《嵌入式系统原理与应用技术》

袁志勇 王景存
章登义 刘树波

北京: 北京航空航天大学出版社, 2009.11

PPT教学课件

第十章 嵌入式网络接口技术

9.1 网络接口技术概述

9.2 IIC接口

9.3 CAN总线接口技术

9.4 以太网接口技术

9.3 CAN总线接口

- 9.3.1 CAN总线
  - 1.CAN总线概述

控制器局域网CAN (Controller Area Net)是一种现场总线(Field Bus)，能有效地支持分布式控制和实时控制的串行通信网络。 1993年11月，ISO正式颁布了控制器局域网CAN国际标准(IS011898)。 CAN最初是由德国BOSCH公司(中文网站http://www.bosch.com.cn)于1983年为汽车监测和控制应用而开发的，目前CAN已逐步应用到其它工业控制中，现已成为ISO-11898、ISO11519-2国际标准。

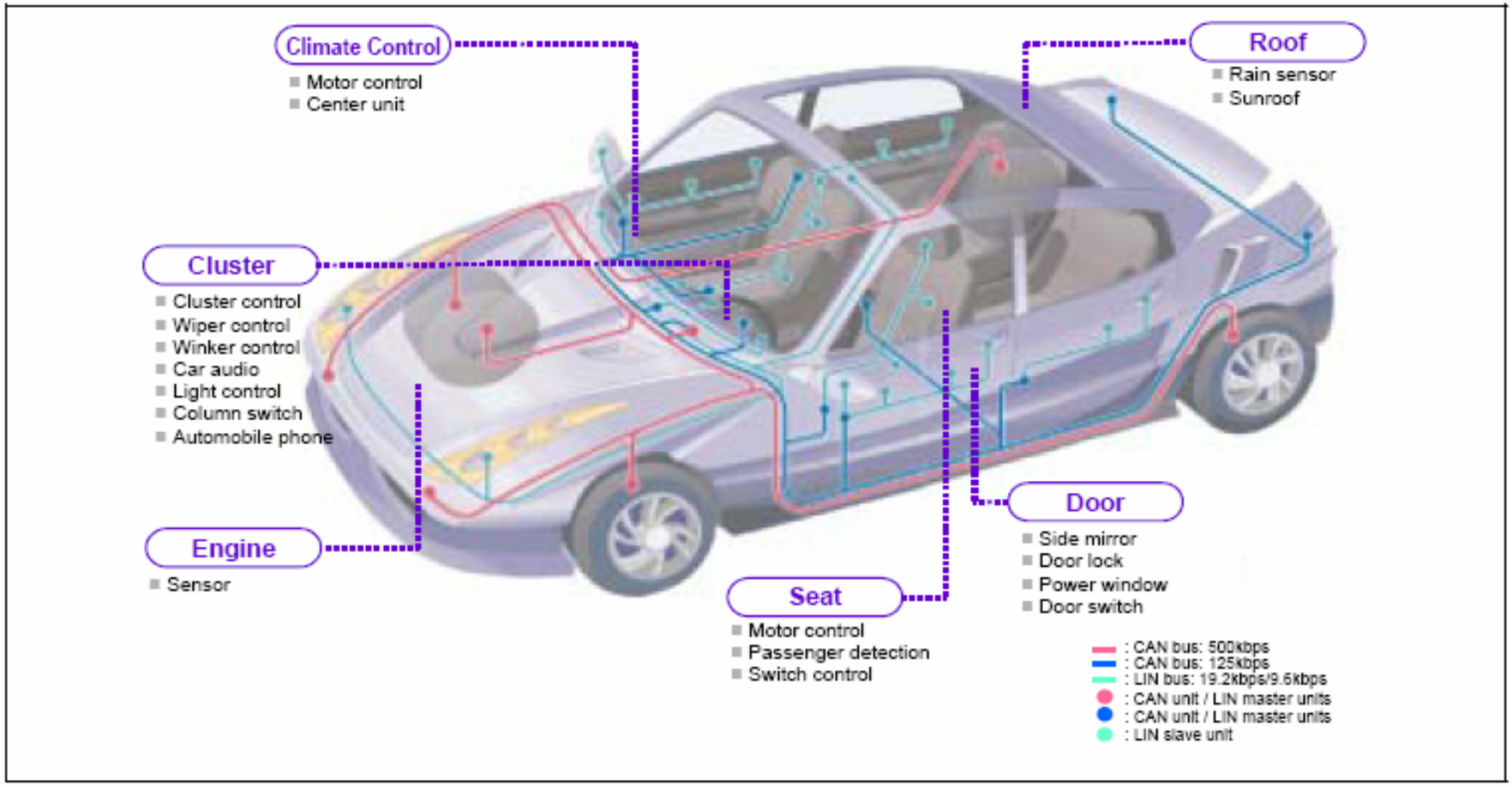


图 CAN总线在汽车电子中的应用

- 一个理想的由CAN总线构成的单一网络中可以挂接任意多个节点，实际应用中节点数目受网络硬件的电气特性所限制。例如：当使用Philips P82C250作为CAN收发器时，同一网络中允许挂接110个节点。CAN可提供1Mbps的数据传输速率。CAN总线是一种多主方式的串行通信总线。基本设计规范要求有高的位速率，高抗电磁干扰性，并可以检测出产生的任何错误。当信号传输距离达到10Km时CAN总线仍可提供高达5Kbps的数据传输速率。CAN总线具有很高的实时性能，已经在汽车工业、航空工业、工业控制、安全防护等领域中得到了广泛应用。

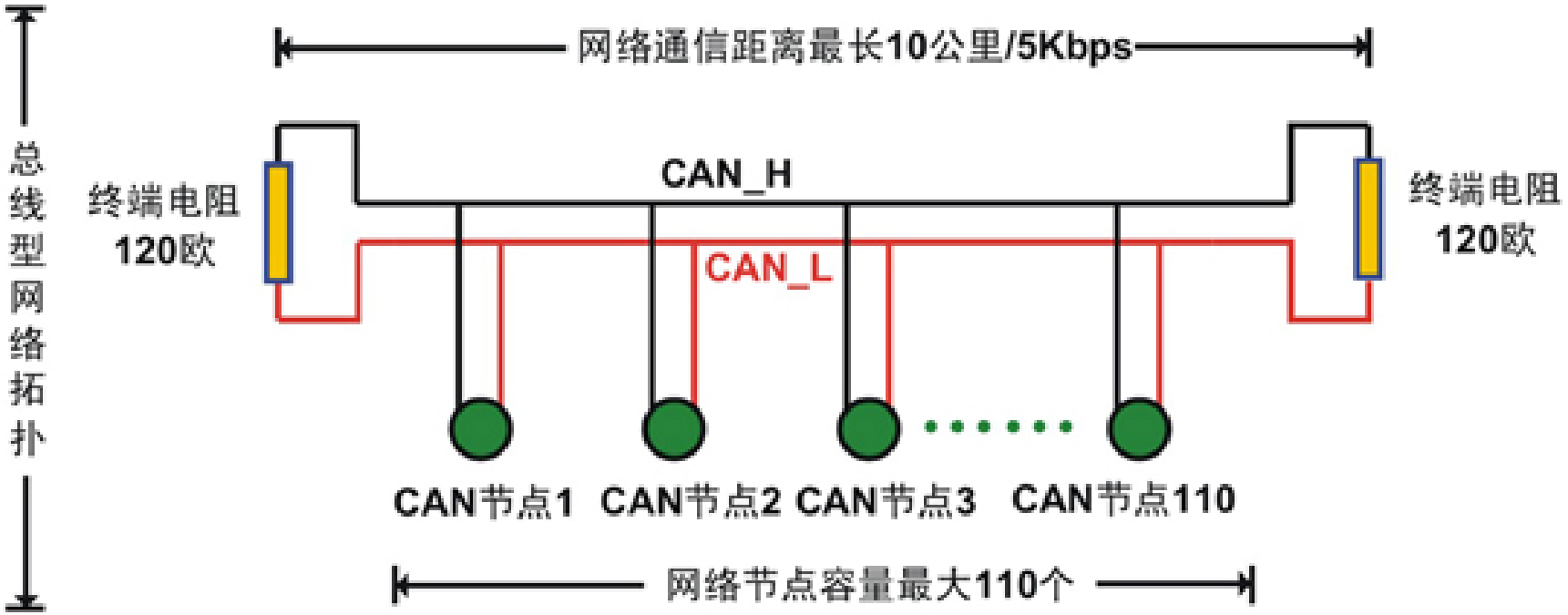


图9.12 标准CAN总线网络示意图

- CAN总线的通信介质可采用双绞线、同轴电缆和光导纤维，最常用的是双绞线。通信距离与波特率有关，最大通信距离可达10km，最大通信波特率可达1Mbps。CAN总线仲裁采用11位标识和非破坏性位仲裁总线结构机制，可以确定数据块的优先级，保证在网络节点冲突时最高优先级节点不需要冲突等待。CAN总线采用了多主竞争式总线结构，具有多主站运行和分散仲裁的串行总线以及广播通信的特点。CAN总线上任意节点可在任意时刻主动向网络上其他节点发送信息而不分主次，因此可在各节点之间实现自由通信。

- CAN总线信号使用差分电压传送，两条信号线被称为CAN_H和CAN_L，静态时均是2.5V左右，此时状态表示为逻辑1，也可以叫做“隐性”。采用CAN_H比CAN_L高表示逻辑0，称为“显性”，通常电压值为CAN_H=3.5V和CAN_L=1.5V。当“显性”位和“隐性”位同时发送的时候，最后总线数值将呈现为“显性”。

- CAN总线的一个位时间可以分成四个部分：同步段，传播时间段，相位(缓冲)段1和相位(缓冲)段2。每段的时间份额的数目都是可以通过CAN总线控制器(如MCP2510)编程控制，而时间份额的大小tq由系统时钟tsys和波特率预分频值BRP决定：

  tq=BRP/tsys。

  下图说明了CAN总线的一个位时间的各个组成部分。

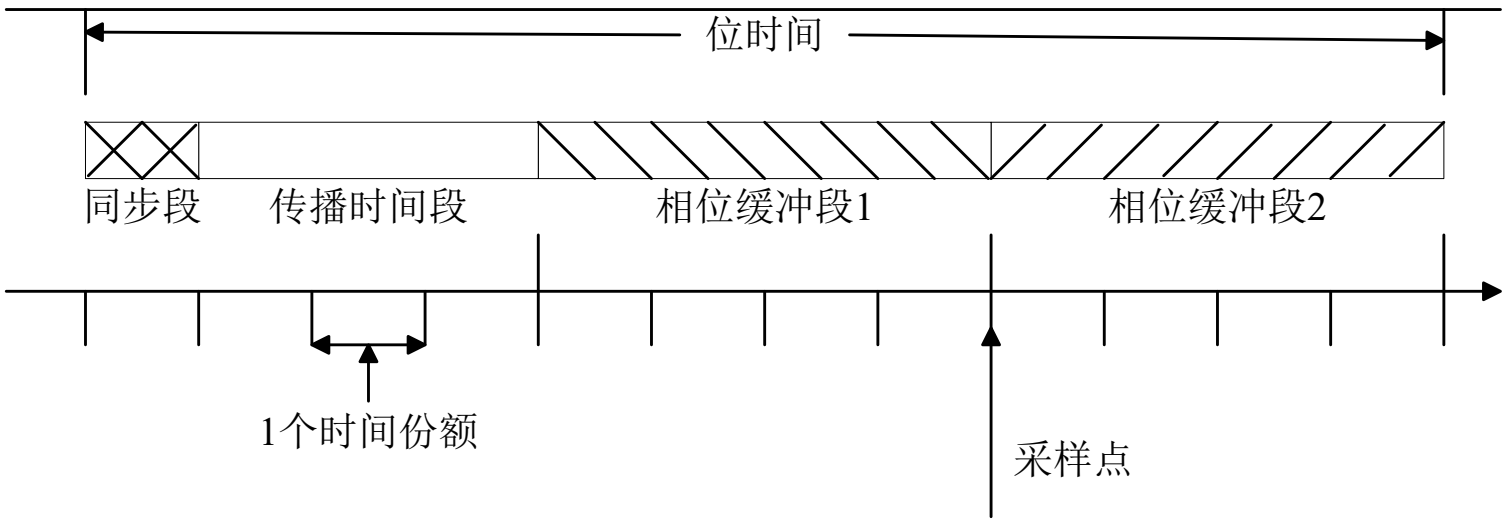


图9.13 CAN总线的位时间

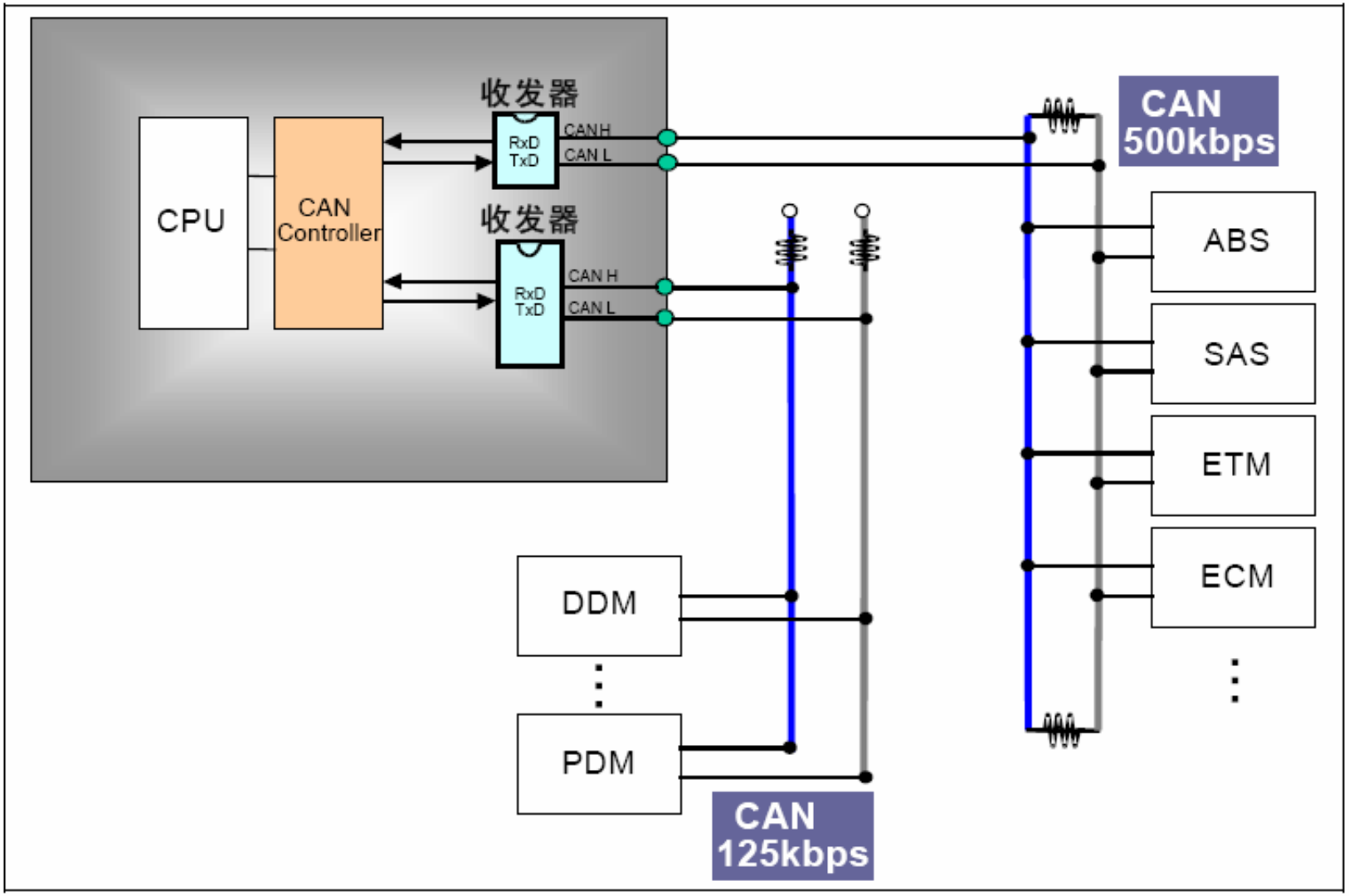
- **同步段**：用于同步总线上的各个节点，在此段内期望有一个跳变沿出现(其长度固定)。如果跳变沿出现在同步段之外，那么沿与同步段之间的长度叫做沿相位误差。**采样点**位于相位缓冲段1的末尾和相位缓冲段2开始处。

- **传播时间段**：用于补偿总线上信号传播时间和电子控制设备内部的延迟时间。因此，要实现与位流发送节点的同步，接收节点必须移相。CAN总线非破坏性仲裁规定，发送位流的总线节点必须能够收到同步于位流的CAN总线节点发送的显性位。
- **相位(缓冲)段1**：重同步时可以暂时延长。
- **相位(缓冲)段2**：重同步时可以暂时缩短。
- 同步跳转宽度：长度小于相位(缓冲)段。

同步段，传播时间段，相位缓冲段1和相位缓冲段2的设定和CAN总线的同步、仲裁等信息有关。其主要思想是要求各个节点在一定误差范围内保持同步。必须考虑各个节点时钟(振荡器)的误差和总线的长度带来的延迟(通常每米延迟为5.5ns)。正确设置CAN总线各个时间段，是保证CAN总线良好工作的关键(具体应用时，可参考CAN总线方面的相关资料)。

2.CAN总线协议

- CAN 控制器根据两根线上的电位差来判断总线电平。总线电平分为显性电平和隐性电平，二者必居其一。发送方通过使总线电平发生变化，将消息发送给接收方。


收发器
CPU
CAN Controller
RxD
TxD
CANH
CAN L
收发器
RxD
TxD
CAN H
CAN L
CAN 500kbps
ABS
SAS
ETM
ECM
DDM
PDM
CAN 125kbps

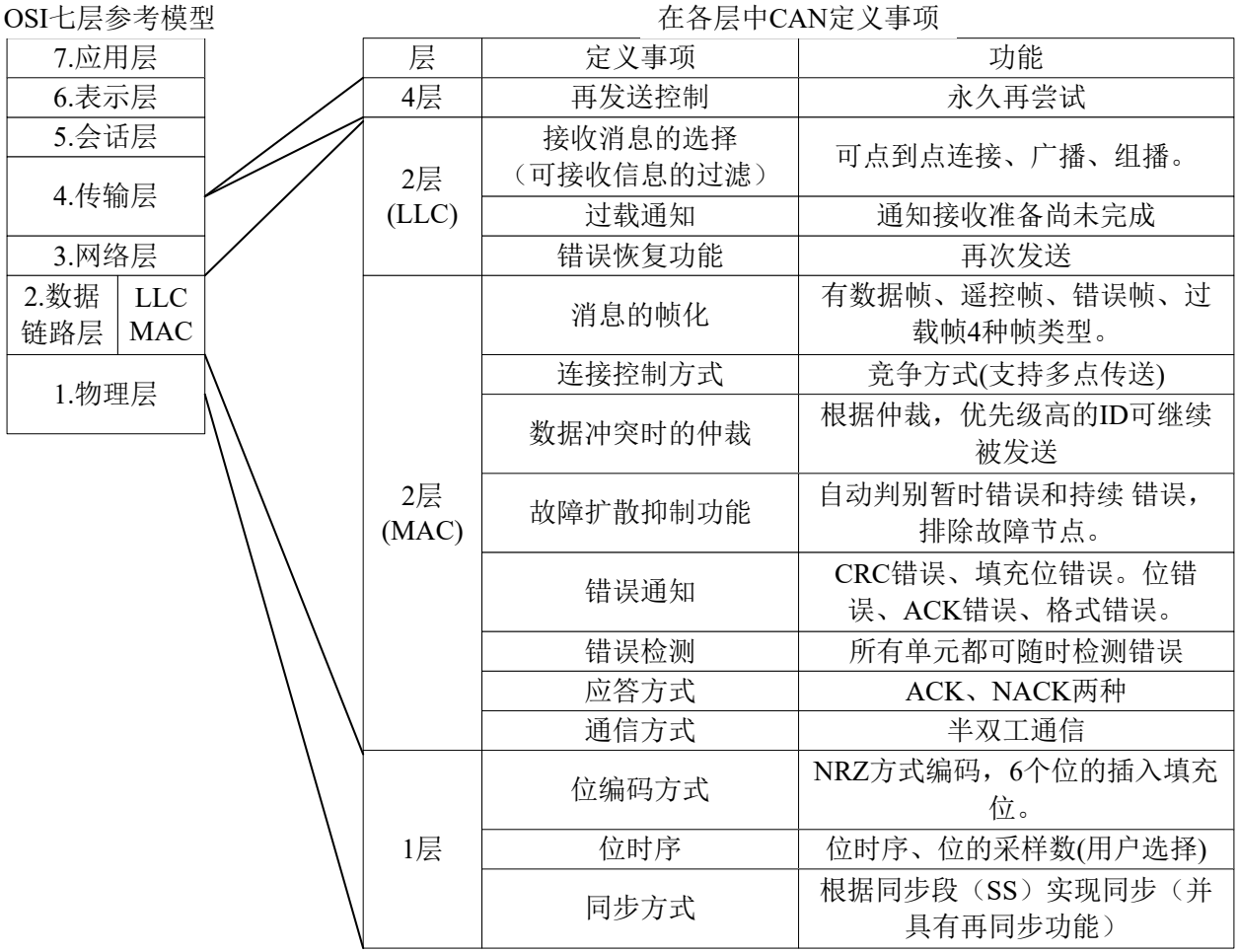
CAN连接示意图

- CAN总线与基于RS-485 构建的分布式控制系统比较，主要有以下优点：

  1).CAN控制器工作于多主方式，网络中的各节点都可根据总线访问优先权(取决于报文标识符)采用无损结构的逐位仲裁的方式竞争向总线发送数据，且CAN协议废除了站地址编码，而代之以对通信数据进行编码，这可使不同的节点同时接收到相同的数据，这些特点使得CAN总线构成的网络各节点之间的数据通信实时性强，并且容易构成冗余结构，提高系统的可靠性和系统的灵活性。而利用RS-485只能构成主从式结构系统，通信方式也只能以主站轮询的方式进行，系统的实时性、可靠性较差。

2). CAN总线通过CAN控制器接口芯片P82C250的两个输出端CANH和CANL与物理总线相连，而CANH端的状态只能是高电平或悬浮状态，CANL端只能是低电平或悬浮状态。这就保证不会出现象在RS-485网络中，当系统有错误，出现多节点同时向总线发送数据时，导致总线呈现短路，从而损坏某些节点的现象。而且CAN节点在错误严重的情况下具有自动关闭输出功能，以使总线上其他节点的操作不受影响，从而保证不会出现象在网络中，因个别节点出现问题，使得总线处于“死锁”状态。

3). CAN具有的完善的通信协议可由CAN控制器芯片及其接口芯片来实现，从而大大降低系统开发难度，缩短了开发周期，这些是RS-485所无法比拟的。



在各层中CAN定义事项

层	定义事项	功能
4层	再发送控制	永久再尝试
2层 (LLC)	接收消息的选择（可接收信息的过滤）	可点到点连接、广播、组播。
	过载通知	通知接收准备尚未完成
	错误恢复功能	再次发送
2层 (MAC)	消息的帧化	有数据帧、遥控帧、错误帧、过载帧4种帧类型。
	连接控制方式	竞争方式(支持多点传送)
	数据冲突时的仲裁	根据仲裁，优先级高的ID可继续被发送
	故障扩散抑制功能	自动判别暂时错误和持续 错误，排除故障节点。
	错误通知	CRC错误、填充位错误。位错误、ACK错误、格式错误。
	错误检测	所有单元都可随时检测错误
	应答方式	ACK、NACK两种
	通信方式	半双工通信
1层	位编码方式	NRZ方式编码，6个位的插入填充位。
	位时序	位时序、位的采样数(用户选择)
	同步方式	根据同步段（SS）实现同步（并具有再同步功能）

图9.14 OSI基本参考模型和CAN协议

CAN协议遵循OSI七层参考模型，但进行了优化，其体系结构采用三层结构，即物理层、数据链路层和传输层。

CAN协议的帧

- 通信是通过以下5 种类型的帧进行的：
  - 数据帧 、遥控帧、错误帧、过载帧、帧间隔
  - 各种帧用途如下表
  - 各种帧格式参见CAN协议应用手册（此处略）

帧	帧用途
数据帧	用于发送单元向接收单元传送数据的帧
遥控帧	用于接收单元向具有相同ID 的发送单元请求数据的帧
错误帧	用于当检测出错误时向其它单元通知错误的帧
过载帧	用于接收单元通知其尚未做好接收准备的帧
帧间隔	用于将数据帧及遥控帧与前面的帧分离开来的帧

9.3.2 CAN接口

1. MCP2510 CAN通信接口原理

大多数嵌入式处理器都有SPI总线控制器，MCP2510是一种带SPI接口的CAN总线控制器，它支持CAN技术规范V2.0 A/B版本，可以3V~5.5V供电，能够直接和3.3V I/O口的嵌入式处理器连接，电路原理如下图所示。

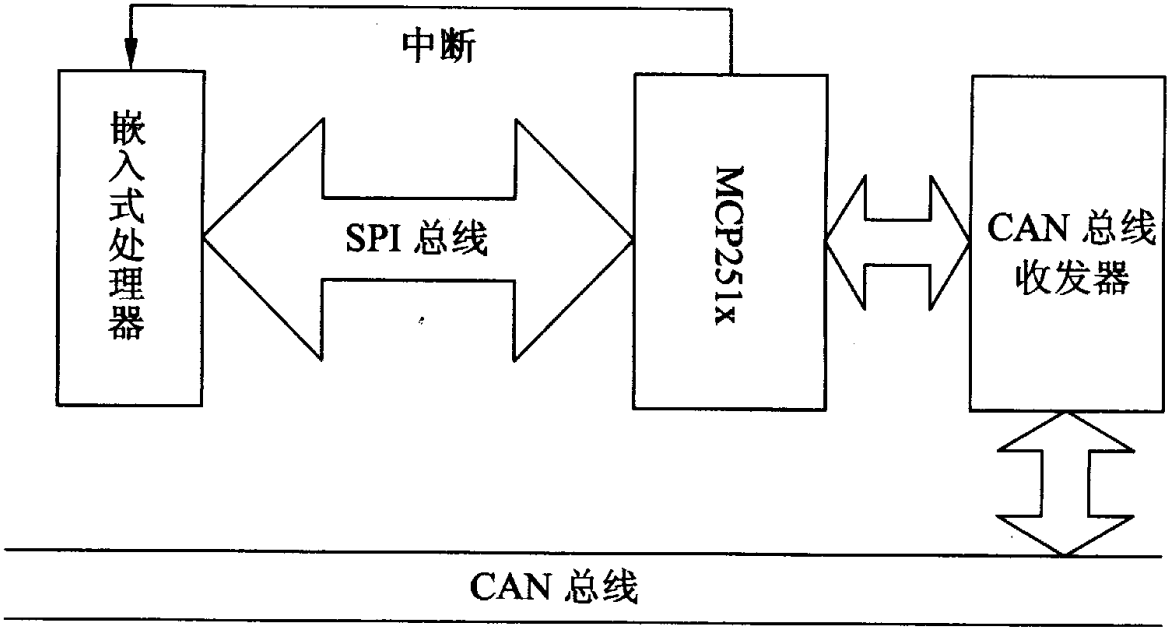


图9.15 MCP251x组成的嵌入式CAN节点

2. S3C2410 SPI接口信号

S3C2410有两个串行外围设备接口SPI(Serial Peripheral Interface)，SPI接口信号如下：

(1) SPICLK[1:0] (I)

SPI时钟信号

(2) SPIMISO[1:0] (I/O)

SPI为主机时输入数据，为从机时输出数据。

(3) SPIMOSI[1:0] (I/O)

SPI为主机时输出数据，为从机时输入数据。

(4) nSS[1:0] (I)

SPI为从机时充当片选信号。

3. CAN总线接口示例

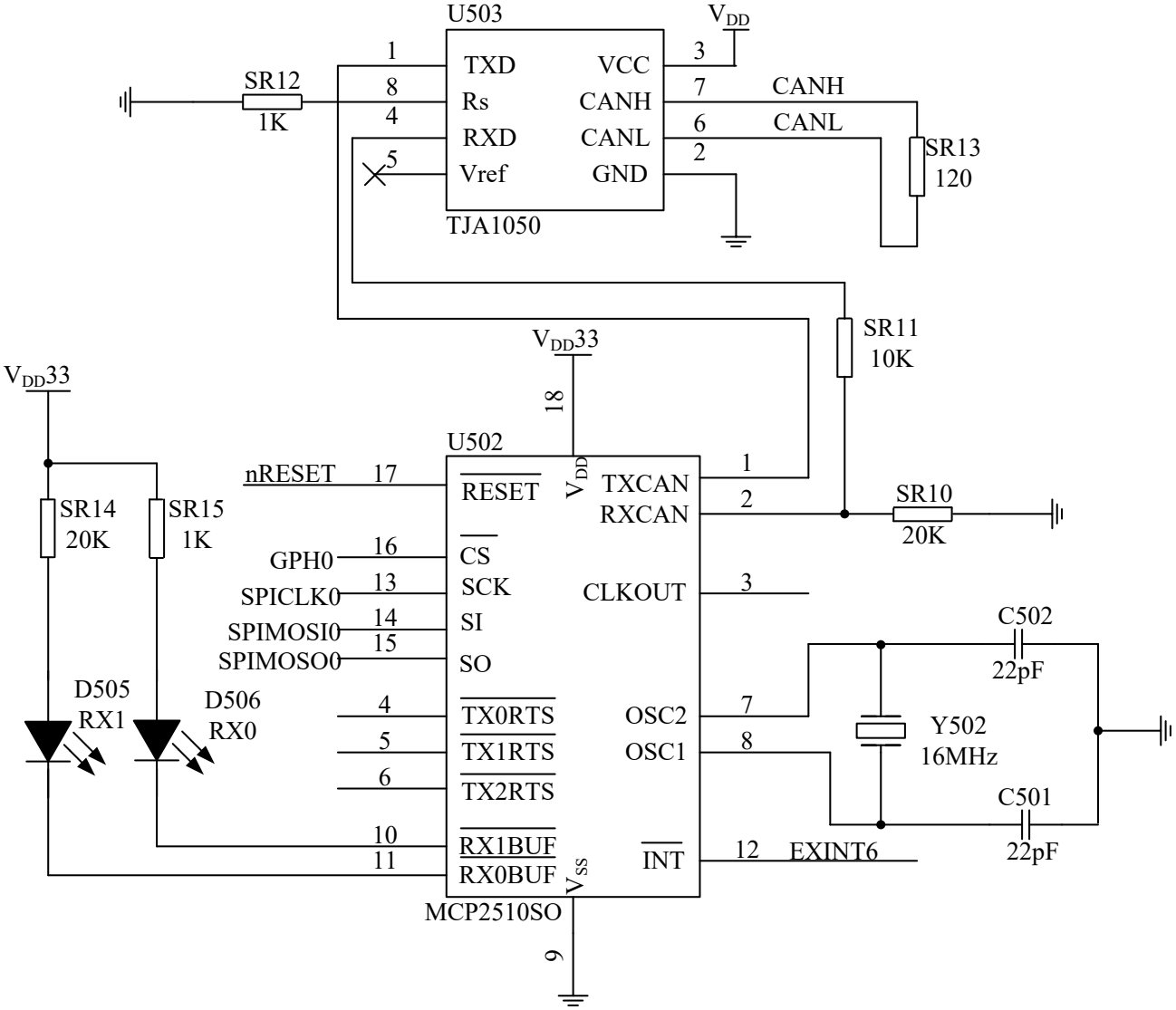


图9.16 MCP2510组成的CAN总线接口

在上面接口电路中，MCP2510使用3.3V电压供电，它可以直接和S3C2410通过SPI总线连接。相关的资源如下：

(1)23C2410的GPH0口线用作控制MCP2510的片选信号，低电平有效。

(2)用S3C2410的外部中断6（EXINT6 ）作为中断引脚，低电平有效。

(3)16MHz晶体作为输入时钟，MCP2510内部有振荡电路，用晶体可以直接起振。

(4) MCP2510内含3个发送缓冲器，2个接收缓冲器。

(5) 使用TJA1050(与Philips P82C250芯片兼容)作为CAN总线收发器。

(6)CAN总线收发器TJA1050必须使用5V供电。但MCP2510和TJA1050连接的两个信号都是单向的信号。对于MCP2510，TXCAN是输出信号，RXCAN是输入信号。

(7)TJA1050为5V供电时，输入高电平Vih的范围是2～5.3V。而3.3V供电的MCP2510输出TXCAN信号高电平Voh最小值为2.6V，可以满足要求。

CAN接口软件实现方案

S3C2410的SPI可以工作在四种模式，但是MCP2510的SPI接口只支持其中的两种。因此应该将S3C2410的SPI接口配置为MCP2510支持的模式工作。下面以S3C2410处理器SPI配置为正常模式的中断方式为例进行简要说明：

(1) MCP2510负责完成总线通信协议的物理层和数据链路层功能，通信控制程序只需在发送/接收模块中写入相应的配置参数即可；

(2) 发送时，只需将发送的内容写入发送缓冲区并触发一次发送请求，即完成一次发送；

(3) 总线接收器收到数据时，启动一次2410芯片的外部中断，然后通过SPI串口访问MCP2510的寄存器进行读操作。

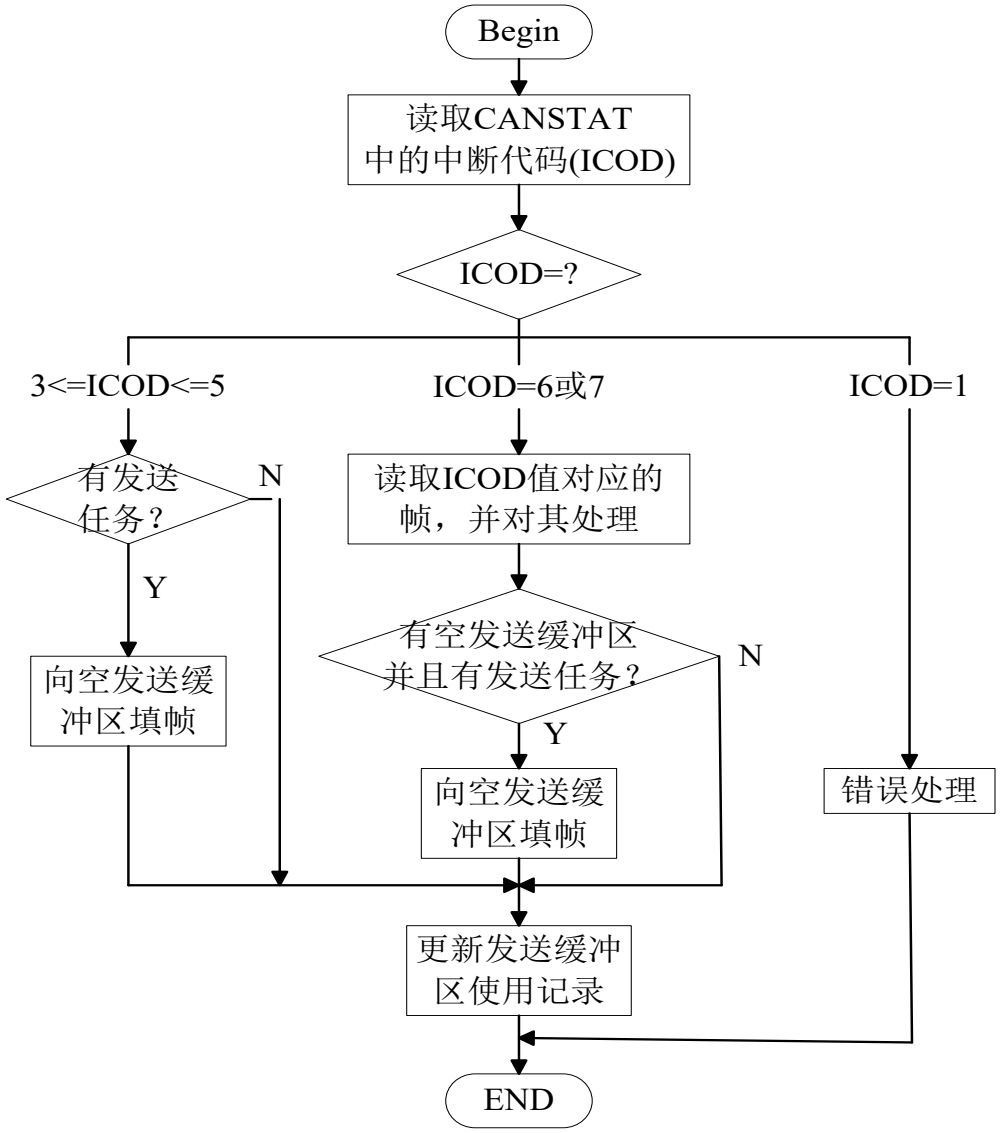


图9.17　中断方式实现CAN总线数据发送/接收流程图

(1)发送处理

- 在两种情况会下被执行，其一发送中断响应中，如果当前有发送任务，通过发送缓冲区记录标志确定空的发生缓冲区并向其中填发送帧，然后启动发送；
- 其二在接收中断响应中，在接收处理完后如果有发送任务且有发送缓冲区空，会处理发送任务。在两处处理发送主要是考虑有多个发送、接收缓冲区可用，以提高通信效率。

- (2)接收处理
  - 需要根据ICOD的值来确定当前是那个接收缓冲区的数据准备好，可以读取。接收帧处理过程中首先读取帧的ID值，确定当前帧的数据是哪个节点发送来的，并据此把帧中数据字段的数据存放相应的位置。
- (3)错误处理
  - 引起错误处理的原因很多，并且都产生错误中断，需要访问错误标志寄存器来确定具体错误类型，并据此作相应的处理；
  - 此外，更新发送缓冲区使用记录是为了在发送和接收中断响应中可以清楚地知道发送缓冲区的使用情况，确定当前是否可以做发送处理。

课堂思考与讨论

如何扩展CAN Bus网络的节点数量和通信距离?

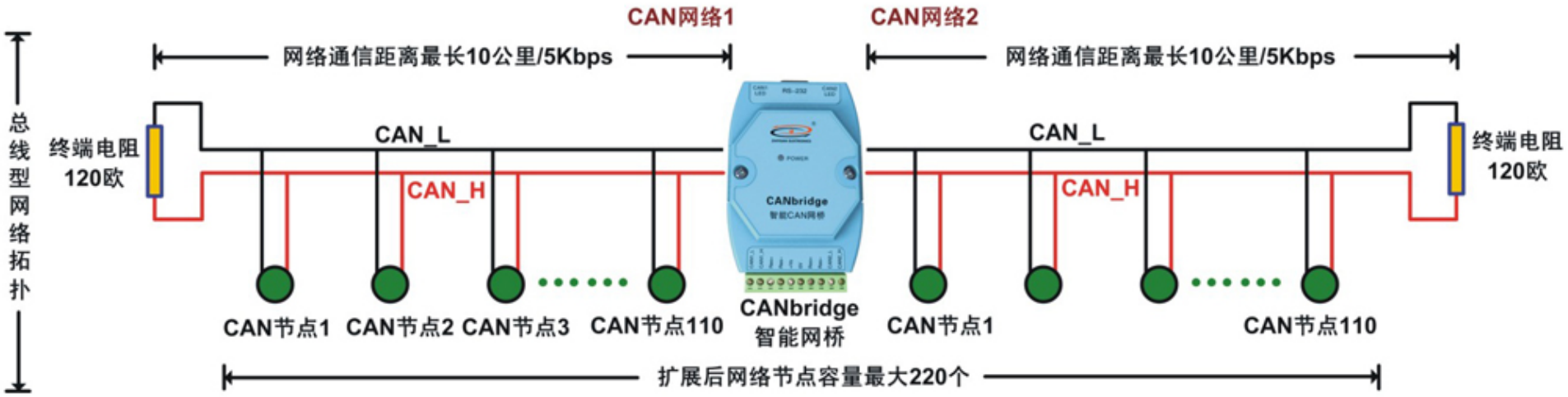
- **参考解答：**

标准CAN Bus网络在网络节点数量(CAN Bus节点数量<110个)、通信距离(网络中节点最远通信距离<10Km)、通信波特率和网络拓扑(必须为总线型)这4个方面都有明确的限制。若需要扩展节点数量和通信距离，可利用CAN网桥、CAN集线器、中继器等方法予以解决，这样就扩展CAN Bus网络的应用范围。

（1）成倍延长CAN网络的通讯距离；
（2）增加CAN网络中的节点数量；
（3）连接不同通信波特率的CAN网络；
（4）改变CAN网络布线的拓扑结构。

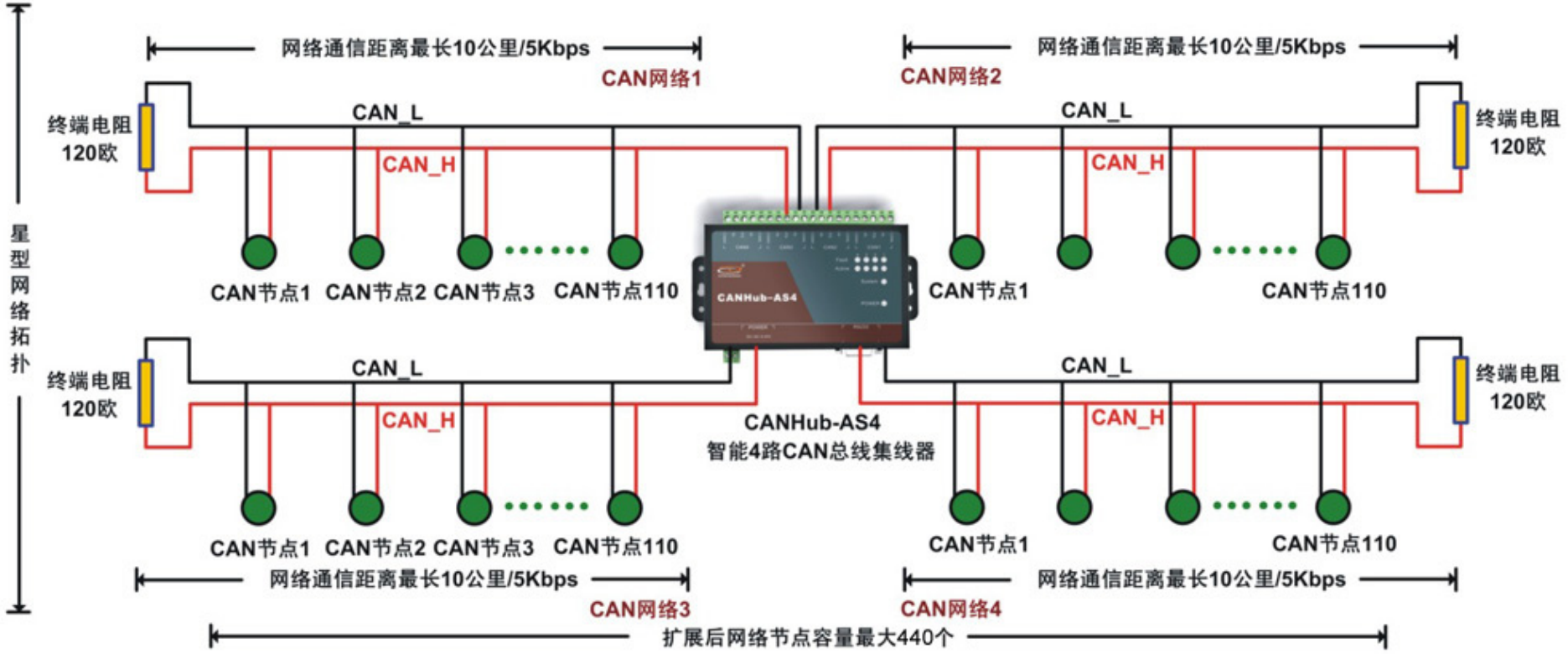
下面是两个扩展应用的案例。

案例1：应用CAN Bus网桥后的扩展设计。



如图上图所示，采用CAN Bridge智能CAN网桥后，实现CAN网络1和CAN网络2的信息传递，并允许CAN-bus网络1和CAN-bus网络2采用不同的通信波特率，使CAN网络1和CAN网络2连接起来形成一个扩展CAN Bus网络，在这个扩展CAN Bus网络中，最长通信距离可达20公里，节点数量可达220个。

案例2：应用CAN集线器后的扩展设计。



如上图所示，采用CAN Hub智能4路CAN Bus总线集线器，实现CAN网络1、CAN网络2、CAN网络3和CAN网络4的信息传递，并允许CAN网络1、CAN网络2、CAN网络3和CAN网络4采用不同的通信波特率，使CAN网络1、CAN网络2、CAN网络3和CAN网络4连接起来形成一个扩展CAN-bus网络，在这个扩展CAN-bus网络中，采用星型网络拓扑，两节点间最长通信距离可达20公里，节点数量可达440个。

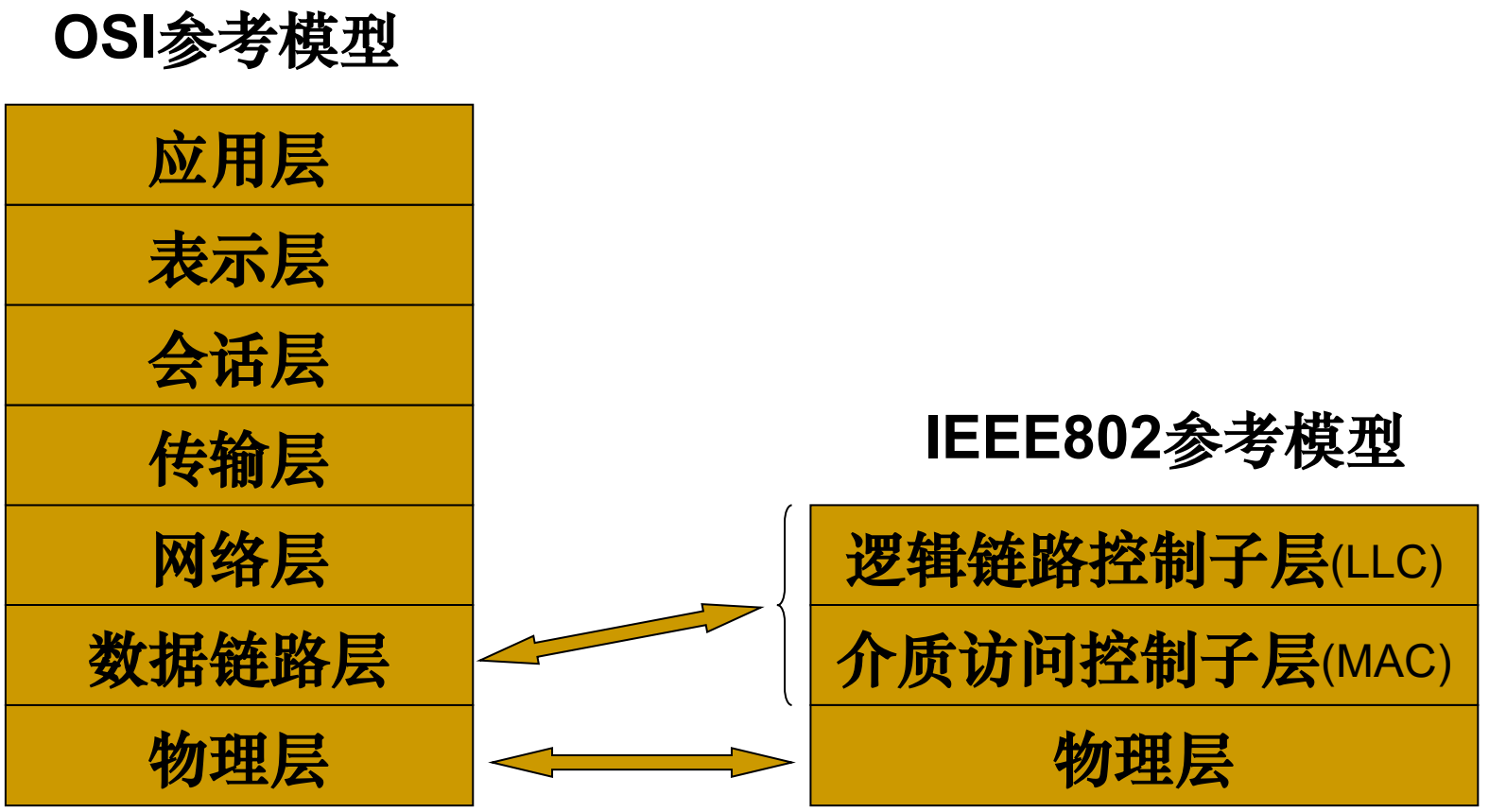
9.4 以太网接口

9.4.1 嵌入式以太网基础知识

- 可以利用I2C、CAN、RS–232、RS–485等总线将MCU组网，但这种网络的有效半径有限，有关的通信协议也比较少，并且一般是孤立于Internet以外
- 如果嵌入式系统能够连接到Internet上，则信息传递的范围急剧扩大
- 在Internet的众多协议中，以太网和TCP/IP协议族已经成为使用最广泛的协议，它的高速、可靠、分层，以及可扩充性使得它在各个领域的应用越来越灵活

- 嵌入式系统接入Internet的准备工作
  - 硬件上，要有一个以太网接口电路
  - 软件上，要能够提供相应的通信协议
- 以太网(Ethernet)技术
  - 开始的标准以太网是由Xeros公司开发的一种基带局域网技术，最初使用同轴电缆作为网络传输介质，采用带冲突检测的载波侦听多路访问 (CSMA/CD) 机制，数据传输速率达10Mbps；
  - 所有的以太网都遵循IEEE 802.3 10BASE－T标准，IEEE 802.3是最早的以太网技术标准，于1980年制定；
  - 在Internet网络中，以太网是应用最广泛的数据链路层协议。现在的操作系统均能支持这种类型的协议格式；
  - 传统的标准以太网技术难以满足日益增长的网络数据流量速度需求。 1995年3月IEEE宣布了IEEE802.3u 100BASE－T快速以太网(Fast Ethernet)标准，数据传输速率达10Mbps，开始了快速以太网时代。

IEEE802标准所描述的局域网参考模型与OSI参考模型的比较如下图所示：



- CSMA/CD协议是一种随机争用型的介质访问控制方法，其工作过程简述如下：

1) 载波检测：要发送报文的工作站必须监视以辨别何时通道可用(该时刻没有其他的网络工作站在发送)；

2) 当工作站检测到通道空闲时，它就发送报文，并以接收工作站设备的地址进行标记；

3)所有空闲的工作站(没有进行传输的工作站)继续监视通道报文；

4)被标记的工作站接收报文，并返回一个应答(确认)帧；

5)当发送通道检测到冲突时，它们就停止传输，延时一个随机时间后重新发送；

6)工作站在通知用户网络太忙时，典型地重发16次。

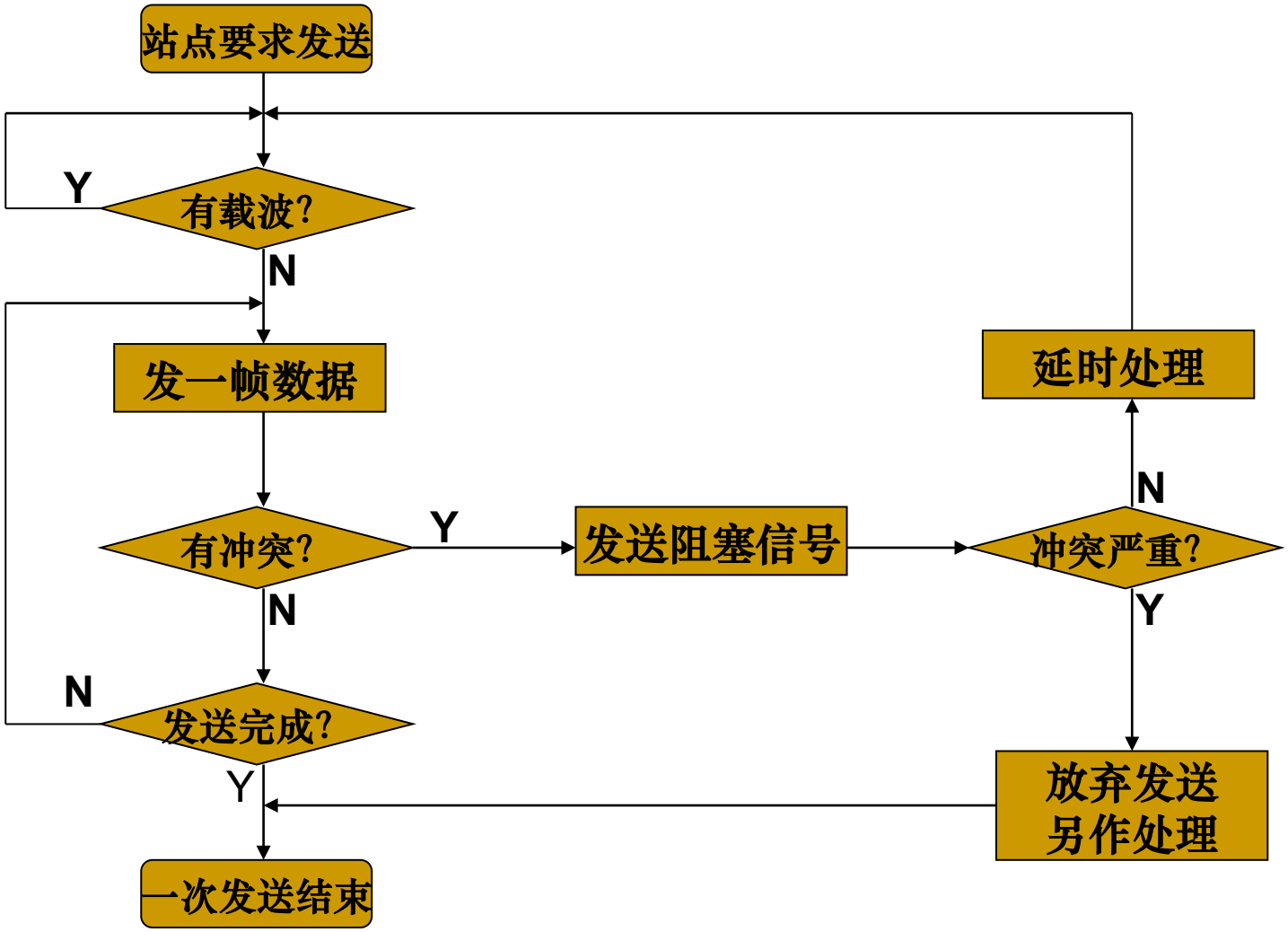


图 CSMA/CD介质访问控制流程图

- 数据帧格式
  - 以太网采用广播机制，所有与网络连接的工作站都能看到网络上传送的数据。通过查看包含在帧中的目标地址，确定是否进行接收或放弃。如果证明数据确实是发给自己的，工作站将会接收数据并传递给高层协议进行处理。
  - 在发送数据之前，工作站首先需要侦听网络是否空闲，如果网络上没有任何数据传送，工作站就会把所要发送的信息投放到网络中。否则，工作站只能等待网络下一次出现空闲的时候再进行数据的发送。

IEEE802.3规定的以太网帧格式

PR	DA	SA	TYPE	DATA	PAD	FCS
同步位 8byte	目的地址 6byte	源地址 6byte	类型 2byte	数据 46~1500byte	填充位 可选	帧校验序列 4byte

(a) 以太网MAC层帧格式

PR	SFD	DA	SA	TYPE	DATA	PAD	FCS
同步位 7byte	分隔位 1byte	目的地址 6byte	源地址 6byte	类型 2byte	数据 46~1500byte	填充位 可选	帧校验序列 4byte

(b) IEEE 802.3 MAC层帧格式

教材P227图9-19

图 数据帧格式

- 同步位/前导码PR (PReamble)：由0和1间隔代码组成，用于收发双方的时钟同步，同时也指明了传输速率(10Mbps/100Mbps) 。IEEE 802.3帧的同步位占7个字节，每个字节均为10101010。
- (开始帧)分隔符SFD (Start Frame Delimiter)： IEEE 802.3帧中的分隔定界字节，与同步位不同的是它以两个连续的代码1结尾，即代码为1010101011，表示后面跟着的是真正的数据，而不是同步时钟。
- 目的地址DA：以太网的地址为6字节，表示该帧传送给网络中的哪个工作站。目的地址可以是单播地址(单播地址: unicast address，即接收方工作站的物理地址/以太网地址)，也可以是广播地址(broadcast address)或多播地址(multicast address)。按照惯例，广播地址(全1，即地址码为0xFFFFFFFFFFFF)被预留下来用于发送到所有节点，广播地址的数据可以被任何以太网节点收到。多播地址(组播地址)提供了一种有限的广播形式，网络上节点的某个子集同意收听一给定的组播地址。

- 源地址SA：表求该帧的数据是哪个网卡发的，即发送工作站的网卡地址，占6个字节。
- 类型TYPE：存在于以太网帧中，占2个字节，表明该帧的数据是什么类型的数据(即用于指定接收数据的高层协议)。不同协议的类型字段有所不同。

  例：0x0800表示数据为IP包；0x0806表示数据为ARP包；0x814C表示数据是SNMP包；0x8137为IPX/SPX包等。
- 数据DATA：数据段长度不能超过1500字节。因为以太网规定整个传输包的最大长度不超过1514字节(14字节为DA+SA+TYPE)

- 填充位PAD(可选)：由于以太网规定帧传输的数据包最小不能小于60字节，去掉DA+SA+TYPE这14字节，还必须传输46字节的数据。若数据段的长度过小，数据段的不足46个字节的最小值时，那么相应软件将会自动填充数据段(数据后面补0000……，当然也可以补其他值)，以确保整个帧的长度不少于46个字节。
- 帧校验序列FSC (Frame Check Sequence)：该序列为4个字节长度的CRC(循环冗余校验)校验，该检验由网卡自动计算、自动生成、自动检验、自动在数据段后面填写，不需要软件进行管理；在接收工作站被重新计算，以确定数据帧在传输过程中是否被损坏。

通常，PR、SD、PAD、FCS这四个段都是由网卡(包括物理层和MAC层的处理)自动产生的，剩下的DA、SA、TYPE、DATA这4个段的内容是由上层的软件控制的。

以太网数据传输的特点

- 所有数据位的传输由低位开始，传输的位流是用**曼彻斯特编码**。
- 以太网是基于冲突检测的总线复用方法，**冲突退避算法**是**由硬件自动执行**的。
- 以太网传输的数据段的长度，DA+SA+TYPE+DATA+PAD最小为60 byte，最大为1514 byte。
- 通常的以太网卡可以接收3种地址的数据，一个是广播地址，一个是多播地址（或者叫组播地址，在嵌入式系统中很少用到），一个是它自己的地址。但有时，用于网络分析和监控，网卡也可以设置为接收任何数据包。

- 任何两个网卡的物理地址都是不一样的，是世界上唯一的，网卡地址由专门机构分配。不同厂家使用不同地址段，同一厂家的任何两个网卡的地址也是唯一的。根据网卡的地址段(网卡地址的前3个字节)可以知道网卡的生产厂家。

嵌入式系统中主要处理的以太网协议

TCP/IP是一个分层的协议，包含有应用层、传输层、网络(互连)层、数据链路层、物理层等。每一层实现一个明确的功能，对应一个或者几个传输协议。每层相对于它的下层都作为一个独立的数据包来实现。典型的分层和每层上的协议见下表所述。

表 TCP/IP协议的典型分层和协议

分　　层	每层上的协议	分　　层	每层上的协议
应用层（Application）	BSD 套接字（BSD Sockets）	数据链路层（Data Link）	IEEE802.3 Ethernet MAC
传输层（Transport）	TCP、UDP	物理层　（Physical）	
网络层（Network）	IP、ARP、ICMP、IGMP		

- **(1) ARP (Address Resolution Protocol，地址解析协议)**
  - 网络层用32位的地址来标识不同的主机(即IP地址)，而链路层使用48位的物理(MAC)地址来标识不同的以太网或令牌环网接口。只知道目的主机的IP地址并不能发送数据帧给它，必须知道目的主机网络接口的物理地址才能发送数据帧。
  - ARP的功能就是实现从IP地址到对应物理地址的转换。源主机发送一份包含目的主机IP地址的ARP请求数据帧给网上的每个主机，称作ARP广播，目的主机的ARP收到这份广播报文后，识别出这是发送端在询问它的IP地址，于是发送一个包含目的主机IP地址及对应的物理地址的ARP回答给源主机。
  - 为了加快ARP协议解析的数据，每台主机上都有一个ARP cache存放最近的IP地址到硬件地址之间的映射记录。其中每一项的生存时间(一般为20分钟)，这样当在ARP的生存时间内连续进行ARP解析的时候，不需要反复发送ARP请求了。

(2) ICMP（Internet Control Messages Protocol，网络控制报文协议）

- ICMP是IP层的附属协议，IP层用它来与其他主机或路由器交换错误报文和其他重要控制信息。ICMP报文是在IP数据包内部被传输的。在Linux或者Windows中，两个常用的网络诊断工具ping和traceroute (Windows下是Tracert)，其实就是ICMP协议。

(3) IP (Internet Protocol，网际协议)

- IP工作在网络层，是TCP/IP协议族中最为核心的协议。所有的TCP、UDP、ICMP及IGMP数据都以IP数据包格式传输 (IP封装在IP数据包中)。IP数据包最长可达65535字节，其中报头占32位。还包含各32位的源IP地址和32位的目的IP地址。
- TTL (time-to-live，生存时间字段) 指定了IP数据包的生存时间 (数据包可以经过的最多路由器数)。TTL的初始值由源主机设置，一旦经过一个处理它的路由器，它的值就减去1。当该字段的值为0时，数据包就被丢弃，并发送ICMP报文通知源主机重发。

- **IP提供不可靠、无连接的数据包传送服务，高效、灵活。**

①不可靠 (unreliable) 的意思是它不能保证IP数据包能成功地到达目的地。如果发生某种错误，IP有一个简单的错误处理算法：丢弃该数据包，然后发送ICMP消息报给信源端。任何要求的可靠性必须由上层来提供 (如TCP)。

②无连接 (connectionless) 的意思是IP并不维护任何关于后续数据包的状态信息。每个数据包的处理是相互独立的。IP数据包可以不按发送顺序接收。如果一信源向相同的信宿发送两个连续的数据包 (先是A，然后是B)，每个数据包都是独立地进行路由选择，可能选择不同的路线，因此B可能在A到达之前先到达。

- IP的路由选择：源主机 IP接收本地TCP、UDP、ICMP、IGMP的数据，生成IP数据包，如果目的主机与源主机在同一个共享网络上，那么IP数据包就直接送到目的主机上。否则就把数据包发往一默认的路由器上，由路由器来转发该数据包。最终经过数次转发到达目的主机。IP路由选择是逐跳（hop-by-hop）进行的。所有的IP路由选择只为数据包传输提供下一站路由器的IP地址。

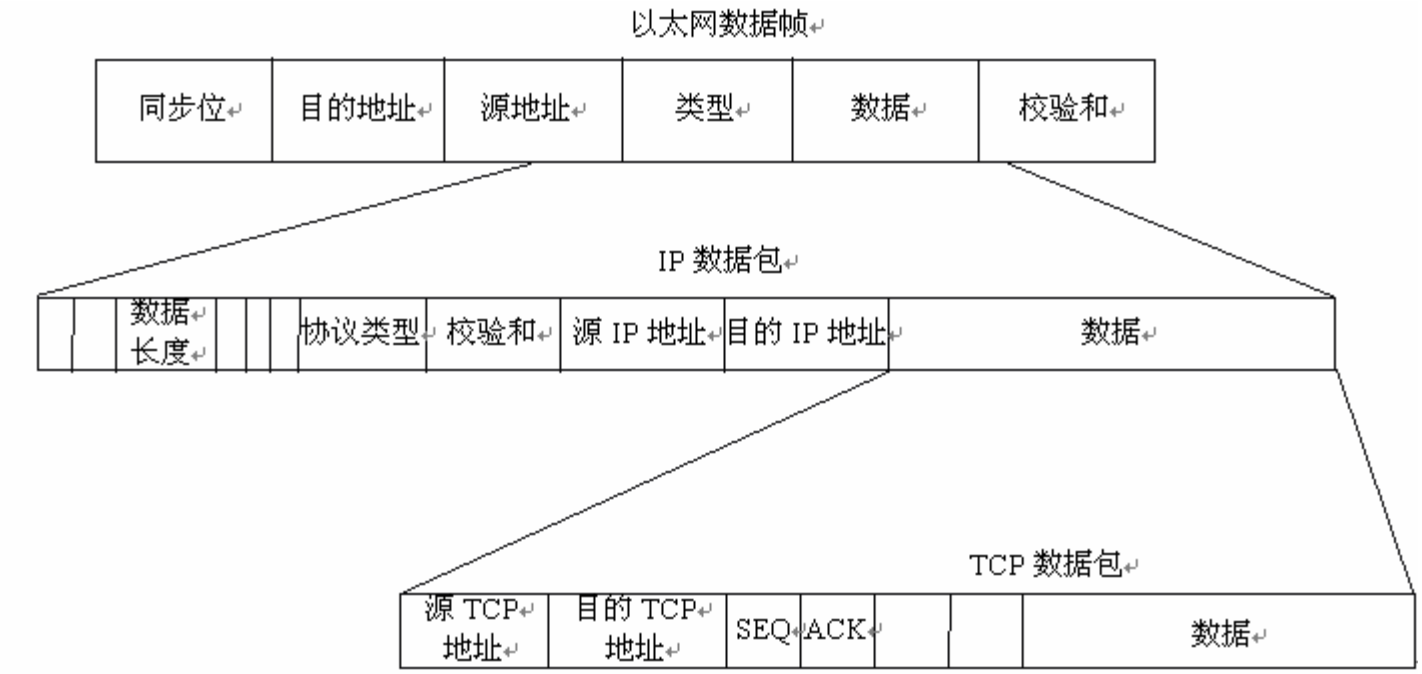


图　　网络协议示意图

(4) TCP (Transfer Control Protocol，传输控制协议)

- TCP协议是一个面向连接的可靠的传输层协议。TCP为两台主机提供高可靠性的端到端数据通信。它所做的工作包括：

① 发送方把应用程序交给它的数据分成合适的小块，并添加附加信息 (TCP头)，包括顺序号，源、目的端口，控制、纠错信息等字段，称为TCP数据包。并将TCP数据包交给下面的网络层处理。

② 接受方确认接收到的TCP数据包，重组并将数据送往高层。

- **(5) UDP (User Datagram Protocol，用户数据包协议)**
  - **UDP协议是一种无连接不可靠的传输层协议。**它只是把应用程序传来的数据加上UDP头（包括端口号，段长等字段），作为UDP数据包发送出去，但是并不保证它们能到达目的地。可靠性由应用层来提供。
  - **因为协议开销少，和TCP协议相比，UDP更适用于应用在低端的嵌入式领域中。**很多场合如网络管理SNMP，域名解析DNS，简单文件传输协议TFTP，大都使用UDP协议。

(6) IP地址和端口号

- **IP地址：**(IPv4:32位)标识计算机等网络设备的网络地址，由4个8bits组成，中间以小数点分隔，由网络标识(network ID)和主机标识(host ID)组成，分为A、B、C、D、E五类。如：166.111.136.3，166.111.52.80。为解决IP地址耗尽问题，提出了IPv6(128位)。
- **端口号：**网络通信时同一机器上的不同进程的标识。如:80（HTTP），21(FTP)，23(Telnet)，25(SMTP)，其中1~1024为系统保留的端口号。

TCP/IP Socket编程

- 通过TCP协议传输，得到的是一个顺序的无差错的数据流。发送方和接收方的两个成对的socket之间必须建立连接，以便在TCP协议的基础上进行通信，当一个socket(通常都是server socket)等待建立连接时，另一个socket可以要求进行连接，一旦这两个socket连接起来，它们就可以进行双向数据传输，双方都可以进行发送或接收操作。

UDP Socket编程

- UDP协议是一种面向非连接的协议，在正式通信前不必与对方先建立连接，不管对方状态就直接发送。每个数据报都是一个独立的信息，包括完整的源地址或目的地址，它在网络上以任何可能的路径传往目的地，因此能否到达目的地，到达目的地的时间以及内容的正确性都是不能被保证的。这与现在手机短信非常相似。 UDP适用于一次只传送少量数据、对可靠性要求不高的应用场合。

9.4.2 S3C2410以太网接口

- 在嵌入式系统中增加以太网接口，通常由如下两种方法实现：

法1：采用带有以太网接口的嵌入式处理器实现

- 这种方法要求嵌入式处理器有通用的以太网接口，通常这种处理器是面向网络应用而设计的，通过内部总线的方法实现处理器和网络数据的交换。

法2：采用嵌入式处理器+ 以太网控制芯片实现

- 这种方法对嵌入式处理器没有特殊要求，只要将以太网控制芯片连接到嵌入式处理器的总线上即可。此方法通用性强， 不受处理器的限制，但是，处理器和网络数据交换通过外部总线交换数据。

- S3C2410芯片内部并没有专用的以太网接口控制器，因此它需要通过并行总线来外接一个以太网控制器，才能实现其连接以太网的需要。

- **几种以太网控制芯片**

(1) RTL8019AS，10 Mbps：Realtek公司生产的16位以太网控制芯片。

(2) AX88796, 10/100 Mbps：台湾Asix公司生产的低功耗10/100 Mbps自适应快速以太网控制芯片(博创ARM9实验箱就采用此芯片)。

(3) CS8900A，10 Mbps： Cirrus Logic公司生产的低功耗16位以太网控制器，芯片内集成了RAM、10BASE-T收发器，直接ISA总线接口。该芯片的物理层接口、数据传输模式和工作模式等都能根据需要而动态调整，通过内部寄存器的设置来适应不同的应用环境。

CS8900A

- CS8900A采用3V供电电压，最大工作电流55mA，具有全双工通信方式，可编程发送功能，数据碰撞自动重发，自动打包及生成CRC校验码，可编程接收功能，自动切换于DMA和片内RAM，提前产生中断便于数据帧预处理，数据流可降低CPU消耗，自动阻断错误包，可跳线控制EEPROM功能，启动编程支持无盘系统，边界扫描和循回测试，待机和睡眠模式，支持广泛的软件驱动，工业级温度范围，LED指示连接状态和网络活动情况等特点。

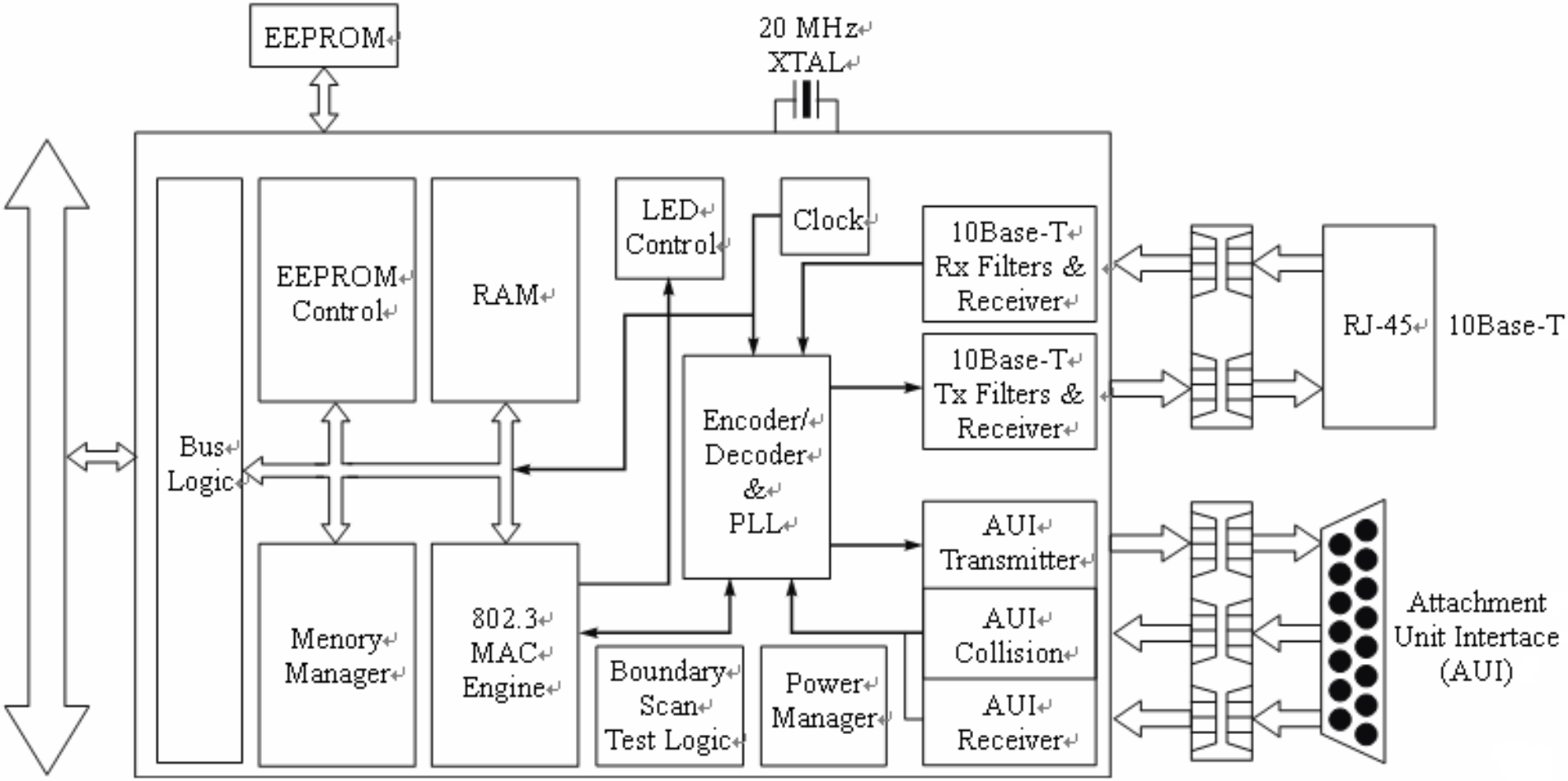


图 CS8900A内部结构框图

CS8900A工作原理

- CS8900A有两种工作模式：Memory和I/O模式。当配置成Memory模式操作时，CS8900A的内部寄存器和帧缓冲区映射到主机内存中连续的4KB的块中，主机可以通过这个块直接访问CS8900A的内部寄存器和帧缓冲区。Memory模式需要硬件上多根地址线和网卡相连。而在I/O模式，对任何寄存器操作均要通过I/O端口0写入或读出。I/O MODE模式在硬件上实现比较方便，而且这也是芯片的默认模式。在I/O模式下，PacketPage存储器被映射到CPU的8个16位的I/O端口上。在芯片被加电后，I/O基地址的默认值被置为300H。

- 使用CS8900A作为以太网的物理层接口，在收到由主机发来的数据报后 (从目的地址域到数据域)，侦听网络线路。如果线路忙，它就等到线路空闲为止，否则，立即发送该数据帧。在发送过程中，首先它添加以太网帧头(包括前导字段和帧开始标志)，然后生成CRC校验码，最后将此数据帧发送到以太网上。
- 在接收过程中，它将从以太网收到的数据帧在经过解码、去帧头和地址检验等步骤后缓存在片内。在CRC校验通过后，它会根据初始化配置情况，通知主机CS8900A收到了数据帧，最后，用某种传输模式(I/O模式、Memory模式、DMA模式) 传到主机的存储区中。

CS8900A引脚信号及功能

- CS8900A的ISA总线接口引脚端和功能如表1所示，EEPROM和引导编程接口引脚端和功能如表2所示，IOBASE-T接口引脚端和功能如表3所示，附加单元接口AUI引脚端和功能如表4所示，通用引脚端和功能如表5所示。

表9.10 ISA总线接口引脚功能

引脚	类型	功能
SA[0:19]	I	地址总线
SD[0:15］	I/O	双向数据总线，三态输出
RESET	I	复位输入端，高电平有效(至少保持400ns)
AEN	I	地址使能，高电平有效
nMEMR	I	存储器读信号，低电平有效
nMEMW	I	存储器写信号，低电平有效
nMEMCS 16	O	存储器16位选择信号，OC(集电极开路)输出
nREFRESH	I	刷新信号，低电平有效。当nREFRESH为低电平时，nMEMR，nMEMW，nIOR，nIOW，nDMACK0，nDMACKl和nDMACK2都被忽略

表9.1 ISA总线接口引脚功能(Cont.)

nIOR	I	I/O读信号，低电平有效
nIOW	I	I/O写信号，低电平有效
nIOCS16	I	16位I/O片选信号，低电平有效
IOCHRDY	O	I/O通道就绪信号，OC(集电极开路)输出
nSBHE	I	系统总线高位使能信号，低电平有效
INTRQ[0:2]	O	中断请求信号，三态输出
DMARQ[0:2]	O	DMA请求信号，三态输出
nDMACK[0:2]	I	DMA应答信号，低电平有效
nCHIPSEL	I	片选信号，低电平有效

表9.11 EEPROM和引导编程接口引脚及功能

引脚	类型	功能
EESK	I	EEPROM时钟输入信号
nEECS	I	EEPROM片选输入信号，低电平有效
EEDataIN	I	EEPROM数据输入，内部上拉
ELCS	I	外部逻辑片选信号，内部上拉
EEDataOUT	O	EEPROM数据输出
CSOUT	O	外部引导编程选择信号输出，低电平有效

表9.12 10BASE-T接口引脚

引脚	类型	功能
TXD+、TXD-	O	数据发送，差分对管输出
RXD+、RXD-	I	数据接收，差分对管输入

表9.13　附加单元接口引脚及功能

引脚	类型	功能
DO+、DO-	O	AUI数据输出，差分对管输出
DI＋、DI-	I	AUI数据输入，差分对管输入
CI＋、CI-	I	AUI振动输入，差分对管输入

表9.14 通用引脚及功能

引脚	类型	功能
XTAL[1:2]	I/O	晶体振荡器输入 / 输出
nSLEEP	I	硬件睡眠控制输入信号，低电平有效，内部上拉
nLINKLED/nHCO	O	线路正常输出信号或主控制器输出0信号，低电平有效，OC（集电极开路）输出
nBSTAUTS/nHC1	O	总线状态输出信号或主控制器输出1信号，低电平有效，OC（集电极开路）输出
nLANLED	O	网络状态指示输出信号，OC（集电极开路）输出
nTEST	I	测试输入使能信号，低电平有效，内部上拉
RES	I	基准电阻输入端
DVDD[1:4]	I	数字电路电源
DVSS［1:4］	I	数字电路地
AVDD[1:4]	I	模拟电路电源
AVSS［1:4]	I	模拟电路地

基于CS8900A的以太网接口

- S3C4410与CS8900A接口芯片连接构成以太网接口

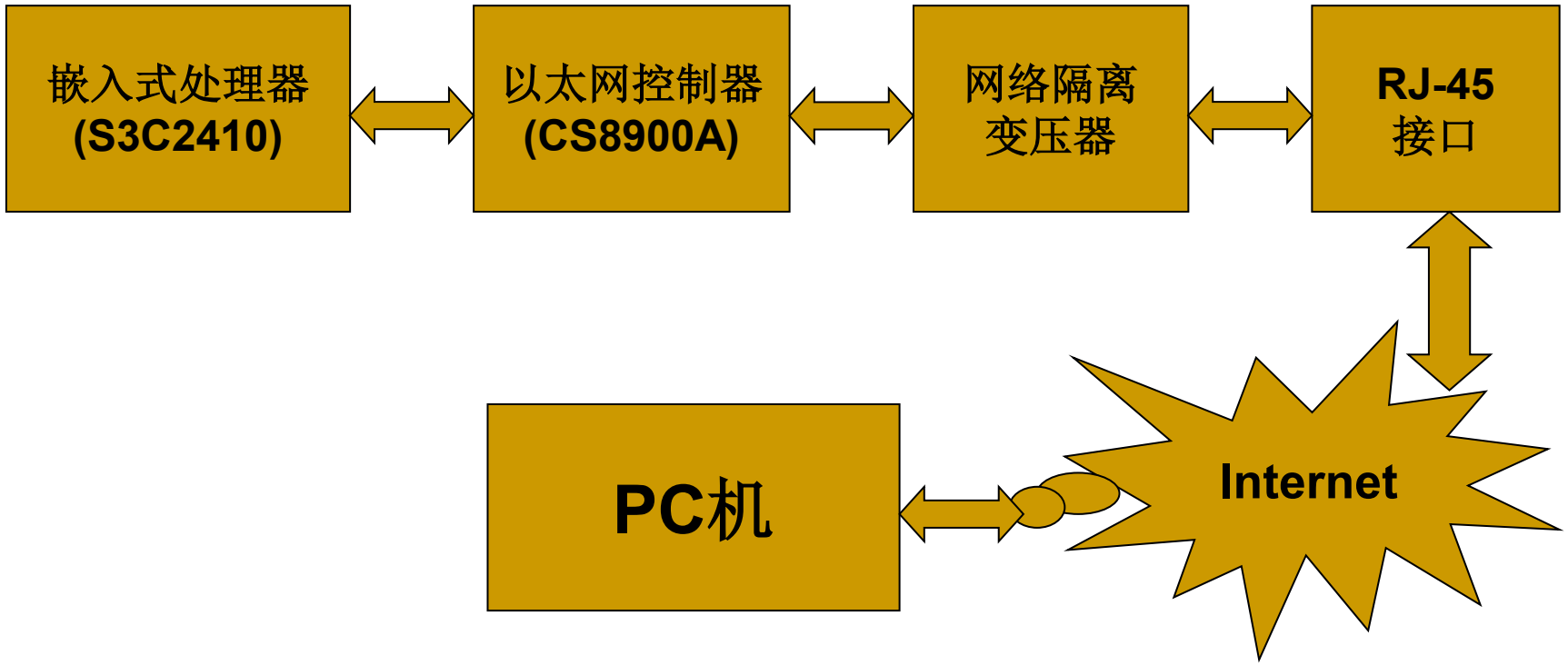


图9.20　S3C2410与CS8900A构成以太网框图

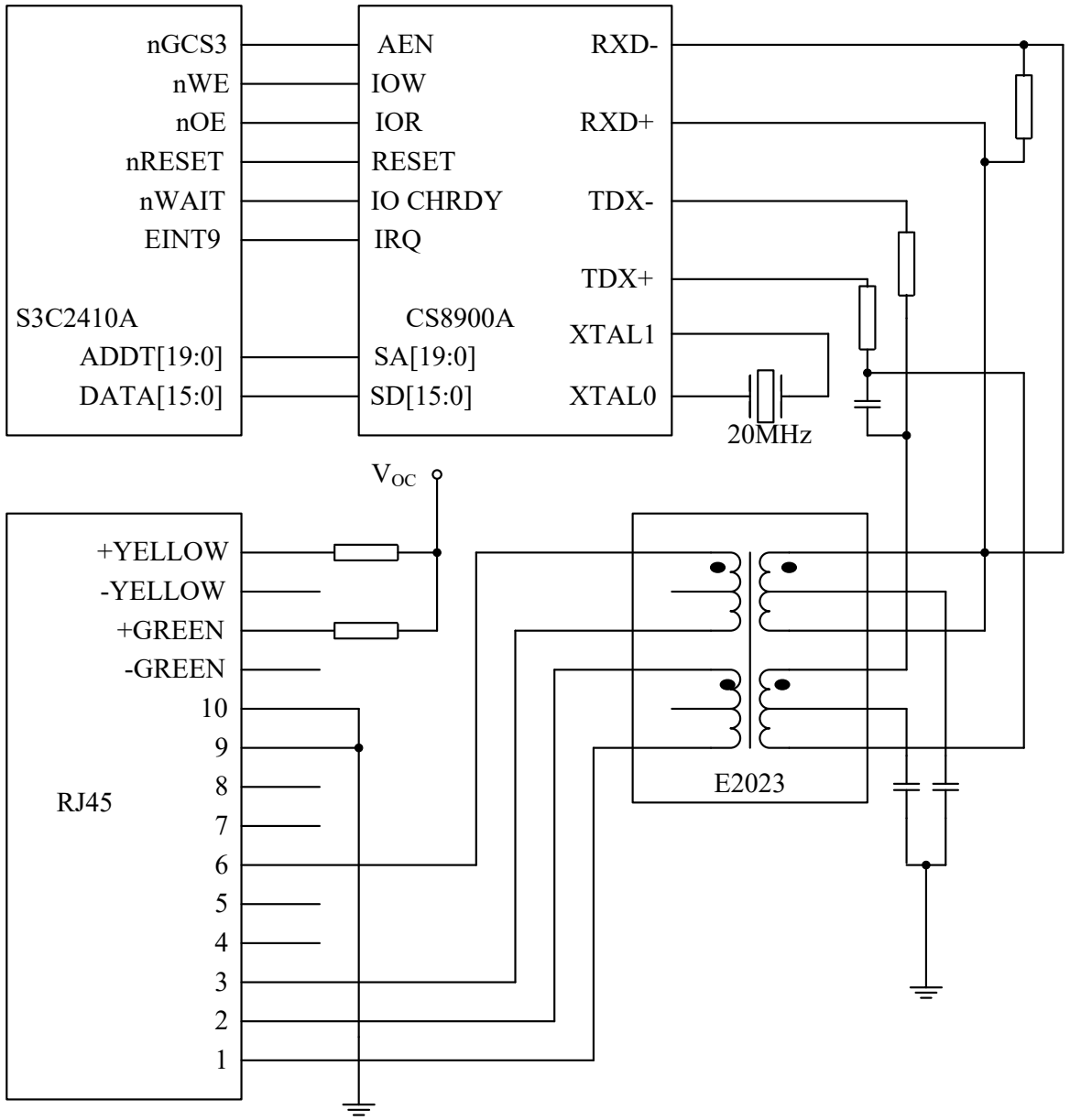


图9.21 S3C2410A与 CS8900A连接构成的以太网接口原理图

CS8900A与S3C2410接口主要引脚说明如下：

(1)SA[19:0]，20位地址线，分别与S3C2410的ADDR[19:0]相连；

(2)SD[15:0]，16位地址线，分别与S3C2410的DATA[15:0]相连；

(3)AEN，地址使能，与nGCS3连接，当系统配置为内存模式或I/O模式时，它都接低电平；

(4)IOR/IOW，分别为I/O模式的读/写信号；

(5)IO CHRDY，I/O通道就绪信号，集电极开路输出，需接上拉电阻，连接至S3C2410的nWAIT。

(6)IRQ，中断请求输出，接S3C2410的某个中断输入，这里是EINT9。CS8900A与RJ-45接口之间需接一个网络变压器，该变压器起到电平转换及电气隔离的作用。

- CS8900A有两种工作模式：MEMORY MODE和I/O MODE。

(1) 在MEMORY MODE下编程操作较为简单，对任何寄存器都是直接操作，不过这需要硬件上多根地址线和网卡相连。

(2) I/O MODE则较为麻烦，因为这种模式下对任何寄存器操作均要通过I/O端口0写入或读出，但这种模式在硬件上实现比较方便，而且这也是芯片的默认模式。

(3) 在I/O模式下，PacketPage存储器被映射到CPU的8个16位的I/O端口上。在芯片被加电后，I/O基地址的默认值被置为300H。

表9.15 CS8900A I/O模式端口分配表

偏移	读写类型	描述
0000H	读/写	接收/发送数据（端口0）
0002H	读/写	接收/发送数据（端口1）
0004H	写	TxCMD发送命令
0006H	写	TxLength发送长度
0008H	读	ISQ中断状态队列
000AH	读/写	PacketPage指针
000CH	读/写	PacketPage数据（端口0）
000EH	读/写	PacketPage数据（端口1）

9.4.3 socket网络编程

1．socket网络函数

嵌入式Linux网络编程需要用到一系列的socket网络函数，下面介绍主要函数。

(1) socket函数

函数原型：

```
int socket(int family, int type, int protocol);
```

(2) bind函数

函数原型：

```
int bind(int sockfd, struct sockaddr * myaddr, int addrlen);
```

(3) 地址结构

```
struct sockaddr
{   unsigned short sa_family; /*通信协议类型族，AF_xxx */
    char sa_data[14];    /*14字节协议地址，包含该socket的IP地址和端口号*/
};
struct sockaddr_in
{   short int       sin_family;  /*通信协议类型族*/
    unsigned short int sin_port; /*端口号*/
    struct in _addr  sin _addr;   /*IP地址*/
    unsigned char  sin_zero[8]; /*填充0以保持与socdaddr结构的长度相同*/
}
```

(4) connect函数

函数原型：

int connect(int sockfd,const struct sockaddr * serv_addr,socklen_t addrlen);

(5) listen函数

函数原型：

int listen(int sockfd,int backlog);

(6) accept函数

函数原型：

int accept(int sockfd,struct sockaddr * cliaddr,socklen_t * addrlen);

(7) send和recv函数

这两个函数分别用于发送和接收数据。

函数原型分别为：

int send(int sockfd, const void * msg, int len, int flags);

int recv(int sockfd, void * buf, int len, unsigned int flags);

(8) sendto函数和recvfrom函数

这两个函数的作用与send函数和recv函数类似，也用于发送和接收数据。

函数原型：

```
int  sendto(int  sockfd,
        const void  * msg,
        int  len
        unaigned int  flags,
        const struct sockaddr  * to,
        int  tolen);

int recvfrom(int   sockfd,
      void  * byf,
      int  len,
      unsigned  int flags,
      struct  sockaddr * from,
      int  * fromlen);
```

2．socket网络编程举例

使用socket方式进行网络数据通信大致分为如下几个步骤：

① 创建服务器socket，绑定建立连接的端口。

② 服务器程序在一个端口处于阻塞状态，等待客户机的连接。

③ 创建客户端socket对象，绑定主机名称或IP地址，指定连接端口号。

④ 客户端socket发起连接请求。

⑤ 建立连接。

⑥ 利用send/sendto函数和tecv/recvfrom函数进行数据传输。

⑦ 关闭socket。

例9.1：编写TCP socket网络服务器程序。

见教材P239。

例9.2：编写TCP socket网络客户端程序。

见教材P241。

9.4.4 嵌入式Web服务器程序设计

1．Web服务器

(1) HTTP协议

HTTP是Hyper Text Transfer Protocol超文本传输协议的缩写，主要用于Web方式的数据传输。它能以文本、超文本、音频和视频等形式传输数据。它之所以被称为超文本传输协议，是因为它能有效地用在超文本环境中，能迅速地从一个文件跳转到另一文件。HTTP协议是TCP协议的一个连接应用，可用于客户端和服务器之间传输数据。HTTP的思想非常简单，客户端发送一个请求给服务器，服务器则返回一个响应给客户端。

(2) Web服务器的工作原理

超文本文档存放在服务器中，当客户端请求访问文档时，服务器将文档的一个副本发送出去。客户端可以使用浏览器显示文档。Web服务器的任务有两个：第一个任务是建立客户端和服务器的socket连接；第二个任务是当服务器接收到客户端的请求后，对请求数据进行解析，按其要求，将存放在服务器的文件内容复制成副本，发送给客户端。

2．Web服务器的程序设计

例9.3：编写一个简易的Web服务器程序。

见教材P246。

End of Chapter 9