应的是单周期方式, 它在 1 个总线周期只传送 1 个数据。

- l 总线时钟频率为 33MHz 或 66MHz, 最高传输率可达 528MB/s。
- l 采用中央集中式总线仲裁。
- l 支持全自动配置、资源分配, PCI 卡内有设备信息寄存器组为系统提供卡的信息, 可实现即插即用。

l PCI 总线规范独立于微处理器，通用性好。

l PCI 设备可以完全作为主控设备控制总线。

图 2.16 所示为一个典型的基于 PCI 总线的计算机系统逻辑示意图，系统的各个部分通过 PCI 总线和 PCI-PCI 桥连接在一起。CPU 和 RAM 通过 PCI 桥连接到 PCI 总线 0（即主 PCI 总线），而具有 PCI 接口的显卡则可以直接连接到主 PCI 总线上。PCI-PCI 桥是一个特殊的 PCI 设备，它负责将 PCI 总线 0 和 PCI 总线 1（即从 PCI 总线）连接在一起，通常 PCI 总线 1 称为 PCI-PCI 桥的下游（downstream），而 PCI 总线 0 则称为 PCI-PCI 桥的上游（upstream）。为了兼容旧的 ISA 总线标准，PCI 总线还可以通过 PCI-ISA 桥来连接 ISA 总线，从而支持以前的 ISA 设备。

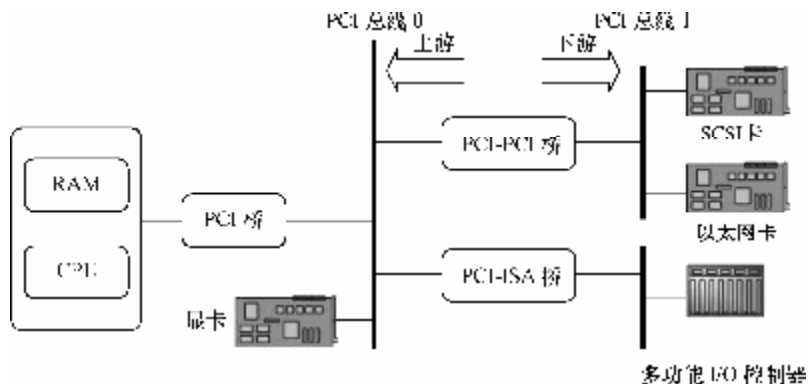


图 2.16 基于 PCI 总线的计算机系统

当 PCI 卡刚加电时，卡上只有配置空间是可被访问的，因而 PCI 卡开始不能由驱动或用户程序访问，这与 ISA 卡有本质的区别（CPU 可直接读取 ISA 卡在存储空间或 I/O 空间映射的地址）。PCI 配置空间保存着该卡工作所需的所有信息，如厂家、卡功能、资源要求、处理能力、功能模块数量、主控卡能力等。通过对这个空间信息的读取与编程，可完成对 PCI 卡的配置。图 2.17 中 PCI 配置空间共为 256 字节，主要包括如下信息。

- l 制造商标识（Vendor ID）：由 PCI 组织分配给厂家。
- l 设备标识（Device ID）：按产品分类给本卡的编号。
- l 分类码（Class Code）：本卡功能的分类码，如图卡、显示卡、解压卡等。
- l 申请存储器空间：PCI 卡内有存储器或以存储器编址的寄存器和 I/O 空间，为使驱动程序和应用程序能访问它们，需申请 CPU 的一段存储区域以进行定位。配置空间的基地址寄存器用于此目的。
- l 申请 I/O 空间：配置空间的基地址寄存器也用来进行系统 I/O 空间的申请。
- l 中断资源申请：配置空间中的中断引脚和中断线用来向系统申请中断资源。中断资源的申请通过中断引脚（interrupt pin）和中断线（interrupt line）来完成的。偏移 3DH 处为中断引脚寄存器，其值表明 PCI 设备使用了哪一个中断引脚，对应关系为 1-INTA#、2-INTB#、3-INTC#、4-INTD#。

PCI 总线上的信号大体可分为如下几组：

- l 系统接口信号；

- | 地址与数据接口信号；
- | 接口控制信号；
- | 仲裁信号；
- | 错误报告信号；
- | 中断接口信号；
- | 其他接口信号。

| | | | | | | | | |
|------------|--|---------|--|----------|--|------|--|--|
| 31 | | 16 | | 15 | | 0 | | |
| 设备标识 | | | | 制造商标识 | | | | 00H |
| 状态 | | | | 命令 | | | | 04H |
| 分类码 | | | | | | 修正标志 | | 08H |
| BIST | | 头类型 | | 延迟定时器 | | 行大小 | | 0CH |
| 基地址寄存器 | | | | | | | | 10H 14H 18H 1CH 20H 24H |
| 卡总线 CIS 指针 | | | | | | | | 28H |
| 子系统标识 | | | | 子系统制造商标识 | | | | 2CH |
| 扩展 ROM 基地址 | | | | | | | | 30H |
| 保留 | | | | | | 容量指针 | | 34H |
| 保留 | | | | | | | | 38H |
| Max_Lat | | Max_Gnt | | 中断引脚 | | 中断线 | | 3CH |

图 2.17 PCI 配置空间

如图 2.18 所示，这些信号的详细定义如下。

- | CLK：系统时钟。
- | AD31~AD0：地址和数据复用信号线信号。
- | C/BE3~C/BE0：总线命令和地址使能信号。
- | PAR：奇偶校验信号。
- | FRAME#：帧周期信号，指示总线操作起始和终止。
- | IRDY#：主设备准备好信号。
- | TRDY#：目标设备准备好信号。
- | STOP#：目标设备要求终止当前数据传输信号。
- | DEVSEL#：目标设备选中信号。
- | IDSEL：配置空间读写时的片选信号。
- | LOCK#：总线锁定信号。

- l RST#：复位信号。
- l INTA#、INTB#、INTC#和 INTD#：中断请求。
- l REQ#、GNT#：PCI 总线请求与仲裁后的授权。
- l AD63~AD32、C/BE7~C/BE4 等：作用于扩展的 64 位 PCI 总线。

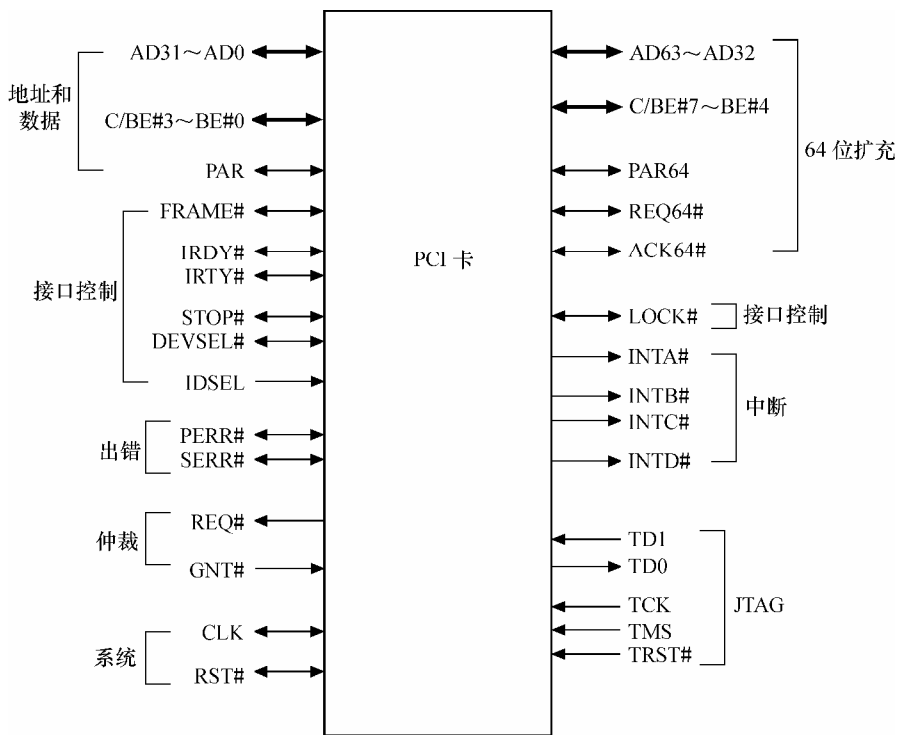


图 2.18 PCI 总线信号

cPCI (Compact PCI, 紧凑型 PCI) 是以 PCI 电气规范为标准的高性能工业用总线，结合了 VME (Visa Module Eurocard, 维萨信用卡模块欧洲卡) 的高性能、可扩展性和可靠性与 PCI 标准的经济有效和灵活性。cPCI 的 CPU 及外设与标准 PCI 是相同的，使用与传统 PCI 相同的芯片和软件，操作系统、驱动和应用程序都感觉不到两者的区别。图 2.19 所示为与 cPCI 总线相关的板卡、背板和机箱，应用于工业控制和大型通信设备。



图 2.19 cPCI 板卡、背板与机箱

2.4

CPLD 和 FPGA

CPLD（复杂可编程逻辑器件）由完全可编程的与或门阵列以及宏单元构成。

CPLD 中基本逻辑单元是宏单元，宏单元由一些“与或”阵列加上触发器构成，其中“与或”阵列完成组合逻辑功能，触发器完成时序逻辑。宏单元中与阵列的输出称为乘积项，其数量标志了 CPLD 的容量。乘积项阵列实际上就是一个“与或”阵列，每一个交叉点都是一个可编程熔丝，如果导通就是实现“与”逻辑。在“与”阵列后一般还有一个“或”阵列，用以完成最小逻辑表达式中的“或”关系。图 2.20 给出了非常典型的 CPLD 的单个宏单元的结构。

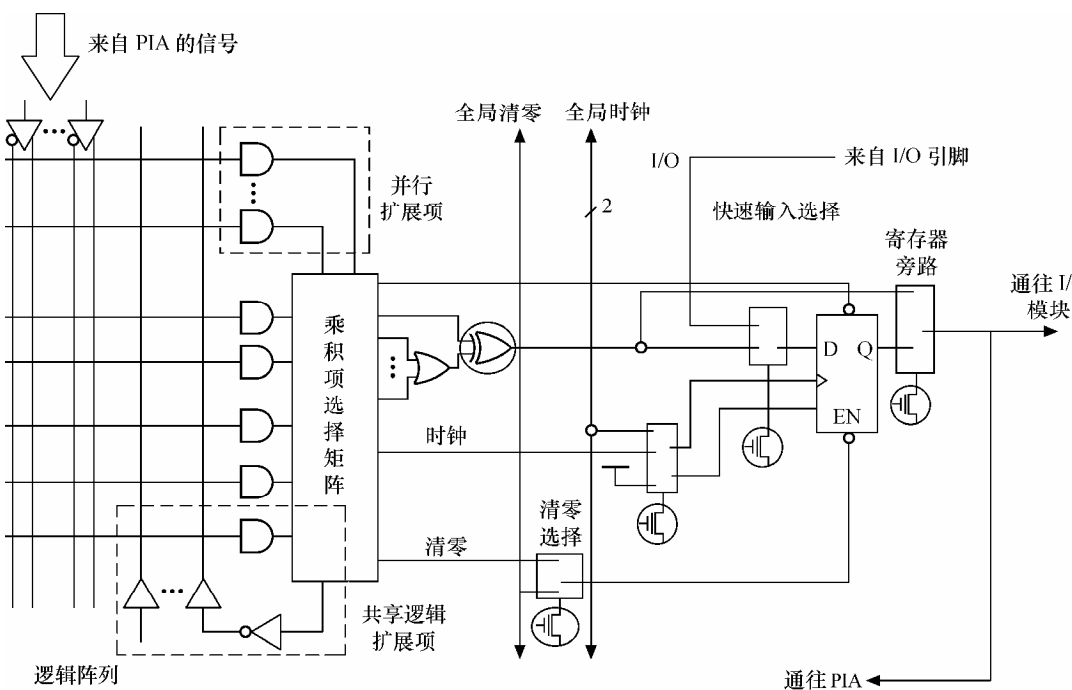


图 2.20 典型的 CPLD 宏单元

图 2.21 给出了一个典型 CPLD 的整体结构。这个 CPLD 由 LAB（逻辑阵列模块，由多个宏单元组成）通过 PIA（可编程互连阵列）互连组成，而 CPLD 与外部的接口则由 I/O 控制模块提供。

图中宏单元的输出会经 I/O 控制块送至 I/O 引脚，I/O 控制块控制每一个 I/O 引脚的工作模式，决定其为输入、输出还是双向引脚，并决定其三态输出的使能端控制。

与 CPLD 不同，FPGA（现场可编程门阵列）基于 LUT（查找表）工艺。查找表本质上是一片 RAM，当用户通过原理图或 HDL（硬件描述语言）语言描述了一个逻辑电路以后，FPGA 开发软件会自动计算逻辑电路的所有可能的结果，并把结果事先

写入 RAM。这样，输入一组信号进行逻辑运算就等于同输入一个地址进行查表，并输出对应地址的内容。

图 2.22 所示为一个典型的 FPGA 的内部结构。这个 FPGA 由 IOC（输入/输出控制模块）、EAB（嵌入式阵列块）、LAB 和 FAST TRACK（快速通道互连）构成。

IOC 是内部信号到 I/O 引脚的接口，它位于 FAST TRACK 的行和列的末端，每个

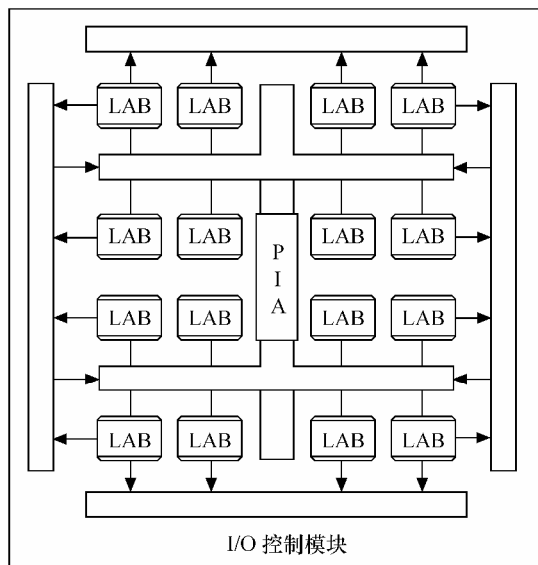


图 2.21 典型 CPLD 的结构

IOC 包含一个双向 I/O 缓冲器和一个既可做输入寄存器，也可做输出寄存器的触发器。

EAB（嵌入式存储块）是一种输入输出端带有寄存器的非常灵活的 RAM。EAB 不仅可以用作存储器，也可以事先被写入查表值来以构成如乘法器、纠错逻辑等电路。当用于 RAM 时，EAB 可配制成 8 位、4 位、2 位和 1 位长度的数据格式。

LAB 主要用于逻辑电路设计，一个 LAB 包括多个 LE（逻辑单元），每个 LE 包括组合逻辑及一个可编程触发器。一系列 LAB 构成的逻辑阵列用来实现普通逻辑功能，如计数器、加法器、状态机等。

器件内部信号的互连和器件引出端之间的信号互连由 FAST TRACK 连线提供，FAST TRACK 遍布于整个 FPGA 器件，是一系列水平和垂直走向的连续式布线通道。

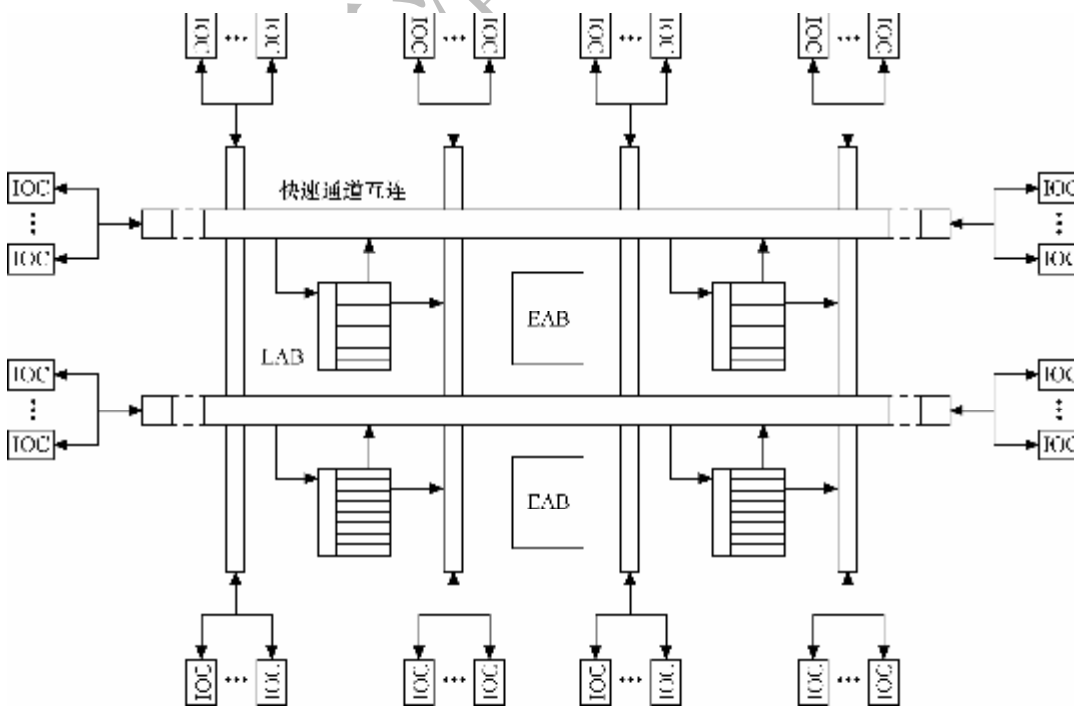
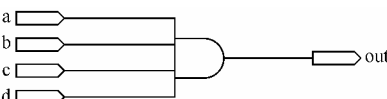
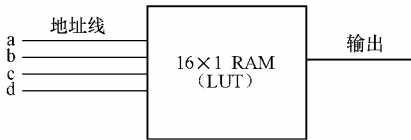


图 2.22 典型 FPGA 的结构

表 2.1 所示为一个 4 输入 LUT 的实际逻辑电路与 LUT 实现方式的对应关系。

表 2.1 实际逻辑电路与查找表的实现

| 实际逻辑电路 | | LUT 的实现方式 | |
|--|------|---|------------|
|  | |  | |
| A、b、c、d 输入 | 逻辑输出 | 地址 | RAM 中存储的内容 |
| 0000 | 0 | 0000 | 0 |
| 0001 | 0 | 0001 | 0 |
| ... | 0 | ... | 0 |
| 1111 | 1 | 1111 | 1 |

CPLD 和 FPGA 的主要厂商有 Altera、Xilinx 和 Lattice 等，采用专门的开发流程，在设计阶段使用 HDL 语言（如 VHDL、Verilog HDL）编程。它们可以实现许多复杂的功能，如实现 UART、I²C 等 I/O 控制芯片、通信算法、音视频编解码算法等，甚至还可以直接集成 ARM 等 CPU 核和外围电路。

对于驱动工程师而言，我们只需要这样看待 CPLD 和 FPGA：如果它完成的是特定的接口和控制功能，我们就直接把它当成由很多逻辑门（与、非、或、D 触发器）组成的完成一系列时序逻辑和组合逻辑的 ASIC；如果它完成的是 CPU 的功能，我们就直接把它当成 CPU。驱动工程师眼里的硬件比 IC 设计师要宏观。

2.5

原理图分析

2.5.1 原理图分析的内容

原理图分析的含义是指通过阅读电路板的原理图获得各种存储器、外设所使用的硬件资源，主要包括存储器和外设控制芯片所使用的片选、中断和 DMA 资源。通过分析片选得出芯片的内存、I/O 基地址，通过分析中断、DMA 信号获得芯片使用的中断号和 DMA 通道，并归纳出如表 2.2 所示的表格。

表 2.2 存储器、外设控制器资源占用表

| 芯 片 | 片 选 | 基 地 址 | 字 长 | 大 小 | 中 断 号 | DMA 通 |
|-------|-----|------------|-----|------------|-------|-------|
| Flash | CS0 | 0x0 | 16 | 0x00100000 | — | — |
| SDRAM | CS1 | 0x00100000 | 32 | 0x00800000 | — | — |
| UART | CS2 | 0x00900000 | 16 | 0x00000400 | 3 | 0 |

| | | | | | | |
|------------|-----|------------|-----|------------|-----|-----|
| 以太网控制 器 | CS3 | 0x00A00000 | 16 | 0x00000400 | 9 | 1 |
| ... | ... | ... | ... | ... | ... | ... |

上述表格对驱动开发的意义重大，在大多数情况下，硬件工程师已经给驱动工程师提供了这个表格。

华清远见

2.5.2 原理图的分析方法

原理图的分析方法是以 CPU 为中心向存储器和外设辐射，步骤如下。

(1) 阅读 CPU 部分，获知 CPU 的哪些片选、中断和集成的外设控制器被使用，列出这些元素 a、b、c...

CPU 引脚比较多时，芯片可能会被分成几个模块单独被画在原理图的不同页上，这时候应该把相应的部分都分析到位。

(2) 对第一步中列出的元素从原理图中对应的外设和存储器电路中分析出实际的使用情况。

硬件原理图中包含如下元素。

I 符号 (symbol)。

symbol 描述芯片的外围引脚以及引脚的信号，复杂的芯片可能被分割为几个 symbol。在 symbol 中，一般把属于同一个信号群的引脚排列在一起。图 2.23 给出了 NOR Flash AM29LV800B 和 E28F128J3A 的 symbol。

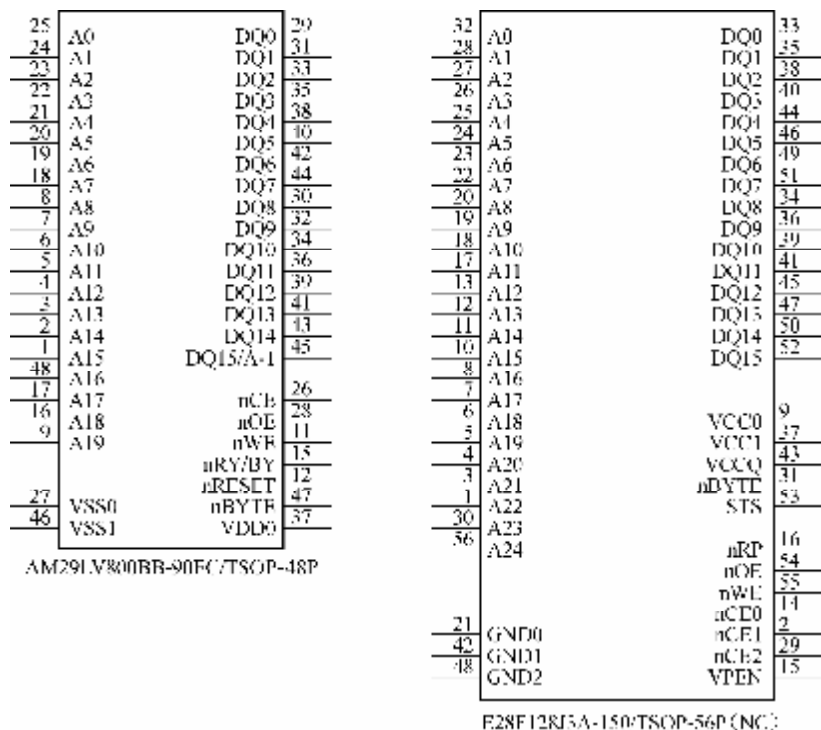


图 2.23 原理图中的 symbol

I 网络 (net)。

描述芯片、接插件和分离元器件引脚之间的互连关系，每个网络需要根据信号的定义赋予一个合适的名字，如果没有给网络取名字，EDA 软件会自动添加一个默认的网络名。添加网络后的 E28F128J3A 如图 2.24 所示。

I 描述。

原理图中会添加一些文字来辅助描述原理图（类似于源代码中的注释），如每页页脚会有该页的功能描述，对重要的信号，在原理图的相应 symbol 和 net 也会附带文字说明。图 2.25 所示为原理图中描述的例子。

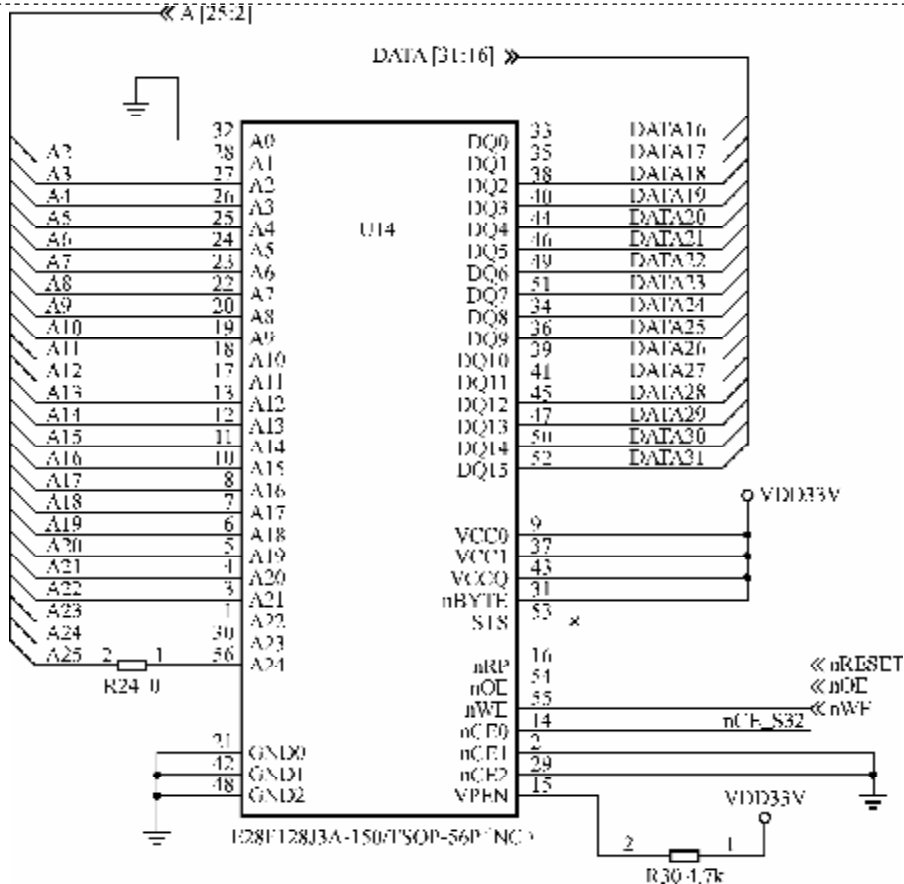


图 2.24 原理图中的 net

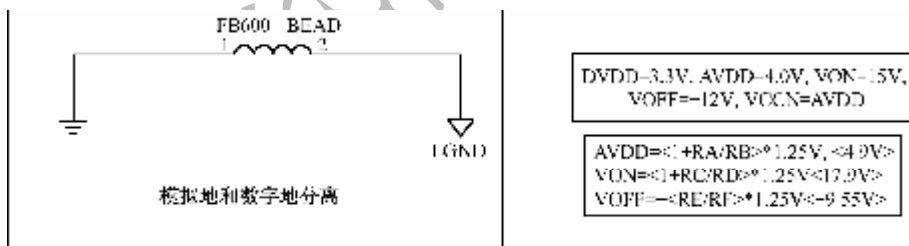


图 2.25 原理图中的描述

2.6

硬件时序分析

2.6.1 时序分析的概念

对驱动工程师或硬件工程师而言，时序分析的意思是让芯片之间的访问满足芯片手册中时序图信号有效的先后顺序、采样建立时间 (setup time) 和保持时间 (hold time) 的要求，在电路板工作不正常的时候，准确地定位时序方面的问题。

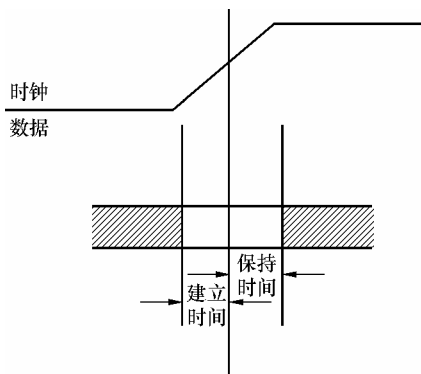


图 2.26 建立时间和保持时间

时序图编辑器及分析引擎。

2.6.2 典型硬件时序

最典型的硬件时序是 SRAM 的读写时序，在读/写过程中涉及的信号包括地址、数据、片选、读/写、字节使能和就绪/忙。对于一个 16 位、32 位（甚至 64 位）的 SRAM，字节使能表明哪些字节被读写。

图 2.27 给出了 SRAM 的一个读时序，写时序与此相似。首先是地址总线上输出要读写的地址，然后 SRAM 片选被发出，接着读/写信号被输出，之后读写信号要经历数个等待周期。当 SRAM 读写速度比较慢时，等待周期可以由 MCU 的相应寄存器设置，也可以通过设备就绪/忙（如图 2.27 中的 nWait）向 CPU 报告，这样，读写过程中会自动添加等待周期。

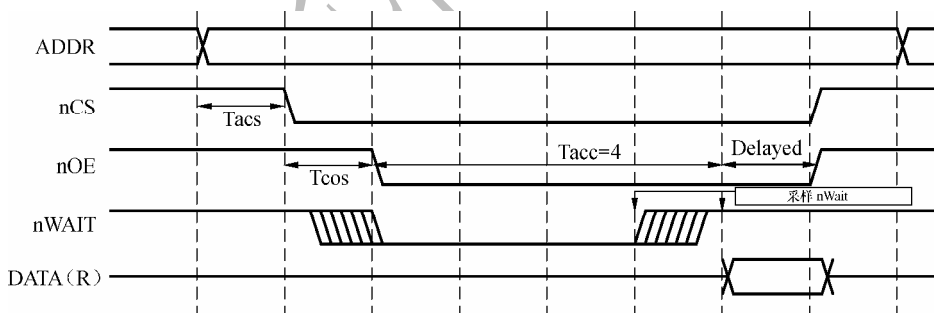


图 2.27 SRAM 读时序图

NOR Flash 和许多外设控制芯片都使用了类似 SRAM 的访问时序，因此，牢固掌握这个时序意义重大。一般，在芯片数据手册给出的时序图中，会给出图中各段时间的含义和要求，真实的电路板必须满足芯片手册上描述的建立时间和保持时间的最小要求。

2.7

仪器仪表使用

2.7.1 万用表

在电路板调试过程中我们主要使用万用表的以下两个功能：

- I 测量电平；
- I 使用二极管挡测量电路板上网络的连通性，当示波器被设置在二极管挡时，测量连通的网络会发出“滴滴”的鸣叫，否则，没有连通。

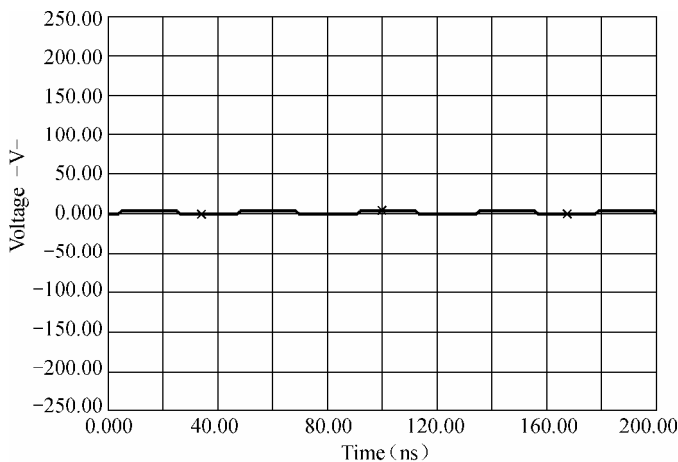
2.7.2 示波器

示波器是利用电子示波管的特性，将人眼无法直接观测的交变电信号转换成图像，显示在荧光屏上以便测量的电子仪器。它是观察数字电路实验现象、分析实验中的问题、测量实验结果必不可少的重要仪器。

使用示波器主要应注意调节垂直偏转因数选择（VOLTS / DIV）和微调、时基选择（TIME / DIV）和微调以及触发方式。

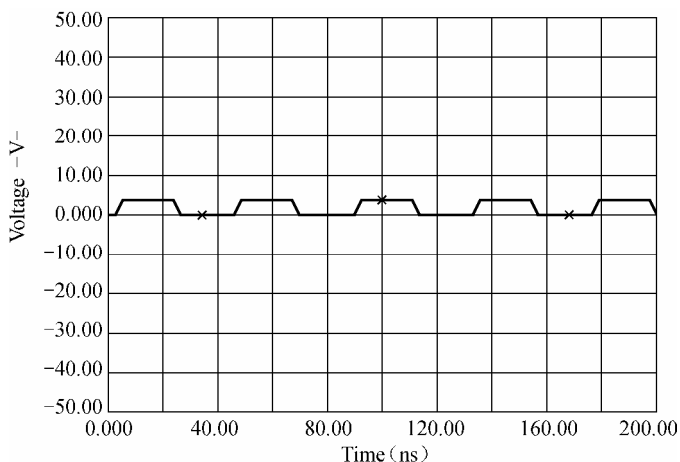
如果 VOLTS / DIV 设置不合理，则可能出现电压幅度超出整个屏幕或在屏幕上变动太过微小无法观测的现象。图 2.28 所示同一个波形在 VOLTS / DIV 设置由大到小变化过程中的示意图。

如果 TIME/DIV 设置不合适，则可能造成波形混迭。混迭意味着屏幕上显示的波形频率低于信号的实际频率。这时候，可以通过慢慢改变 TIME/DIV 到较快的时基挡，如果波形的频率参数急剧改变或者晃动的波形在某个较快的时基挡稳定下来，说明之前发生了波形混迭。根据奈奎斯特定理，采样速率至少高于信号高频成分的两倍才不会发生混迭，如一个 500MHz 的信号，至少需要 1GS/s 的采样速率。图 2.29 所示同一个波形在 TIME/DIV 设置由小到大变化过程中的示意图。

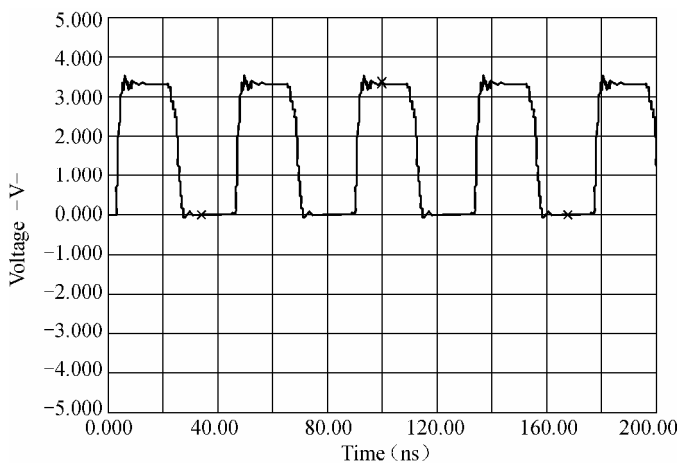


(a) 50V/DIV

图 2.28 示波器的 VOLTS / DIV 设置与波形



(b) 10V/DIV



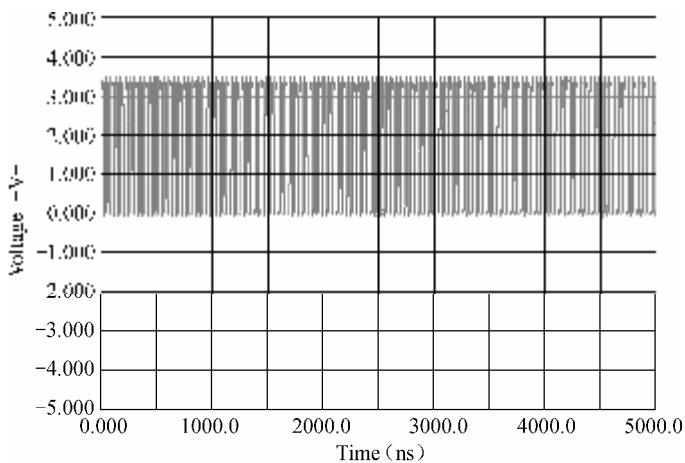
(c) 1V/DIV

图 2.28 示波器的 VOLTS / DIV 设置与波形

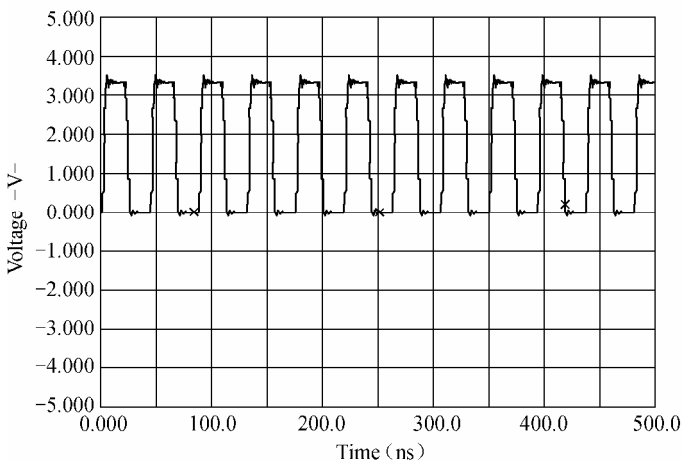


奈奎斯特定理即为采样定理，指当采样频率 $f_{s_{\max}}$ 大于信号中最高频率 f_{\max} 的两倍时，即 $f_{s_{\max}} > 2f_{\max}$ 时，采样之后的数字信号可完整地保留原始信息。

示波器的触发能使信号在正确的位置点同步水平扫描，使信号特性清晰。触发控制按钮可以稳定重复的波形并捕获单次波形。大多数用示波器的用户只采用边沿触发方式，如果拥有其他触发能力在某些应用上是非常有用的，特别是对新设计产品的故障查寻，先进的触发方式可将所关心的事件分离出来，找出用户所关心的非正常问题，从而最有效地利用采样速率和存储深度。触发能力的提高可以较大提高测试过程的灵活性。



(a) 500ns/DIV



(b) 50ns/DIV

图 2.29 示波器的 TIME/DIV 设置与波形

2.7.3 逻辑分析仪

逻辑分析仪是利用时钟从测试设备上采集数字信号并进行显示的仪器，其最主要的作用是用于时序的判定。与示波器不同，逻辑分析仪并不具备许多电压等级，通常只显示两个电压（逻辑 1 和 0）。在设定了参考电压之后，逻辑分析仪对待测试信号通过比较器来进行判定，高于参考电压者为 High，低于参考电压者为 Low。

例如，如果以 n MHz 采样率测量一个信号，逻辑分析仪会以 $1000/n$ ns 为周期采样信号，当参考电压设定为 1.5V 时，超过 1.5V 则判定为 1，低于 1.5V 则为 0，将逻辑 1 和 0 连接成连续的波形，工程师依据此连续波形可寻找时序问题。

高端的逻辑分析仪会安装有 Windows XP 操作系统并提供非常友善的逻辑分析应用软件，在其中可方便地编辑探针、信号并查看波形，如图 2.30 所示。

逻辑分析仪的波形可以显示地址、数据、控制信号及任意外部探针信号的变化轨迹，在使用之前应先编辑每个探针的信号名。

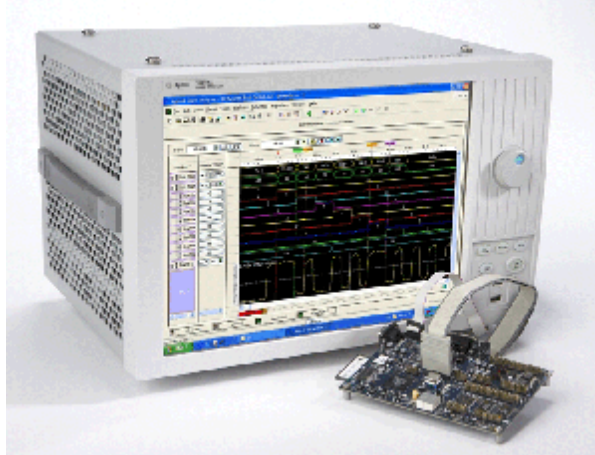


图 2.30 逻辑分析仪及配套软件

逻辑分析仪具有超强的逻辑跟踪分析功能，它可以捕获并记录嵌入式处理器的总线周期，也可以捕获如实时跟踪用的 ETM 接口的程序执行信息，并对这些记录进行分析、译码并还原出应用程序的执行过程。因此，可使用逻辑分析仪通过触发接口与 ICD（在线调试器）协调工作以补充 ICD 在跟踪功能方面的不足。逻辑分析仪与 ICD 协作可为工程师提供断点、触发和跟踪调试手段，如图 2.31 所示。



ICD 是一个容易与 ICE（在线仿真器）混淆的概念，ICE 本身需要完全仿真 CPU 的行为，可以从物理上完全替代 CPU，而 ICD 则只是与芯片内部提供的嵌入式 ICE 单元通过 JTAG 等接口互通。因此，对 ICD 的硬件性能要求远低于 ICE。

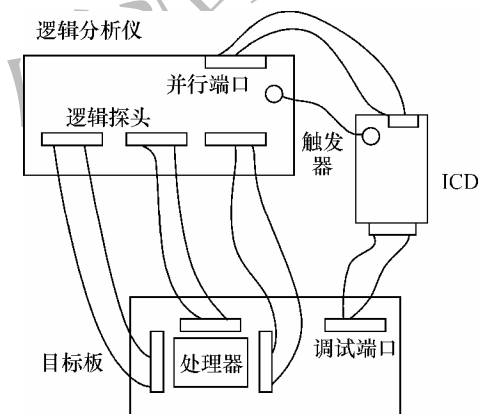


图 2.31 逻辑分析仪与 ICD 协作

2.8

总结

本章简单地讲解了驱动工程师必备的硬件基础知识，包括处理器、存储器的分类、

原理与用途，并分析了常见的外围设备接口与总线的工作方式。

此外，本章还讲解了原理图、硬件时序分析方法以及万用表、示波器和逻辑分析仪的使用方法。

推荐课程：嵌入式学院-嵌入式 Linux 长期就业班

- 招生简章：<http://www.embedu.org/courses/index.htm>
- 课程内容：<http://www.embedu.org/courses/course1.htm>
- 项目实战：<http://www.embedu.org/courses/project.htm>
- 出版教材：<http://www.embedu.org/courses/course3.htm>
- 实验设备：<http://www.embedu.org/courses/course5.htm>

推荐课程：华清远见-嵌入式 Linux 短期高端培训班

- 嵌入式 Linux 应用开发班：
<http://www.farsight.com.cn/courses/TS-LinuxApp.htm>
- 嵌入式 Linux 系统开发班：
<http://www.farsight.com.cn/courses/TS-LinuxEMB.htm>
- 嵌入式 Linux 驱动开发班：
<http://www.farsight.com.cn/courses/TS-LinuxDriver.htm>