MII接口详解

MII (Media Independent Interface(介质无关接口)；或称为媒体独立接口，它是IEEE-802.3定义的以太网行业标准。它包括一个数据接口，以及一个MAC和PHY之间的管理接口。数据接口包括分别用于发送器和接收器的两条独立信道。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。管理接口是个双信号接口：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。MII （Management interface）只有两条信号线。   
    MII标准接口用于连快Fast Ethernet MAC-block与PHY。"介质无关"表明在不对MAC硬件重新设计或替换的情况下，任何类型的PHY设备都可以正常工作。在其他速率下工作的与 MII等效的接口有：AUI（10M 以太网）、GMII（Gigabit 以太网）和XAUI（10-Gigabit 以太网）。   
MII总线   
    在IEEE802.3中规定的MII总线是一种用于将不同类型的PHY与相同网络控制器（MAC）相连接的通用总线。网络控制器可以用同样的硬件接口与任何PHY进行连接。   
MII相关接口介绍   
    以太网媒体接口有：MII RMII SMII GMII   
    所有的这些接口都从MII而来，MII是(Medium Independent Interface）的意思，是指不用考虑媒体是铜轴、光纤、电缆等，因为这些媒体处理的相关工作都有PHY或者叫做MAC的芯片完成。   
    MII支持10兆和100兆的操作，一个接口由14根线组成，它的支持还是比较灵活的，但是有一个缺点是因为它一个端口用的信号线太多，如果一个8端口的交换机要用到112根线，16端口就要用到224根线，到32端口的话就要用到448根线，一般按照这个接口做交换机，是不太现实的，所以现代的交换机的制作都会用到其它的一些从MII简化出来的标准，比如RMII、SMII、GMII等。   
    RMII是简化的MII接口，在数据的收发上它比MII接口少了一倍的信号线，所以它一般要求是50兆的总线时钟。RMII一般用在多端口的交换机，它不是每个端口安排收、发两个时钟，而是所有的数据端口公用一个时钟用于所有端口的收发，这里就节省了不少的端口数目。RMII的一个端口要求7个数据线，比MII少了一倍，所以交换机能够接入多一倍数据的端口。和MII一样，RMII支持10兆和100兆的总线接口速度。   
    SMII是由思科提出的一种媒体接口，它有比RMII更少的信号线数目，S表示串行的意思。因为它只用一根信号线传送发送数据，一根信号线传输接受数据，所以在时钟上为了满足100的需求，它的时钟频率很高，达到了125兆，为什么用125兆，是因为数据线里面会传送一些控制信息。SMII一个端口仅用4根信号线完成100信号的传输，比起RMII差不多又少了一倍的信号线。SMII在工业界的支持力度是很高的。同理，所有端口的数据收发都公用同一个外部的125M时钟。   
    GMII是千兆网的MII接口，这个也有相应的RGMII接口，表示简化了的GMII接口。   
MII工作原理   
    “媒体独立”表明在不对MAC硬件重新设计或替换的情况下，任何类型的PHY设备都可以正常工作。包括分别用于发送器和接收器的两条独立信道。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，包括TX\_ER，TXD，TX\_EN，TX\_CLK，COL，RXD，RX\_EX，RX\_CLK，CRS，RX\_DV等。   
    MII以4位半字节方式传送数据双向传输，时钟速率25MHz。其工作速率可达100Mb/s。   
    MII管理接口是个双信号接口，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，其管理是使用SMI（Serial Management Interface）总线通过读写PHY的寄存器来完成的。   
    PHY里面的部分寄存器是IEEE定义的，这样PHY把自己的目前的状态反映到寄存器里面，MAC通过SMI总线不断的读取PHY的状态寄存器以得知目前PHY的状态，例如连接速度，双工的能力等。   
    当然也可以通过SMI设置PHY的寄存器达到控制的目的，例如流控的打开关闭，自协商模式还是强制模式等。   
    不论是物理连接的MII总线和SMI总线还是PHY的状态寄存器和控制寄存器都是有IEEE的规范的，因此不同公司的MAC和PHY一样可以协调工作。当然为了配合不同公司的PHY的自己特有的一些功能，驱动需要做相应的修改。   
    PHY是物理接口收发器，它实现物理层。包括MII/GMII（介质独立接口）子层、PCS（物理编码子层）、PMA（物理介质附加）子层、PMD（物理介质相关）子层、MDI子层。100BaseTX采用4B/5B编码。   
    PHY在发送数据的时候，收到MAC过来的数据（对PHY来说，没有帧的概念，对它来说，都是数据而不管什么地址，数据还是CRC），每4bit就增加1bit的检错码，然后把并行数据转化为串行流数据，再按照物理层的编码规则把数据编码，再变为模拟信号把数据送出去。收数据时的流程反之。   
    PHY还有个重要的功能就是实现CSMA/CD的部分功能。   
    它可以检测到网络上是否有数据在传送，如果有数据在传送中就等待，一旦检测到网络空闲，再等待一个随机时间后将数据送出去。如果两个碰巧同时送出了数据，那样必将造成冲突，这时候，冲突检测机构可以检测到冲突，然后各等待一个随机的时间重新发送数据。这个随机时间很有讲究的，并不是一个常数，在不同的时刻计算出来的随机时间都是不同的，而且有多重[算法](http://lib.csdn.net/base/31)来应付出现概率很低的同两台主机之间的第二次冲突。   
    通信速率通过双方协商，协商的结果是两个设备中能同时支持的最大速度和最好的双工模式，这个技术被称为Auto Negotiation或者NWAY。   
    隔离变压器把PHY送出来的差分信号用差模耦合的线圈耦合滤波以增强信号，并且通过电磁场的转换耦合到连接网线的另外一端。   
    RJ-45中1、2是传送数据的，3、6是接收数据的。   
    新的PHY支持AUTO MDI-X功能，也需要隔离变压器支持，它可以实现RJ-45接口的1、2上的传送信号线和3、6上的接收信号线的功能自动互相交换。   
GMII简介   
    GMII (Gigabit MII)   
    GMII采用8位接口数据，工作时钟125MHz，因此传输速率可达1000Mbps。同时兼容MII所规定的10/100 Mbps工作方式。   
    GMII接口[数据结构](http://lib.csdn.net/base/31)符合IEEE以太网标准。该接口定义见IEEE 802.3-2000。   
    发送器：   
    ◇ GTXCLK——吉比特TX..信号的时钟信号（125MHz）   
    ◇ TXCLK——10/100M信号时钟   
    ◇ TXD[7..0]——被发送数据   
    ◇ TXEN——发送器使能信号   
    ◇ TXER——发送器错误（用于破坏一个数据包）   
    注：在千兆速率下，向PHY提供GTXCLK信号，TXD、TXEN、TXER信号与此时钟信号同步。否则，在10/100M速率下，PHY提供 TXCLK时钟信号，其它信号与此信号同步。其工作频率为25MHz（100M网络）或2.5MHz（10M网络）。   
    接收器：   
    ◇ RXCLK——接收时钟信号（从收到的数据中提取，因此与GTXCLK无关联）   
    ◇ RXD[7..0]——接收数据   
    ◇ RXDV——接收数据有效指示   
    ◇ RXER——接收数据出错指示   
    ◇ COL——冲突检测（仅用于半双工状态）   
    管理配置   
    ◇ MDC——配置接口时钟   
    ◇ MDIO——配置接口I/O   
    管理配置接口控制PHY的特性。PHY器件有32个寄存器，MAC控制器可通过管理配置接口来配置这些寄存器，每个寄存器的地址为16位。其中前16个寄存器已经在“IEEE 802.3,2000-22.2.4 Management Functions”中规定了用途，其余的则由各器件自己指定。   
RMII简介   
    RMII: Reduced Media Independant Interface 即简化媒体独立接口；是标准的以太网接口之一，比MII有更少的I/O传输。   
    关于RMII口和MII口的问题   
    RMII口是用两根线来传输数据的，   
    MII口是用4根线来传输数据的，   
    GMII是用8根线来传输数据的。   
    MII/RMII只是一种接口，对于10M线速,MII的速率是2.5M，RMII则是5M；对于100M线速，MII的速率是25M，RMII则是50M。   
    MII/RMII 用于传输以太网包，在MII/RMII接口是4/2bit的，在以太网的PHY里需要做串并转换、编解码等后，才能在双绞线和光纤上进行传输，其帧格式遵循IEEE 802.3(10M)/IEEE 802.3u(100M)/IEEE 802.1q(VLAN)。   
    以太网帧的格式为：前导符+开始位+目的mac地址+源mac地址+类型/长度+数据+padding(optional)+32bitCRC   
    如果有vlan，则要在类型/长度后面加上2个字节的vlan tag，其中12bit来表示vlan id，另外4bit表示数据的优先级！