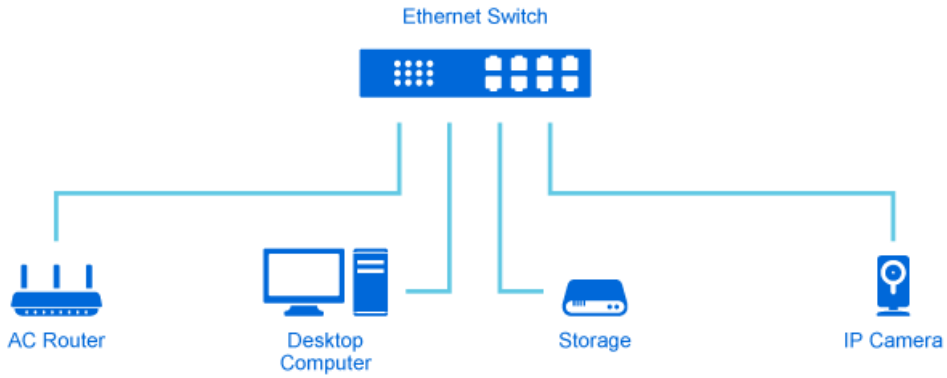
پروتکل ارتباطی Ethernet ، از تئوری تا عمل



تالیف و گردآوری : محمد حسین کوهی قمصری ، با حمایت سایت

KnowledgePlus.ir

مسابقه علمی سال ۱۳۹۵

۰۹۳۶۱۹۵۵۳۵۰

Mohammadghamsari@ieee.org

mohammadghamsari18@yahoo.com

فهرست

فصل اول ۱
۱-۱. مقدمه ۱
۱-۲. چکیده مقاله ۲
فصل دوم ۴
اصول شبکه های کامپیوتری ۴
۲-۱. مقدمه ۴
۲-۲. اصطلاحات شبکه ۵
۲-۳. شبکه های کامپیوتری ۶
۲-۳-۱. شبکه محلی و شبکه گسترده ۷
۲-۳-۲. انواع توپولوژی های شبکه ۸
۲-۳-۲-۱. توپولوژی Bus ۸
۲-۳-۲-۲. توپولوژی Star ۹
۲-۳-۲-۳. توپولوژی Ring ۱۰
۲-۳-۲-۴. توپولوژی Mesh ۱۱
۲-۴. تجهیزات شبکه ۱۱
۲-۴-۱. تکرارکننده (Repeater) ۱۳
۲-۴-۲. پل (Bridge) ۱۳
۲-۴-۳. مسیرده (Router) ۱۴
۲-۴-۴. سوییچ (Switch) ۱۵
۲-۴-۵. کارت شبکه (Network Interface Card یا NIC) ۱۷
۲-۵. کابل های شبکه ۱۸
۲-۵-۱. کابل های هم محور (Coaxial) ۱۸
۲-۵-۲. کابل های زوج به هم تابیده شده (Twisted Pairs) ۱۸
۲-۵-۳. فیبر نوری (Optical-Fiber) ۱۹
۲-۶ آدرس های شبکه ۱۹

۲-۶-۱. آدرس MAC و آدرس IP ۱۹
۲-۶-۲. نام دامنه ۲۲
۲-۶-۳. ساختار آدرس IP ۲۲
۲-۶-۴. زیرشبکه (Sub-network) ۲۳
۲-۶-۵. پروتکل DHCP ۲۴
۲-۷. اتصال شبکه محلی به اینترنت ۲۴
فصل سوم ۲۷
تاریخچه ۲۷
۳-۱. مقدمه ۲۷
۳-۲. تاریخچه ۲۷
فصل چهارم ۳۵
اینترنت ۳۵
۴-۱. مقدمه ۳۵
۴-۲. راهنمای واژگان مورد استفاده در شبکه های Ethernet ۳۷
۴-۳. مدل مرجع OSI و TCP/IP ۳۹
۴-۳-۱. مدل OSI ۳۹
۴-۳-۱-۱. لایه Application (کاربرد) ۴۰
۴-۳-۱-۲. لایه Presentation (ارائه) ۴۱
۴-۳-۱-۳. لایه Session (جلسه) ۴۱
۴-۳-۱-۴. لایه Transport (انتقال) ۴۲
۴-۳-۱-۵. لایه Network (شبکه) ۴۳
۴-۳-۱-۶. لایه Data Link (ارتباط داده) ۴۴
۴-۳-۱-۷. لایه Physical (فیزیکی) ۴۵
۴-۳-۱-۸. خلاصه عملکرد هر لایه ۴۶
۴-۳-۲. مدل TCP/IP ۴۷
۴-۳-۳. مقایسه مدل OSI با مدل TCP/IP ۴۸
۴-۴. طبقه بندی پروتکل ها و جایگاه اینترنت ۵۰

۴-۴-۱. بسته بندی (encapsulation) فریم ها ۵۰
۴-۴-۲. پروتکل های لایه کاربرد ۵۲
۴-۴-۳. پروتکل های لایه انتقال ۵۳
۴-۴-۴. پروتکل های لایه شبکه ۵۳
۴-۴-۵. پروتکل های لایه پیوند داده و فیزیکی ۵۴
۴-۵. مفهوم Client/Server ۵۴
۴-۶. مفهوم port و socket ۵۴
۴-۷. مشخصات و ویژگی های اینترنت ۵۷
۴-۸. استانداردهای اینترنت ۵۹
۴-۸-۱. تاریخچه ۵۹
۴-۸-۲. قوائد نام گذاری ۶۰
۴-۸-۳. اینترنت 10Mbit/s (Standard Ethernet) ۶۱
۴-۸-۳-۱. 10Base-5 ..... ۶۳
۴-۸-۳-۲. 10Base-2 ..... ۶۳
۴-۸-۳-۳. 10Base-T ۶۴
۴-۸-۳-۴. 10Base-FL ۶۵
۴-۸-۴. اینترنت 100Mbit/s (Fast Ethernet) ۶۵
۴-۸-۵. اینترنت 1000Mbit/s (Gigabit Ethernet) ۶۷
۴-۸-۶. اینترنت First mile ۶۹
۴-۹. کابل های اینترنت ۷۱
۴-۹-۱. کابل های CAT ۷۱
۴-۹-۲. انواع کابل های محافظ دار ۷۴
۴-۹-۳. کد رنگی استاندارد کابل ها ۷۶
۴-۹-۴. کابل های Patch ۷۷
۴-۹-۵. فیبر نوری ۷۹
۴-۱۰. کانکتورهای اینترنت ۸۰
۴-۱۱. استانداردهای TIA/EIA 568A/B ، کابل های Cross-Over و Straight-Thtough ۸۳
۴-۱۲. AUTO-CROSSOVER ۸۶

۴-۱۳. پیاده سازی استانداردها ۸۷
۴-۱۴. فریم های ایترنت ۹۱
۴-۱۴-۱. ساختار پایه فریم ها ۹۱
۴-۱۴-۲. فریم های کنترلی (Control Frames) ۹۳
۴-۱۴-۳. فریم های VLAN Tagged ۹۴
۴-۱۴-۴. ظرفیت فریم ها ۹۴
۴-۱۵. ارتباط Half-Duplex و پروتکل CSMA/CD ۹۶
۴-۱۶. ارتباط Full-Duplex ۱۰۱
۴-۱۷. زیرلایه های MAC و PHY ۱۰۲
۴-۱۸. سیگنال های ایترنت 10Mbit/s ۱۰۵
۴-۱۹. کد 4B/5B ۱۰۷
۴-۲۰. بلوک دیاگرام کلی کدینگ و دیکدینگ ۱۰۹
۴-۲۱. مذاکره خودکار (Auto-Negotiation) ۱۱۰
۴-۲۲. ایترنت تعبیه شده (Embeded Ethernet) ۱۱۲
۴-۲۳. ایترنت صنعتی (Industrial Ethernet) ۱۱۵
فصل پنجم ۱۲۰
نرم افزار ۱۲۰
۵-۱. مقدمه ۱۲۰
۵-۲. پروتکل HTTP ۱۲۲
۵-۲-۱. جلسه HTTP (HTTP Session) ۱۲۲
۵-۲-۲. روش های درخواست (Request methods) ۱۲۳
۵-۲-۳. ساختار پیام ها (Message Format) ۱۲۳
۵-۳. پروتکل TCP ۱۲۵
۵-۴. پروتکل UDP ۱۲۹
۵-۵. پروتکل ICMP ۱۳۱
۵-۶. پروتکل IP ۱۳۳
۵-۷. پروتکل ARP ۱۳۵
فصل ششم ۱۴۰

سخت افزار ۱۴۰
۶-۱. مقدمه ۱۴۰
۶-۲. بلوک دیاگرام اتصال به شبکه ایترنت ۱۴۰
۶-۳. چند نکته برای اتصال به شبکه ایترنت ۱۴۲
۶-۴. معرفی تراشه های مورد نیاز برای حالت پیاده سازی اول ۱۴۶
۶-۴-۱. میکروکنترلر PIC18F87J60 ..... ۱۴۶
۶-۴-۲. میکروکنترلر PIC18F97J60 ..... ۱۴۶
۶-۵. معرفی تراشه های مورد نیاز برای حالت پیاده سازی دوم ۱۴۷
۶-۵-۱. میکروکنترلرهای LPC1769/68/67/66/65/64/63 ..... ۱۴۷
۶-۵-۲. میکروکنترلرهای LPC2364/66/68 ..... ۱۴۷
۶-۵-۳. میکروکنترلرهای سری STM32F105xx و STM32F107xx ۱۴۸
۶-۶. معرفی تراشه های مورد نیاز برای حالت پیاده سازی سوم ۱۵۰
۶-۶-۱. تراشه ENC28J60 ..... ۱۵۰
۶-۶-۲. تراشه ENC424J600/626J600 ..... ۱۵۲
۶-۶-۳. تراشه RTL8019AS ۱۵۵
۶-۶-۴. تراشه DM9000AEP ۱۵۷
۶-۶-۵. تراشه W5300 ..... ۱۵۹
۶-۷. معرفی تراشه های مورد نیاز برای حالت پیاده سازی چهارم ۱۶۴
۶-۷-۱. تراشه W3100A ۱۶۴
۶-۸. تراشه های Transceiver ایترنت ۱۶۶
۶-۸-۱. تراشهRTL8201 ..... ۱۶۶
۶-۸-۲. تراشه DM916 ..... ۱۶۷
۶-۸-۳. تراشه DP83848 ..... ۱۶۹
۶-۸-۴. تراشه LAN8710A ۱۷۱
۶-۸-۵. تراشه LAN8720 ..... ۱۷۴
۶-۸-۶. تراشه RTL8201BL ۱۷۸
۶-۸-۷. تراشه DP83822 ..... ۱۷۹
۶-۸-۸. تراشه DP83849IF ۱۸۲

۶-۹. پیاده سازی Gigabit Ethernet ۱۸۴
۶-۹-۱. تراشه BCM5482 ..... ۱۸۴
۶-۹-۲. تراشه DP83867IR/CR ۱۸۵
۶-۱۰. کاربردهای بدون ترانسفورماتور ۱۸۹
فصل هفتم ۱۹۵
تراشه ENC28J60 ..... ۱۹۵
۷-۱. مقدمه ۱۹۵
۷-۲. ویژگی های تراشه ENC28J60 ..... ۱۹۶
۷-۲-۱. ویژگی های بخش کنترل کننده ی Ethernet ۱۹۶
۷-۲-۲. ویژگی های بخش Buffer ۱۹۶
۷-۲-۳. ویژگی های بخش MAC ۱۹۶
۷-۲-۴. ویژگی های بخش PHY (لایه فیزیکی) ۱۹۷
۷-۲-۵. ویژگی های عملیاتی (Operational) ۱۹۷
۷-۳. بلوک دیاگرام ۱۹۸
۷-۴. اتصال تراشه ENC28J60 به کنترل کننده ۱۹۹
۷-۵. توضیحات پایه های تراشه ENC28J60 ..... ۲۰۰
۷-۶. واحد اسیلاتور ۲۰۱
۷-۶-۱. تایمر زمان شروع (Oscillator Start-Up Timer) ۲۰۱
۷-۷. پایه CLKOUT ۲۰۲
۷-۸. مدارات جانبی ۲۰۳
۷-۹. سطوح منطقی پایه های I/O ۲۰۵
۷-۱۰. پیکربندی LED های وضعیت شبکه ۲۰۶
۷-۱۱. مدیریت حافظه ۲۰۷
۷-۱۲. رجیسترهای کنترلی ۲۰۸
۷-۱۳. رجیستر ECON1 ..... ۲۰۹
۷-۱۴. حافظه بافر ایترنت ۲۱۰
۷-۱۵. رجیسترهای لایه فیزیکی (PHY) ۲۱۳
۷-۱۵-۱. پویش (Scan) رجیسترهای بخش فیزیکی ۲۱۴

۷-۱۵-۲. رجیستر MICMD ۲۱۵
۷-۱۵-۳. رجیستر MISTAT ۲۱۵
۷-۱۶. رجیسترهای PHSTAT ۲۱۶
۷-۱۶-۱. رجیستر PHSTAT1 ..... ۲۱۶
۷-۱۶-۲. رجیستر PHSTAT2 ..... ۲۱۷
۷-۱۷. رابط SPI ۲۱۸
۷-۱۷-۱. دستور READ CONTROL REGISTER ۲۱۹
۷-۱۷-۲. دستور READ BUFFER MEMORY ۲۲۰
۷-۱۷-۳. دستور WRITE CONTROL REGISTER ۲۲۱
۷-۱۷-۴. دستور WRITE BUFFER MEMORY ۲۲۲
۷-۱۷-۵. دستورات BIT FIELD SET ، BIT FIELD CLEAR ، SYSTEM RESET ۲۲۳
۷-۱۸. ساختار بسته های ایترنت ۲۲۳
۷-۱۸-۱. PREAMBLE/START OF FRAME DELIMITER ۲۲۳
۷-۱۸-۲. DESTINATION ADDRESS ۲۲۳
۷-۱۸-۳. SOURCE ADDRESS ۲۲۴
۷-۱۸-۴. DATA ۲۲۴
۷-۱۸-۵. PADDING ۲۲۴
۷-۱۸-۶. CRC ۲۲۵
۷-۱۹. پیکربندی اولیه (INITIALIZATION) ۲۲۵
۷-۱۹-۱. بافر گیرنده ۲۲۵
۷-۱۹-۲. بافر فرستنده ۲۲۶
۷-۱۹-۳. فیلرتهای گیرنده ۲۲۶
۷-۱۹-۴. زمان OST ۲۲۶
۷-۱۹-۵. تنظیمات اولیه MAC ۲۲۷
۷-۱۹-۶. رجیستر MACON1 ..... ۲۲۸
۷-۱۹-۷. رجیستر MACON3 ..... ۲۲۸
۷-۱۹-۸. رجیستر MACON4 ..... ۲۲۸
۷-۱۹-۹. رجیستر MABBIPG ۲۲۸

۷-۱۹-۱۰. تنظیمات اولیه رجیسترهای لایه فیزیکی ۲۲۹
۷-۲۰. بسته های ارسالی و دریافتی ۲۳۰
۷-۲۰-۱. بسته های ارسالی ۲۳۰
۷-۲۰-۲. بسته های دریافتی ۲۳۲
۷-۲۰-۳. خواندن بسته های دریافتی ۲۳۳
۷-۲۰-۴. آزادسازی فضای بسته های دریافتی ۲۳۳
۷-۲۱. فیلترهای گیرنده ۲۳۵
۷-۲۱-۱. رجیستر ERXFCON ۲۳۵
۷-۲۱-۲. فیلتر Unicast ۲۳۶
۷-۲۱-۳. فیلتر تطبیق الگو (Pattern Match Filter) ۲۳۶
۷-۲۱-۴. فیلتر Magic Packet ۲۳۶
۷-۲۱-۵. فیلتر جدول Hash ۲۳۷
۷-۲۱-۶. فیلتر Multicast ۲۳۸
۷-۲۱-۷. فیلتر Broadcast ۲۳۸
۷-۲۲. وقفه ۲۳۹
۷-۲۲-۱. رجیستر ESTAT ۲۴۰
۷-۲۲-۲. رجیستر EIE ۲۴۰
۷-۲۲-۳. رجیستر EIR ۲۴۱
۷-۲۲-۴. رجیستر PHIE ۲۴۱
۷-۲۲-۵. رجیستر PHIR ۲۴۲
فصل هشتم ۲۴۳
پیاده سازی عملی ایترنت ۲۴۳
۸-۱. مقدمه ۲۴۳
۸-۲. سخت افزار پروژه ۲۴۵
۸-۲-۱. روش اول ۲۴۵
۸-۲-۲. روش دوم ۲۴۹
۸-۲-۳. روش سوم ۲۵۰
۸-۳. نرم افزار پروژه ۲۵۲

۸-۳-۱. پروژه اول ۲۵۲
۸-۳-۲. پروژه دوم ۲۵۳
۸-۳-۲-۱. شرح کلی برنامه ۲۵۶
۸-۳-۲-۲. فایل main.c ۲۶۰
۸-۳-۲-۳. فایل ethernet.h ۲۷۲
۸-۳-۲-۴. تست پروژه ۲۷۸
۸-۳-۳. پروژه سوم ۲۷۹
۸-۳-۳-۱. فایل main.c ۲۸۱
۸-۳-۳-۲. فایل global.c ۲۸۳
۸-۳-۳-۳. فایل WinAVR_CAMP.h ۲۸۴
۸-۳-۳-۴. فایل Ethernet.c ۲۸۵
۸-۳-۳-۵. فایل struct.h ۲۸۶
۸-۳-۳-۶. کتابخانه ARP ۲۸۶
۸-۳-۳-۷. کتابخانه IP ۲۸۹
۸-۳-۳-۸. کتابخانه ICMP ۲۸۹
۸-۳-۳-۹. کتابخانه TCP ۲۹۱
۸-۳-۳-۱۰. کتابخانه UDP ۲۹۲
۸-۳-۳-۱۱. کتابخانه HTTP ۲۹۳
۸-۳-۳-۱۲. تست عملی پروژه ۳۰۲
۸-۴. پروژه های پیشنهادی ۳۰۸
پیوست ها ۳۰۹
پیوست الف) ساختن کابل های Straight-Through و Cross-Over ۳۰۹
پیوست ب) توسعه دهنده اینترنت (Ethernet Extender) ۳۱۷
پیوست ج) ایزوله کننده شبکه (Network Isolator) ۳۱۸
پیوست د) تبدیل کننده فیبر نوری (Fiber Media Converter) ۳۱۹
پیوست ه) مقایسه اینترنت، ایترنت، اینترانت و اکسترانت ۳۲۰
مراجع (References) ۳۲۱

[مطالب و مستندات فنی ارئه شده در این پروژه آموزشی شامل Schematic ، PCB و کدهای نرم افزاری، با رویکرد آموزشی و پیاده سازی کاربردهای متداول مورد نیاز در صنعت می باشد لذا تراشه ها، روش طراحی و مقادیر المان های پروژه های عملی به این منظور ارائه گردیده است.

استفاده از مطالب آموزشی مطرح شده و مستندات فنی در کاربردهای خاص با ویژگی هایی مانند حساس به زمان بودن (time-critical)، حساس به مسائل حفاظتی (safety-critical) مانند پروژه های صنایع پزشکی، صنایع هوا و فضا، صنایع نظامی و... باید با در نظر گرفتن ملاحظات مربوطه در هر حوزه انجام گردد]

فصل اول

۱-۱. مقدمه

در نگارش این مقاله سه هدف به شکل همزمان دنبال می شود :

سادگی مطالب برای کسانی که آشنایی قبلی با مبحث شبکه های کامپیوتری و تکنولوژی Ethernet ندارند.

رویکردی عملی، به شکلی که خواننده پس از مطالعه ی مقاله قادر به طراحی سخت افزار مناسب جهت پیاده سازی پروتکل ارتباطی Ethernet باشد.

کامل و جامع بودن مطالب، به شکلی که نیازی به مراجعه به مقالات،کتب، سایت ها و... نباشد.

همان طور که ذکر شد هدف از نگارش این مقاله ایجاد توانایی در خواننده برای انجام پروژه های عملی با پروتکل ارتباطی Ethernet که امروزه جایگاه ویژه ای در استانداردهای شبکه های صنعتی دارد می باشد.

در این مقاله سعی شده است ضمن حفظ رویکرد عملی، آشنایی با جنبه های غیرعملی شبکه های مبتنی بر Ethernet تا حد لزوم انجام شود.

با توجه به این که در عمل در کاربردهایی لازم است که سیستم طراحی شده به شبکه Ethernet متصل گردد (بدون نیاز به کارت شبکه)، در بخش پروژه عملی این مقاله، هدف ایجاد سخت افزار لازم به

منظور اتصال سیستم های Embeded در کاربردهای مختلف به شبکه های LAN مبتنی بر پروتکل ارتباطی Ethernet می باشد.

۱-۲. چکیده مقاله

با توجه به این که مطالعه و پیاده سازی پروتکل Ethernet در چارچوب مفهوم شبکه های کامپیوتری صورت می پذیرد و به منظور به دست آوردن شناخت و درک کافی از عملکرد این پروتکل، آشنایی با مفاهیم شبکه های کامپیوتری لازم می باشد، در فصل دوم به مرور مفاهیم، اصطلاحات و تجهیزات رایج مورد استفاده در شبکه های کامپیوتری می پردازیم.

در فصل سوم تاریخچه پیدایش تکنولوژی Ethernet ، شرکت ها و افرادی که در توسعه این پروتکل نقش داشته اند و همچنین روند تکامل این پروتکل به شکل تاریخ مند (chronological) بررسی می شود.

در ابتدای فصل چهارم، اصطلاحات و واژه هایی که در ارتباط با شبکه های کامپیوتری و پروتکل Ethernet کاربرد دارند و می توان آنها را به وفور در مقالاتی که در مورد این پروتکل نوشته شده است و در دیتاشیت قطعات مرتبط با این پروتکل مشاهده نمود، در یک جدول جمع آوری شده است. مطالعه تمامی این اصطلاحات و یادگیری آنها در ابتدا لزومی ندارد ولی پیشنهاد می شود مطالعه ای سطحی بر واژه های این جدول داشته باشید و در مطالعه بخش های مختلف این مقاله یا مقالات دیگر در این موضوع برحسب نیاز به این جدول مراجعه فرمایید.

در ادامه ی این فصل در مورد دو مدل رایج برای توصیف مفهومی شبکه های کامپیوتری، یعنی مدل OSI و مدل TCP/IP صحبت می کنیم و تعریف و جایگاه پروتکل Ethernet را در شبکه و نقشی که در کنار دیگر پروتکل ها به عهده دارد را مشخص می کنیم.

در بخش بعدی به منظور ایجاد آشنایی اولیه با Ethernet ، به طور خلاصه مشخصاتی از این پروتکل ارائه شده است.

در بخش های بعدی این فصل در مورد استانداردهای ارائه شده، نسخه های مختلف هر استاندارد و همچنین کابل ها و کانکتورهای متداول مورد استفاده در ایترنت توضیح می دهیم. در بخش بعدی اشاره ای به پیاده سازی شبکه با توجه به استانداردهای ذکر شده می کنیم.

از دیگر بخش های مهم این فصل ساختار فریم های Ethernet می باشد که مورد بررسی قرار می گیرد. در ادامه ی این فصل در مورد مشخصات، کدها و زمان بندی های Ethernet توضیح داده می شود که مطالعه آن به منظور شناخت دقیق تر عملکرد این پروتکل مناسب می باشد.

در انتهای این فصل در مورد دو دسته بندی کاربردی و مهم، یعنی Embeded Ethernet یا ایترنت نهفته و Industrial Ethernet یا ایترنت صنعتی توضیح داده می شود.

در فصل پنجم در مورد طراحی بخش نرم افزاری پروژه های ایترنت نهفته صبحت خواهد شد و با توجه به پروژه ای که می خواهیم در این مقاله پیاده سازی کنیم، در این فصل به توضیح در مورد پروتکل های لایه های مختلف مدل TCP/IP می پردازیم.

در فصل ششم در مورد طراحی سخت افزار Ethernet و روش های مختلف طراحی سخت افزار توسط تراشه های ارائه شده در بازار دنیا می پردازیم.

با توجه به استفاده از تراشه ENC28J60 در پروژه این مقاله، در فصل هفتم با توجه به دیتاشیت این تراشه، اطلاعات لازم برای پیاده سازی این تراشه ارائه خواهد شد.

در نهایت در فصل هشتم با توجه به آموخته های خود در فصول قبلی مقاله، پروژه ای عملی و کاربردی برای پیاده سازی یک وب سرور طراحی و آن را به شکل عملی آزمایش خواهیم کرد.

در بخش پیوست های مقاله نیز به نکات جانبی مانند ساخت کابل شبکه و تجهیزات کمکی مورد استفاده در شبکه ایترنت اشاره خواهیم کرد.

نکته : شکل صحیح نوشتاری و تلفظ پروتکل Ethernet ، ایتِرنت می باشد اما برای کاهش احتمال اشتباه با واژه مصطلح Internet یا اینترنت، در ادامه ی مقاله به جای واژه "ایتِرنت" از واژه "اترنت" استفاده خواهد شد.

نکته : فهرست مقاله دارای قابلیت cross-reference می باشد و با کلیک بر روی عناوین و تیترها، به آن بخش از مقاله انتقال پیدا خواهید کرد.

فصل دوم

اصول شبکه های کامپیوتری

۲-۱. مقدمه

در ابتدای این فصل به مروری کوتاه بر اصطلاحات مورد استفاده در شبکه های کامپیوتری که در بخش های مختلف این مقاله نیز از آنها استفاده می شود می پردازیم.

در بخش بعدی در مورد شبکه های کامپیوتری و انواع شبکه های کامپیوتری شامل شبکه های LAN و شبکه های WAN توضیح خواهیم داد و نسبت آنها را با شبکه های مبتنی بر اترنت توضیح می دهیم.

در بخش بعدی با توجه به این که شبکه های اترنت قدیمی از توپولوژی خطی و شبکه های اترنت امروزی از توپولوژی ستاره ای استفاده می کنند، در مورد انواع توپولوژی یا پیکربندی شبکه های کامپیوتری توضیح خواهیم داد.

در بخش آخر نیز در مورد تجهیزات مورد استفاده در شبکه های کامپیوتری و کابل های مورد استفاده برای متصل کردن بخش های مختلف شبکه توضیح خواهیم داد. امروزه پرکاربردترین تجهیزات مورد استفاده در شبکه های مبتنی بر اترنت Switch ها هستند، همچنین رایج ترین کابل های مورد استفاده، کابل های CAT و فیبرهای نوری هستند که در فصل های بعدی نیز در مورد نحوه ی انتخاب کابل مناسب برای شبکه اترنت توضیح خواهیم داد.

۲-۲. اصطلاحات شبکه

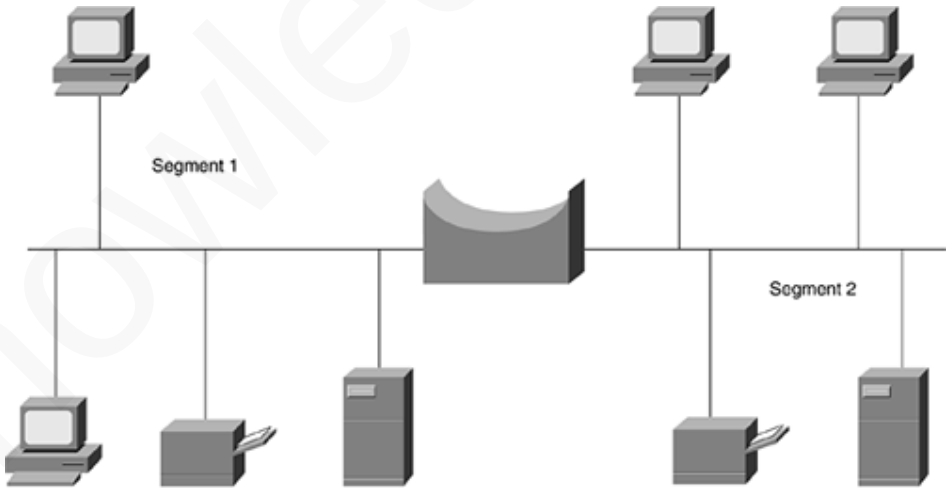
در شبکه های کامپیوتری از اصطلاحات و و لغاتی استفاده می شود که در شبکه اترنت نیز کاربرد دارند. در اینجا به طور خلاصه برخی از پرکاربردترین این اصطلاحات را معرفی می کنیم.

پروتکل : در شبکه های کامپیوتری، به مجموعه قوانینی اطلاق می گردد که نحوه ی ارتباطات را قانونمند می نماید. نقش پروتکل در کامپیوتر نظیر نقش زبان برای انسان است. برای مطالعه یک کتاب نوشته شده به فارسی میبایست خواننده شناخت مناسبی از زبان فارسی داشته باشد.

به منظور ارتباط موفقیت آمیز دو دستگاه در شبکه می بایست هر دو دستگاه از یک پروتکل مشابه استفاده نمایند.

محیط انتقال، رسانه (Medium) : دستگاه های موجود در شبکه از طریق یک محیط انتقال به یکدیگر متصل می گردند که میتواند کابل هم محور (Coaxial)، سیم های زوج به هم تابیده (Twisted pairs)، فیبر نوری (Optical fiber) یا حتی بدون سیم باشد.

بخش (Segment) : به محیط انتقال به اشتراک گذاشته شده منفرد گفته می شود.



شکل ۲-۱

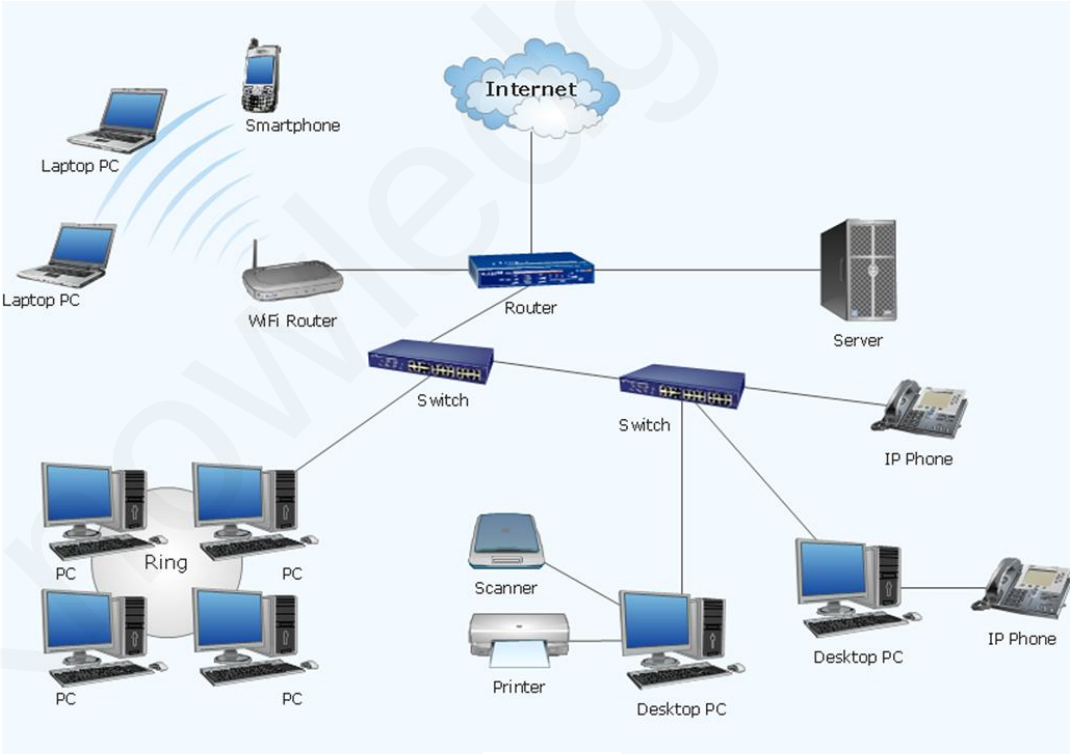
گره (Node)، ایستگاه (Station) : به دستگاه ها و تجهیزات متصل شده به شبکه یا به هر segment مانند کامپیوترها، تجهیزات شبکه و... گفته می شود. در این مقاله از واژه گره استفاده می گردد.

قالب (Frame)، بسته (Packet)، داده نما (datagram) : به یک بلوک اطلاعات که گره ها از طریق

ارسال آنها با یکدیگر ارتباط برقرار می کنند گفته می شود. در واقع فرایند ارسال و دریافت اطلاعات در طول شبکه از طریق قالب ها صورت می پذیرد. قالب ها در شبکه ها مشابه جملات در زبان انسان ها است. در پروتکل های مختلف از واژه های مختلف استفاده می شود، به عنوان مثال در پروتکل UDP به قالب های داده datagram گفته می شود. در پروتکل اترنت استفاده از واژه frame یا قالب کاربرد بیشتری دارد.

۲-۳. شبکه های کامپیوتری

شبکه های کامپیوتری ارتباط بین کامپیوترها را برقرار می کنند و به اشتراک گذاری و انتقال اطلاعات را ممکن می سازند. اینترنت نمونه ای عینی از شبکه های کامپیوتری است و در آن میلیون ها کامپیوتر در سراسر جهان به یکدیگر متصل شده اند. در حقیقت اینترنت بزرگترین شبکه در سراسر جهان به شمار می رود که خود از شبکه های کوچکتر تشکیل شده است.



شکل ۲-۲

Ethernet یک نوع تکنولوژی ارسال اطلاعات مبتنی بر قالب (Frame Based) در شبکه های کامپیوتری LAN به شمار می رود.

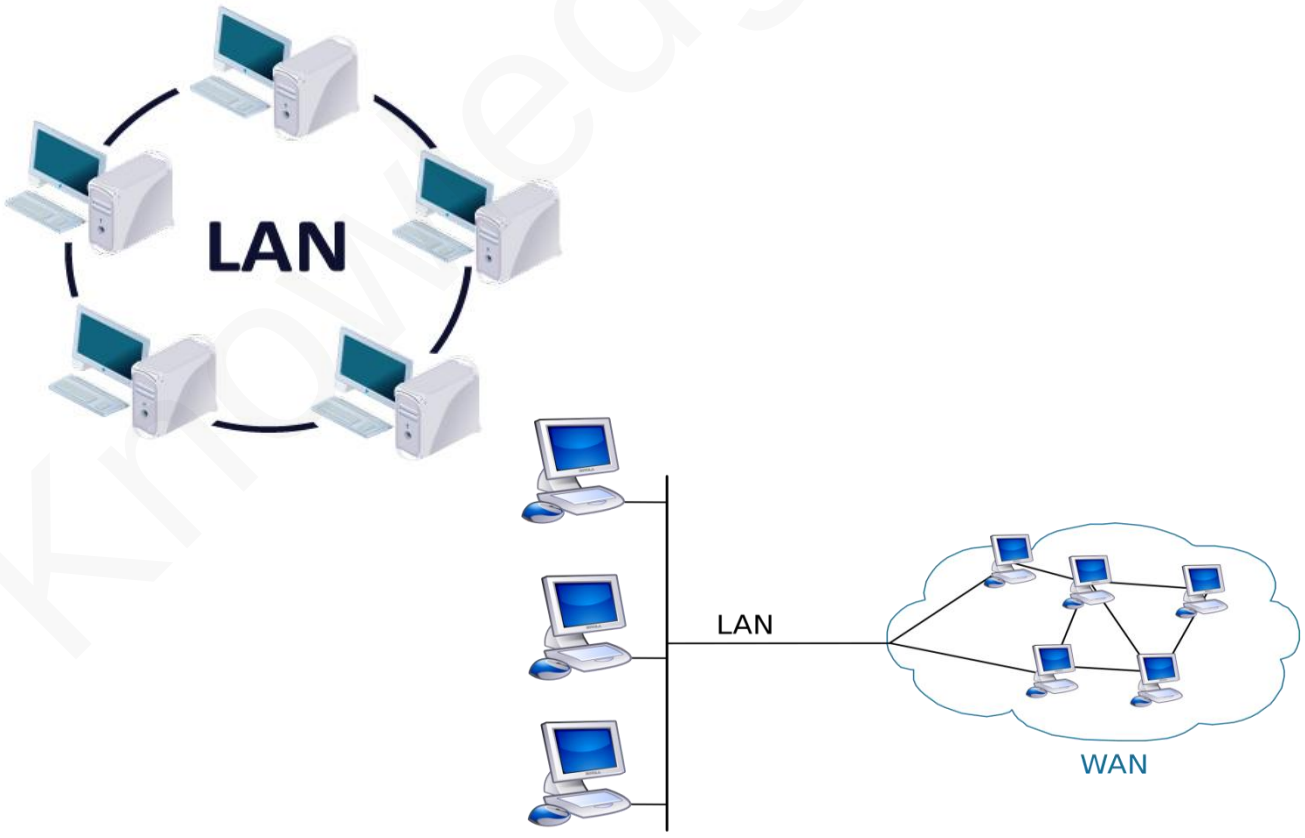
اترنت در مدت زمان کوتاهی به عنوان یکی از تکنولوژی های رایج برای پیاده سازی شبکه های LAN در سطح دنیا مطرح شد.

در بخش های بعدی با انواع تقسیم بندی شبکه ها و تجهیزات مورد استفاده در پیاده سازی آنها که اغلب آنها در شبکه های مبتنی بر اترنت نیز استفاده می شود آشنا می شویم.

۲-۳-۱. شبکه محلی و شبکه گسترده

به طور کلی شبکه ها براساس موقعیت جغرافیایی به دو دسته LAN (Local area network) یا شبکه محلی و WAN(Wide area network) یا شبکه گسترده تقسیم بندی می شوند.

در شبکه های LAN مجموعه ای از دستگاه های موجود در یک موقعیت جغرافیایی محدود، نظیر یک ساختمان، به یکدیگر متصل می گردند. در شبکه های WAN تعدادی دستگاه که از یکدیگر کیلومترها فاصله دارند به یکدیگر متصل می شوند. به عبارتی می توان LAN را زیرمجموعه WAN به حساب آورد، البته ممکن است یک شبکه LAN با سایر شبکه ها ارتباطی نداشته باشد و صرفا برای برقراری ارتباط محلی به کار رود.



شکل ۲-۳

به طور کلی شبکه های LAN نسبت به شبکه های WAN دارای سرعت بیشتری می باشند که البته سرعت شبکه های WAN نیز با توجه به رشد و توسعه تجهیزات مخابراتی، روز به روز در حال افزایش است. امروزه با استفاده از فیبر نوری، انتقال پرسرعت اطلاعات بین شبکه ها و در فاصله های طولانی فراهم شده است.

Ethernet یک تکنولوژی محلی (LAN) می باشد. اکثر شبکه های اولیه در حد و اندازه یک ساختمان بوده و دستگاه ها نزدیک به هم بودند و حداکثر طول کابل شبکه حدود چند صد متر بیشتر نبود.

اخیرا با توجه به توسعه امکانات مخابراتی و رسانه ی انتقال، زمینه استقرار دستگاه های موجود در یک شبکه اترنت با مسافت های چند کیلومتر نیز فراهم شده است.

نکته : شبکه MAN (Metropolitan area network) از LAN ها و تعداد زیادی کامپیوتر در فاصله های زیاد تشکیل شده است. به عنوان مثال شبکه بین شرکت ها یا دانشگاه ها از این نوع می باشند.

Internet شبکه ای جهانی است (World wide network) که شامل LAN، MAN و WAN ها می باشد و در سرتاسر دنیا و توسط صدها هزار میلیون از مردم قابل دسترسی است.

۲-۳-۲. انواع توپولوژی های شبکه

به نحوه ی اتصال گره های شبکه به یکدیگر، پیکر بندی (Topology) شبکه می گویند. در این بخش با انواع توپولوژی های متداول مورد استفاده در شبکه های کامپیوتری آشنا می شویم.

- شبکه خطی یا Bus
- شبکه ستاره‌ای یا Star
- شبکه حلقه‌ای یا Ring
- شبکه Mesh

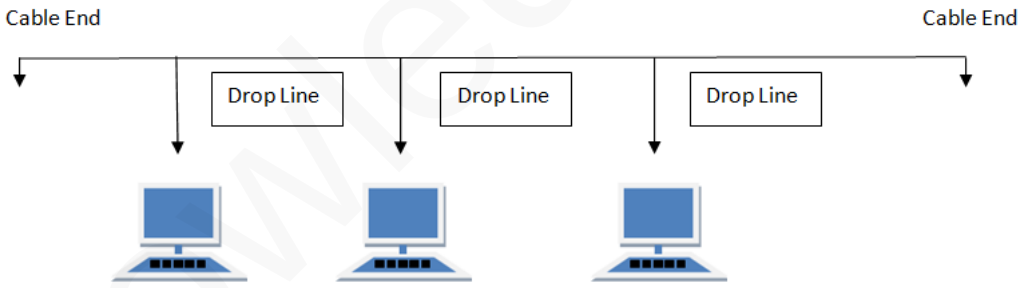
۲-۳-۲-۱. توپولوژی Bus

در این نوع از توپولوژی، معمولا از کابل های هم محور (Coaxial) استفاده می شود. کابل های هم

محور دو نوع کلی دارند، Thick (کلفت) و Thin (نازک).

برای اتصال این کابل به کارت شبکه از کانکتور های BNC و کانکتورهای T شکل استفاده می شود. حداکثر طول کابل در شبکه Thinnet برابر ۱۸۵ متر و در شبکه Thicknet برابر ۵۰۰ متر است. در صورتی که طول کابل بیشتر از مقدار تعریف شده باشد، به علت مقاومت موجود در کابل، جریان ایجاد شده در آن رفته رفته ضعیف شده، به گونه ای که کامپیوتر مقصد قادر به تشخیص جریان نخواهد بود که به این پدیده Attenuation یا تضعیف می گویند.

در این شبکه وقتی کامپیوتری شروع به ارسال داده می کند، جریان وارد کابل شده و در هر دو جهت پیش میرود تا به انتهای کابل برسد، در این فاصله جریان به تمام کامپیوترها می رسد ولی تنها کامپیوتر مقصد از آن استفاده میکند. وقتی جریان به انتهای کابل برسد، برگشت پیدا میکند (Bouncing) و با جریان داخل سیم تداخل پیدا کرده و تصادم (collision) رخ میدهد. برای جلوگیری از بروز چنین مشکلی، در انتهای کابل از Terminator یا خاتمه دهنده استفاده می شود که در واقع مقاومتی است که در Thinnet معادل ۵۰ اهم و در Thicknet برابر ۷۵ اهم میباشد. حداکثر تعداد کامپیوترها در شبکه با توپولوژی Bus برابر ۳۰ عدد میباشد. مشکل اصلی این پیکربندی این است که اگر یک مشکل کوچک در یکی از کانکتورها، ترمیناتورها یا کابل شبکه به وجود بیاید، کل شبکه دچار مشکل می شود.

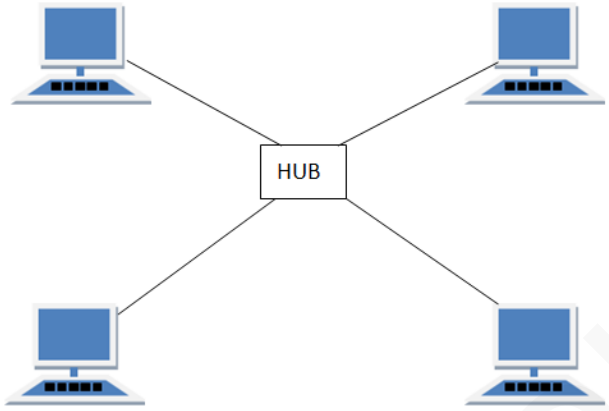


شکل ۲-۴

۲-۲-۳-۲. توپولوژی Star

در این نوع از توپولوژی، از یک وسیله مرکزی برای کابل کشی استفاده میشود که hub نامیده می شود. در یک شبکه با توپولوژی ستاره، هر یک از کامپیوترها توسط یک کابل مجزا به هاب وصل می شوند. اغلب LAN های مبتنی بر تکنولوژی اترنت امروزی از این توپولوژی استفاده می کنند. اولین مزیت توپولوژی ستاره این است که به دلیل اینکه هر کامپیوتر با کابل جداگانه ای با هاب متصل میشود، تحمل خطا در

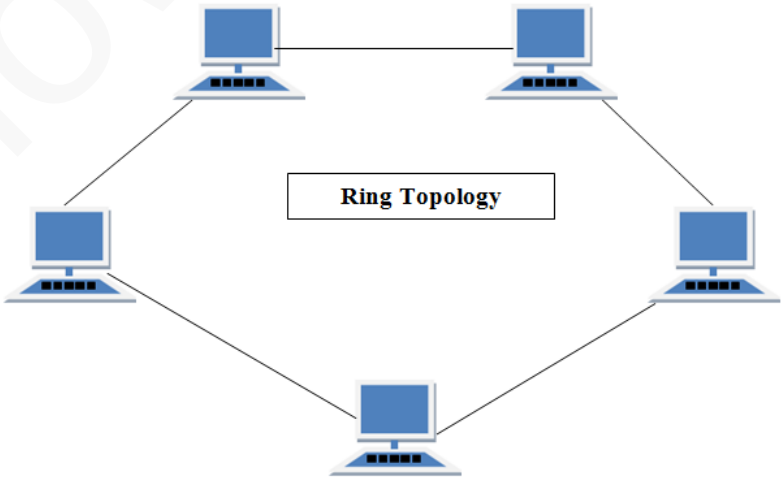
چنین شبکه هایی بالاتر است و دیگر این که اگر یک کابل یا کانکتور دچار مشکل شود، فقط آن سیستمی که با آن کابل یا کانکتور دچار مشکل بوده تحت تاثیر قرار میگیرد و آسیبی به کل شبکه نمیرسد.



شکل ۲-۵

۲-۳-۲-۳. توپولوژی Ring

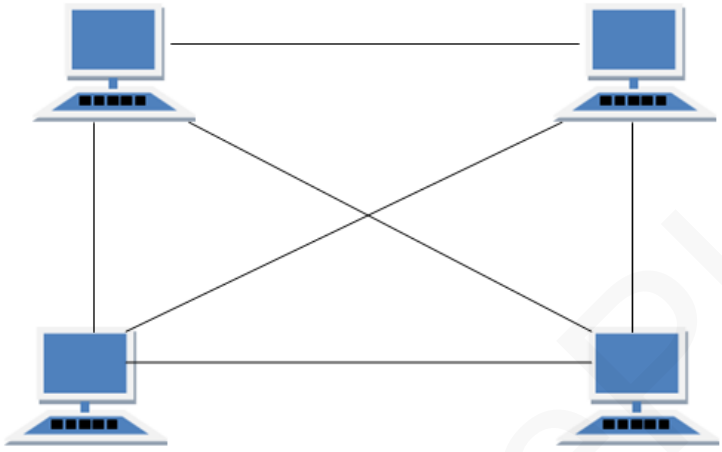
در این نوع از پیکربندی، هر کامپیوتر منطقاً به کامپیوتر همسایه خود متصل است، مانند توپولوژی bus اما با این تفاوت که در این نوع شبکه به جای اینکه دو انتهای کابل شبکه بسته باشد، به همدیگر متصل میشوند و یک حلقه را تشکیل میدهند، به اینصورت که سیگنالی که از یک کامپیوتر تولید شده است، بعد از گذر از تمام کامپیوترها دوباره به کامپیوتر تولید کننده بر می گردد و خودش آن را از شبکه حذف می کند. البته حلقه در این پیکربندی یک ساختار منطقی است نه فیزیکی.



شکل ۲-۶

۲-۳-۲-۴. توپولوژی Mesh

در این نوع از توپولوژی، تمامی کامپیوترها با یکدیگر رابطه مستقیم دارند. توضیح بیشتر در مورد این شبکه‌ها از حوصله این مقاله آموزشی خارج است.



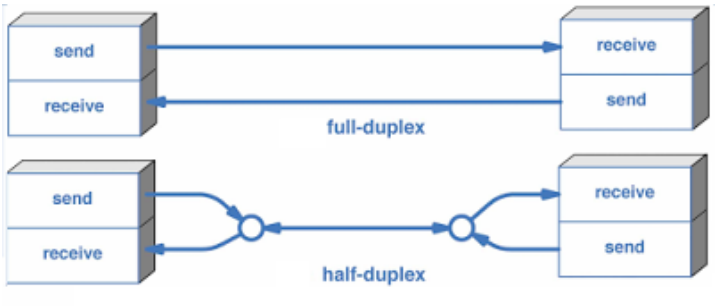
شکل ۲-۷

۲-٤. تجهیزات شبکه

در این بخش در مورد تجهیزات مورد استفاده برای پیاده سازی شبکه های کامپیوتری توضیح می دهیم.
قبل از آن بهتر است با چند اصطلاح که در شبکه های کامپیوتری کاربرد زیادی دارد آشنا شویم :

Half-Duplex و Full-Duplex :

به طور ساده اگر در ارتباط دو گره (مثلا دو کامپیوتر یا دو میکروکنترلر)، هر دو گره قابلیت ارسال و دریافت داشته باشند اما نه به صورت همزمان (در هر لحظه تنها یک گره قابلیت ارسال داده داشته باشد)، آن شبکه در حالت Half-Duplex یا یک طرفه عمل می کند و اگر به طور همزمان امکان ارسال داده هر دو گره وجود داشته باشد، آن شبکه به عنوان Full-Duplex یا دو طرفه عمل می کند. پروتکل I2C به عنوان یک مثال برای ارتباط Half-Duplex و پروتکل SPI به عنوان یک مثال به عنوان ارتباط Full-Duplex می باشد.



شکل ۲-۸

ارتباط Point to Point (Unicast) :

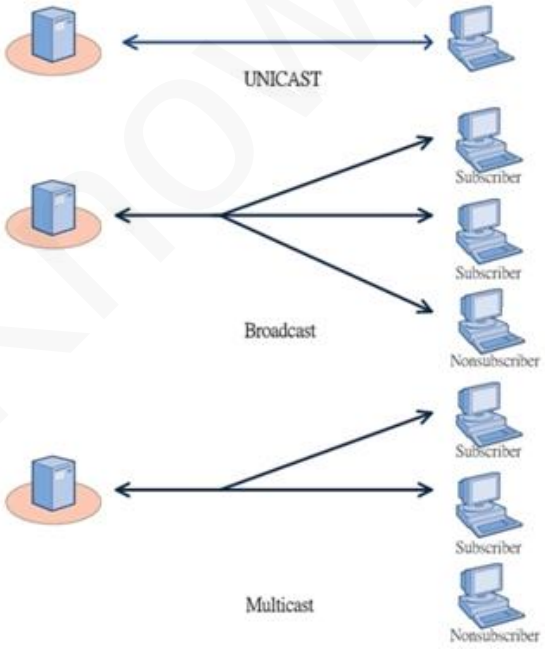
در این نوع از ارتباط، یک گره از شبکه با گره دیگر از شبکه ارتباط برقرار می کند. به عنوان مثال در شبکه I2C با تخصیص آدرس مشخص به پیام، با یک تراشه مشخص ارتباط برقرار می کنیم.

ارتباط Multipoint (Multicast) :

در این نوع از ارتباط، یک گره از شبکه با چندین گره از شبکه ارتباط برقرار می کند. به عنوان مثال با صفر کردن پایه chip select چندین تراشه در پروتکل SPI ، با چندین تراشه ارتباط برقرار می کنیم.

ارتباط Broadcast :

در این نوع از ارتباط، یک گره از شبکه با همه ی گره های دیگر شبکه ارتباط برقرار می کند. در شبکه های اترنت از این نوع ارسال استفاده می شود.



شکل ۲-۹

۲-٤-۱. تکرارکننده (Repeater)

با افزایش طول کابل شبکه، میزان تضعیف سیگنال به علت افزایش امپدانس خط افزایش پیدا می کند. همچنین تاخیر انتشار سیگنال نیز افزایش پیدا می کند. از این رو محدودیتی در حداکثر طول کابل شبکه وجود دارد. به منظور افزایش طول کابل شبکه میتوان از تجهیزاتی که به این منظور تولید شده اند استفاده نمود.

تکرارکننده، سگمنت های متفاوت یک شبکه را به یکدیگر متصل می کند. در این حالت تکرارکننده سیگنال ورودی خود را از یک سگمنت اخذ و با تقویت سیگنال، آن را برای سگمنت های دیگر ارسال می کند. بدین ترتیب با استفاده از تکرارکننده ها میتوان طول مسافت شبکه را افزایش داد.

شکل ۲-۱۰

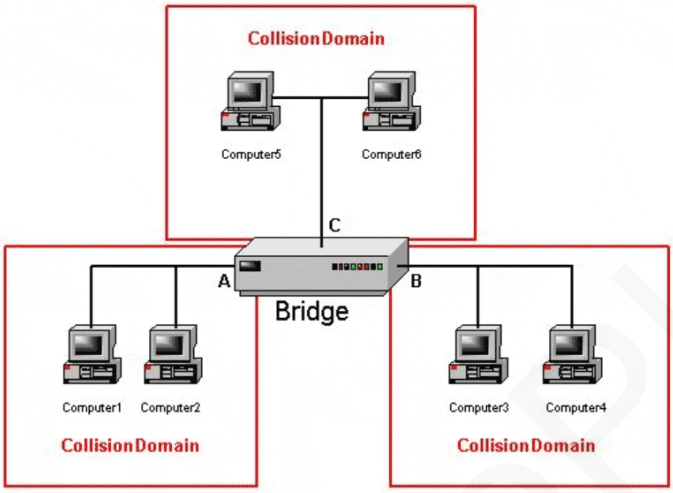
۲-٤-۲. پل (Bridge)

در شبکه هایی که تعداد گره های متصل شده در یک سگمنت زیاد است، باتوجه پروتکل CSMA/CD در هر لحظه فقط یک گره میتواند از شبکه برای انتقال داده های خود استفاده کند، در چنین شرایطی تعداد تصادم در شبکه افزایش پیدا می کند و عملا کارایی شبکه افت پیدا خواهد کرد.

یکی از راه های برطرف نمودن مشکل تراکم در شبکه، تقسیم یک سگمنت به چندین سگمنت است. بدین منظور از تجهیزاتی به اسم Bridge در شبکه استفاده می شود.

Bridge دو یا چند سگمنت را به یکدیگر متصل می کند.

عملکرد Bridge مشابه Repeater است با این تفاوت که قادر به ایجاد نظم در ترافیک شبکه نیز خواهد بود که این کار را توسط فیلتر کردن فریم ها با توجه به آدرس مقصد آنها انجام می دهد.

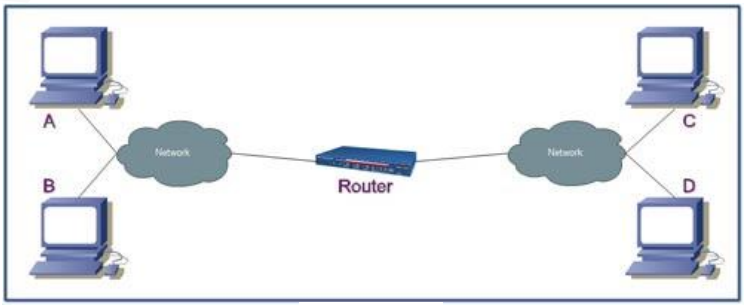


شکل ۲-۱۱

۲-٤-۳. مسیرده (Router)

Bridge در رابطه با ترافیک موجود در یک سگمنت عملیات خاصی را انجام نمی دهد. یکی از مشخصات Bridge ارسال فریم هایی با آدرس Broadcast برای تمام سگمنت های شبکه است که در صورت زیاد بودن تعداد گره های موجود در شبکه، تراکم اطلاعاتی به وجود آمده به مراتب بیشتر از زمانی خواهد بود که تمامی گره ها در یک سگمنت قرار دارند.

Router یکی از تجهیزات پیشرفته در شبکه بوده که قادر به تقسیم شبکه به چندین شبکه منطقی مجزا است. روترها بر اساس پروتکل هایی که مستقل از تکنولوژی خاص در یک شبکه است، فعالیت می نمایند. ویژگی فوق این امکان را برای روتر فراهم خواهد کرد که چندین شبکه با تکنولوژی های متفاوت را به یکدیگر مرتبط نماید.



شکل ۲-۱۲

۲-٤-٤. سوییچ (Switch)

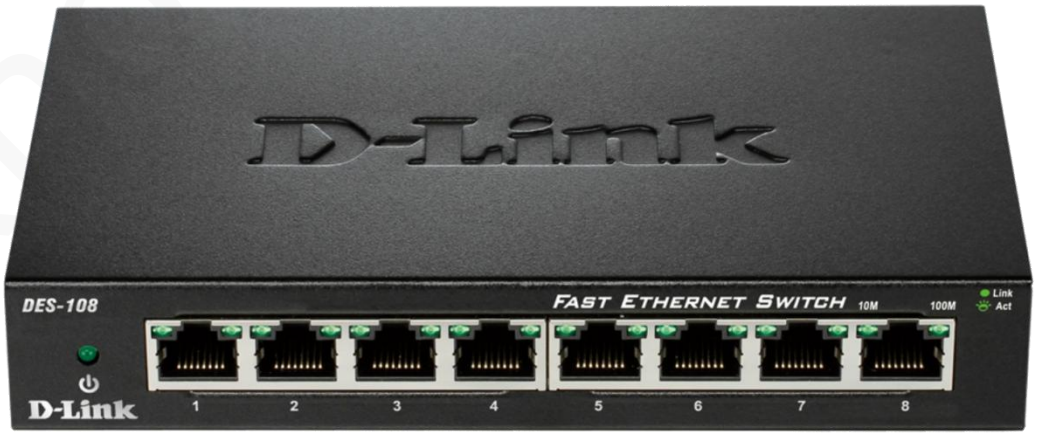
Switch ها مانند Bridge سگمنت های شبکه را به یکدیگر متصل می کنند با این تفاوت عمده که امکان اتصال تعداد زیادی سگمنت (در برخی موارد تا صدها سگمنت) را به یکدیگر فراهم می کند.

در شبکه های مبتنی بر سوییچ، گره ها صرفا با سوییچ ارتباط برقرار کرده و قادر به ارتباط مستقیم با یکدیگر نمی باشند.

سوییچ تمام فریم های در شبکه را دریافت می کند و برخلاف شبکه هایی که در آنها از Bridge استفاده می شود، با توجه به آدرس مقصد آنها صرفا به دریافت کننده ی واقعی ارسال خواهند شد. بدین ترتیب امکان برقراری ارتباط همزمان بین تعداد زیادی گره در شبکه های مبتنی بر سوییچ فراهم خواهد شد و در سرعت های بالاتر، پهنای باند کمتری را اشغال می کند.

بدین ترتیب بعد از ورود سوییچ ها به شبکه های مبتنی بر اترنت (Ethernet Switch)، امکان برقراری ارتباط به صورت Full-duplex در شبکه های اترنت پدید آمد و این مسئله امروزه به عنوان یکی از مهم ترین تحولات ایجاد شده در شبکه های اترنت شناخته می شود. (قبل از ورود سوییچ ها، شبکه های ایتنرنت تنها به صورت Half-duplex قابل پیاده سازی بودند)

شبکه های مبتنی بر سوییچ عاری از از تصادم (Collision-Free) بوده و همزمان با ارسال اطلاعات توسط یک گره، امکان ارسال اطلاعات توسط سوییچ برای گره ی دیگر نیز فراهم خواهد شد. (در مورد تصادم و روش CSMA/CD که در اترنت برای جلوگیری از آن استفاده می شود در فصل چهارم توضیح داده خواهد شد)



شکل ۲-۱۳

سوییچ آدرس MAC تمام دستگاه های متصل به خود را دریافت می کند، سپس هر بسته ای که دریافت کرد، آدرس MAC آن را بررسی کرده و به مقصد مورد نظر ارسال می کند. اگر آدرس MAC آن را تشخیص نداد، آن بسته را به صورت Broadcast منتشر می کند. (در مورد آدرس MAC در ادامه توضیح داده خواهد شد)

نکته : Repeater در لایه اول شبکه یعنی فیزیکی، Switch ها در لایه دوم یعنی لایه پیوند داده و router ها در لایه سوم یعنی شبکه کار می کنند.

نکته : hub و gateway اسامی عمومی می باشند که به تجهیزات مختلف مانند switch ، repeater و... اطلاق می شود.

شکل ۲-۱۴

۲-٤-٥. کارت شبکه (Network Interface Card یا NIC)

کارت شبکه، یکی از مهمترین عناصر سخت افزاری در زمان پیاده سازی یک شبکه کامپیوتری است. هر کامپیوتر موجود در شبکه، نیازمند استفاده از یک کارت شبکه است. کارت شبکه، ارتباط بین کامپیوتر و محیط انتقال را فراهم می نماید. اکثر مادربردهای جدیدی که از آنان در کامپیوترهای شخصی استفاده می شود دارای کارت شبکه onboard می باشند.

هر کارت شبکه دارای یک آدرس فیزیکی (MAC) می باشد. آدرس فوق یک عدد شش بایتی بوده که سه بایت اول آن مشخص کننده سازنده کارت شبکه و سه بایت دوم شماره سریال کارت شبکه است.

شکل ۲-۱۵

نکته : برای اتصال مدار طراحی شده به شبکه Ethernet لزومی به استفاده از کارت شبکه وجود ندارد. در این حالت از روش های Embeded Ethernet استفاده می شود که در فصل های بعدی به طور کامل شرح داده خواهد شد و در انتهای مقاله چند پروژه نیز در همین رابطه انجام خواهد شد.

۲-۵. کابل های شبکه

انواع کابل های متداول مورد استفاده در شبکه عبارت اند از :

- کابل های هم محور (Coaxial)
- کابل های زوج به هم تابیده شده (Twisted Pairs) تحت نام CAT
- فیبر نوری (Optical-Fiber)

۲-۵-۱. کابل های هم محور (Coaxial)

این کابل ها همان کابل آنتن تلویزیون خانگی هستند و در شبکه های با توپولوژی خطی استفاده می شوند. همان طور که قبلا نیز اشاره شد، کابلهای هم محور به دو نوع کلی Thick (کلفت) و Thin (نازک) تقسیم بندی می‌شوند. برای اتصال این کابل به کارت شبکه از کانکتور های BNC و کانکتورهای T شکل استفاده می شود.

۲-۵-۲. کابل های زوج به هم تابیده شده (Twisted Pairs)

این کابل ها مرسوم ترین کابل در ایجاد شبکه های کامپیوتری مانند اترنت هستند. این نوع از کابل در ۷ دسته بندی یا Category که به اختصار CAT نیز گفته می شود وجود دارند. به کابل های زوج به هم تابیده شده بدون محافظ UTP (Unshielded Twisted Pairs) و به کابل های دارای محافظ STP (Shielded Twisted Pairs) گفته می شود و در مکان هایی مانند آسانسور یا کنار کابل های فشار قوی برق که نویز وجود دارد استفاده می شوند.

کانکتور متداول برای این نوع از کابل ها از نوع RJ-45 می باشد. این کانکتور شباهت زیادی به کانکتورهای تلفن (RJ-11) دارد. واژه RJ نیز مخفف Registered Jack می باشد.

این نوع کابل ها دسته بندی های مختلفی دارند که برای سرعت های مختلفی استفاده می شوند که در فصل های بعدی به طور کامل در مورد آنها توضیح خواهیم داد.

۲-۵-۳. فیبر نوری (Optical-Fiber)

فیبر نوری نسبت به دیگر کابل های مورد استفاده در شبکه مانند کابل های CAT از مزایای مختلفی مانند پهنای باند بالاتر، مصونیت بیشتر در برابر نویزهای محیط، ضریب تضعیف کمتر در مسافت های طولانی و امنیت بیشتر بهره می برد و از آن در استانداردهای اترنت با سرعت های زیاد مانند سرعت های 1Gbit/s استفاده می شود.

به طور کلی فیبرهای نوری از نظر ساختمان به دو دسته تقسیم می شوند :

- تک حالتی (Single-mode)
- چند حالتی (Multi-mode)

فیبرهای تک حالتی در مقایسه با چند حالتی دارای پهنای باند بیشتر و ضریب تضعیف کمتری هستند و در مسافت های طولانی استفاده می شوند.

۲-٦. آدرس های شبکه

در بخش های قبلی در مورد مفاهیم پایه در شبکه های کامپیوتری و تجهیزات سخت افزاری لازم برای پیاده سازی شبکه ها صحبت کردیم. در این بخش به موضوع آدرس ها و روش های تقسیم بندی شبکه برای هدایت بسته های داده در شبکه می پردازیم.

۲-٦-۱. آدرس MAC و آدرس IP

هر گره در شبکه دارای دو آدرس است : آدرس سخت افزاری یا آدرس MAC (MAC Address) و آدرس اینترنت یا آدرس IP (IP Address)

آدرس MAC (مخفف Media Access Control یا کنترل دسترسی به رسانه)، یک آدرس سخت افزاری است که به عنوان یک شناسه برای سخت افزار در شبکه عمل می کند و بر روی کارت شبکه یا سخت افزار مورد استفاده به این منظور ثبت می شود.

آدرس IP (مخفف Internet Protocol یا پروتکل اینترنت) که به آدرس میزبان (Host) نیز شناخته می شود، یک آدرس منطقی است که فراهم کننده ی اطلاعات مسیردهی (routing) بسته ها در شبکه است که باعث می شود کامپیوترهای دیگر بتوانند آن را در شبکه پیدا کنند.

آدرس MAC یک شماره ۱۲ رقمی (۶ بایتی) منحصر به فرد در مبنای هگزادسیمال یا مبنای شانزده است. معمولا هر بایت این آدرس توسط دو نقطه یا خط فاصله مانند زیر از هم جدا می شوند :

00:06:35:00:6B:BF یا 00-06-35-00-6B-BF

نکته : از آنجایی که همیشه این آدرس ۱۲ رقمی است، قرار دادن صفر ها اجباری است.

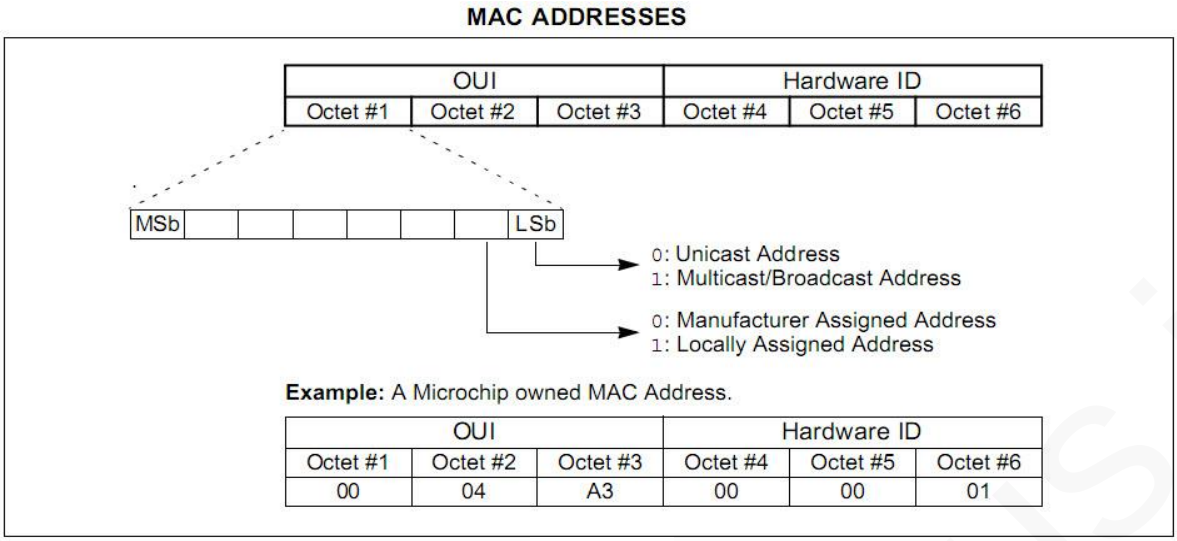
۲۴ بیت یا سه بایت اول آدرس MAC مربوط به OUI (Organizationally Unique Identifier) یا شناسه منحصر به فرد سازمانی و ۲۴ بیت دوم یا سه بایت دوم آن مربوط به شناسه های سخت افزاری است. OUI توسط سازمان IEEE به شرکت های و تولیدکنندگان داده می شود ولی شناسه های سخت افزاری توسط خود تولیدکنندگان تخصیص داده می شود.

ترتیب ارسال مطابق شکل زیر، ابتدا octet1 یا بایت با درجه بالاتر فرستاده می شود و بیت های LSB ابتدا فرستاده می شوند.

آدرس های MAC با LSB بایت octet1 برابر مقدار یک برای بسته های multicast استفاده میشود، به عنوان مثال فریم pause با آدرس 01-80-c2-00-00-01 یک فریم multicast می باشد.

آدرس MAC برابر FF-FF-FF-FF-FF-FF یک آدرس از نوع broadcast است، یعنی این فریم برای همه ی گره های شبکه فرستاده می شود.

در شکل ۲-۱۶ ، علاوه بر نمایش ساختار آدرس MAC، آدرس MAC سخت افزارهای شرکت Microchip (مانند تراشه ENC28J60 که در این مقاله نیز از آن استفاده شده است) نشان داده شده است.



شکل ۲-۱۶

چون آدرس MAC یک آدرس سخت افزاری است، معمولا بر روی کارت شبکه برنامه ریزی (Program) می شود و پس از آن دیگر قابل تغییر نیست؛ بر خلاف آدرس IP که آدرسی منطقی و قابل تغییر است.

آدرس IP منحصر به فرد است و قابلیت استفاده مجدد را ندارد، برخلاف آدرس MAC که قابل استفاده مجدد در شبکه های محلی است.

نکته : بین آدرس IP و آدرس MAC هیچ ارتباطی وجود ندارد.

آدرس IP یک شماره ۳۲ بیتی یا ۴ بایتی باینری یا در مبنای دو می باشد که به عنوان شناسه هر بسته داده ای که در طول شبکه ارسال می شود استفاده می شود. آدرس IP معمولا توسط چهار عدد در مبنای دسیمال یا مبنای ده که هر کدام نشان دهنده ی ۸ بیت است، بیان می شود. به عنوان مثال :

1100 0000 . 1010 1000 . 0001 1011 . 0001 0000
192 . 168 . 27 . 16

در مورد ساختار IP در بخش های بعدی بیشتر توضیح می دهیم.

نکته : باید توجه شود که آدرس IP با پروتکل IP (که در فصل های بعدی در مورد آن توضیح خواهیم داد) اشتباه گرفته نشود. همچنین آدرس MAC با لایه ی MAC که یکی از زیر لایه های لایه پیوند داده در پروتکل اترنت است و در فصل بعدی در مورد آن توضیح می دهیم اشتباه گرفته نشود.

۲-٦-۲. نام دامنه

به نسخه نوشتاری IP که توسط حروف انگلیسی مشخص می شود نام دامنه (domain-name) گفته می شود، به عنوان مثال knowledgeplus.ir ؛ این آدرس به یک IP استاتیک (ثابت) توسط DNS (مخفف Domain Name System) اختصاص داده می شود.

DNS یک منبع اطلاعاتی سلسله مراتبی است که نام دامنه را تبدیل به آدرس IP می کند. در واقع زمانی که در مرورگر آدرس اینترنتی را تایپ می کنید و قصد باز کردن آن صفحه را دارید، درخواستی به DNS ارسال می شود و DNS آدرس IP اختصاص داده شده به نام دامنه را ارسال می کند.

۲-٦-۳. ساختار آدرس IP

آدرس IP دارای دو بخش است، بخش آدرس شبکه (Network Address) که برای همه گره های متصل شده در یکه شبکه یکسان است و بخش پر ارزش تر آدرس را تشکیل می دهد و بخش آدرس گره یا هاست (Host Address) که برای هر گره منحصر به فرد است و بخش کم ارزش تر آدرس را تشکیل می دهد. برای تعیین این دو بخش، پروتکل IP ساختار آدرس های IP را به پنج کلاس یا دسته بندی با نام های A تا E تقسیم می کند (به این روش اصطلاح classful routing گفته می شود). جدول زیر مشخصات این کلاس ها را نشان می دهد.

Network Class	Most Significant Bit(s) in Network Address	Range of Most Significant Byte in Network Address	Number of Bytes in Network Address	Maximum Number of Networks	Number of Bytes in Host Address	Maximum Number of Hosts
A	0	1-126	1	126	3	16 million+
B	10	128-191	2	16,384	2	65,534
C	110	192-223	3	2 million+	1	254
D	1110	224-239	reserved for multicasting			
E	1111	240-255	reserved for future use			

جدول ۲-۱

به عنوان مثال در کلاس A ، بیت سمت چپ (بیت پر ارزش تر) همیشه صفر است. در این کلاس یک بایت (بایت سمت چپ) برای آدرس شبکه در نظر گرفته شده است و با توجه به این که بیت سمت چپ

همیشه صفر است، محدوده آدرس شبکه از 0 تا 126 می باشد. سه بایت دیگر مربوط به آدرس گره های متصل شده به شبکه می باشد، بنابراین امکان اتصال ۱۶ میلیون گره در این شبکه وجود دارد.

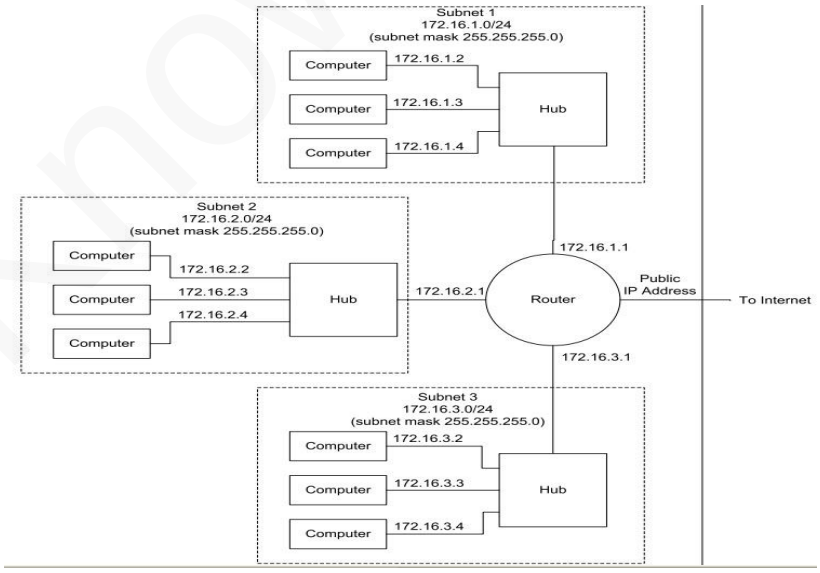
۲-٦-٤. زیرشبکه (Sub-network)

زیرشبکه (Sub-network) به بخش جدا شده از یک شبکه به شکل منطقی از یک شبکه بزرگتر گفته می شود. در واقع توسط این روش، یک آدرس IP به چند بخش تقسیم و به چندین شبکه محلی تخصیص داده می شود. برای این کار بخشی از آدرس IP که مربوط به آدرس گره های شبکه است، به عنوان آدرس subnet در نظر گرفته می شود. به عنوان مثال شرکتی به تعداد 254 آدرس و کمتر از 65,533 آدرس برای شبکه خود نیاز دارد، در اینصورت اگر از کلاس B استفاده کند تعداد زیادی آدرس بدون استفاده خواهد ماند و هدر خواد رفت ولی با استفاده از آدرس های subnet می تواند هر تعداد از آدرس ها را که نیاز دارد استفاده کند و بقیه را به شرکتی دیگر بدهد.

برای تعیین کردن این مسئله که کدام بخش آدرس گره و کدام بخش به عنوان آدرس subnet در نظر گرفته شود، از subnet mask استفاده می شود. از این آدرس برای جدا کردن بخش آدرس شبکه و آدرس subnet از آدرس گره های شبکه استفاده می شود. (به صورت منطقی آدرس IP با subnet mask به اصطلاح AND می شود و حاصل آدرس شبکه و آدرس subnet خواهد بود)

هر کدام از بخش های آدرس subnet mask می توانند یک یا صفر باشند. (255 یا صفر)

شکل زیر کاربرد استفاده از این روش را نشان می دهد.



شکل ۲-۱۷

۲-۶-۵. پروتکل DHCP

DHCP (Dynamic Host Configuration Protocol یا پروتکل پیکربندی پویای هاست)، پروتکلی است که طبق آن آدرس IP گره های متصل به شبکه توسط فرایندی خودکار توسط سرویس دهنده شبکه تعیین می شود.

با توسعه شبکه اینترنت، استفاده از این روش باعث صرفه جویی اقتصادی برای شرکت ها می شود.

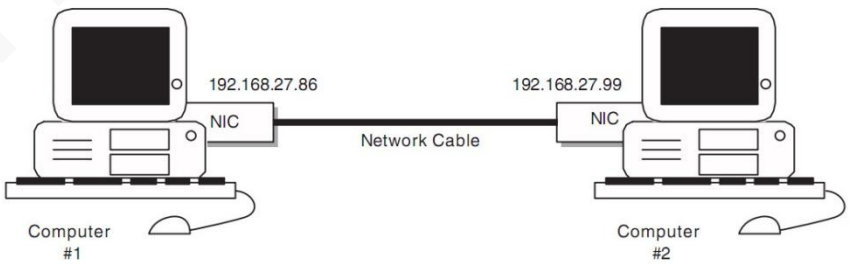
توسط این روش می توان بر اختصاص آدرس های IP به گره های شبکه نظارت و مدیریت کرد و با اتصال هر گره در هر نقطه از شبکه، IP مورد نظر را به آن گره اختصاص داد.

نکته : از آنجایی که برای کار با اترنت لازم است با آدرس های موجود در شبکه و نحوه تخصیص آنها به بسته ها و گره های شبکه آشنایی داشته باشم، در این بخش به صورت کوتاه خلاصه ای از مباحث مربوط به آدرس دهی در شبکه ها را مرور کردیم. علاقه مندان می توانند برای اطلاعات بیشتر در مورد آدرس دهی در شبکه های کامپیوتری به مراجع در این زمینه برای مطالعه بیشتر مراجعه فرمایند.

۲-۷. اتصال شبکه محلی به اینترنت

در این بخش به اختصار اشاره ای در مورد روش های اتصال به شبکه اینترنت خواهیم داشت.

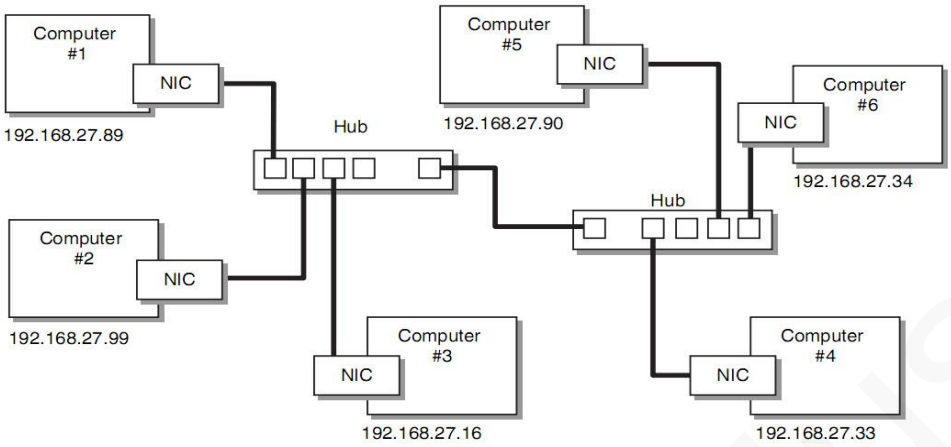
شکل زیر یک شبکه ساده که از دو کامپیوتر تشکیل شده است را نشان می دهد. کامپیوترها از طریق NIC به هم وصل می شوند.



A really simple computer network

شکل ۲-۱۸

شکل زیر شبکه ای که کمی از شبکه بالا پیچیده تر است را نشان می دهد.



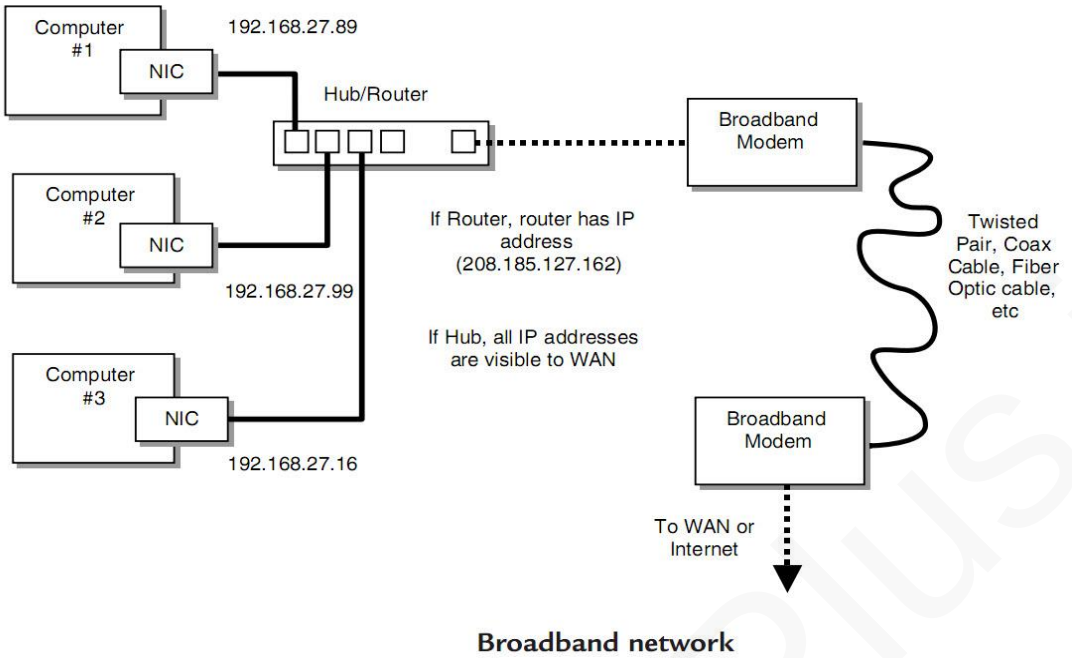
A complex network

شکل ۲-۱۹

برای اتصال hub ها به یکدیگر از درگاهی به نام uplink port که روی hub موجود می باشد استفاده شده است. اگر این درگاه وجود نداشت باید از کابل cross-over برای اتصال hub ها به یکدیگر استفاده شود. (در مورد کابل های straight-through و cross-over در فصل های بعدی به طور کامل توضیح داده خواهد شد)

پس از طراحی شبکه محلی، در صورت لزوم این شبکه باید به شبکه های دیگر مانند LAN های دیگر یا اینترنت متصل شود. برای این منظور روش های مختلفی وجود دارد که در اینجا به دو مورد از آنها اشاره می کنیم.

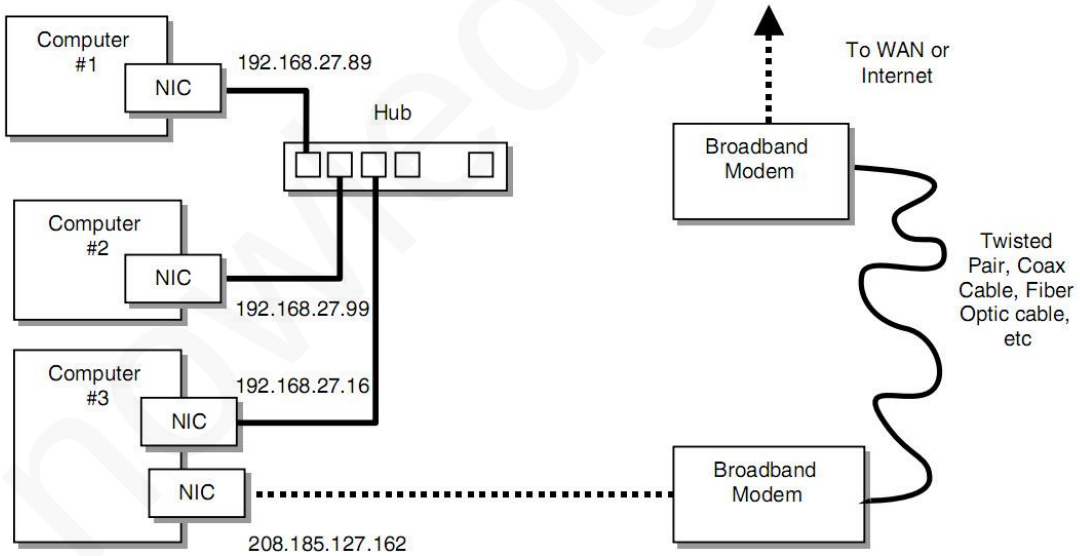
در روش اول باید تجهیزاتی که توسط آن گره های مختلف شبکه به یکدیگر متصل شده اند را (مانند سوییچ) به مودم متصل کنیم. مودم که انواع مختلفی دارد، سیگنال های دریافتی از تجهیزات مختلف را تبدیل به سیگنال آنالوگ مناسب برای ارسال از طریق خطوط انتقالی مانند سیم های تلفن، کابل کواکسیال، DSL یا فیبر نوری می کند و سیگنال های دریافتی از خط انتقال را تبدیل به سیگنال دیجیتال مناسب برای تجهیزات متصل شده به آن می کند. در شکل ۲-۲۰ این مسئله نشان داده شده است.



Broadband network

شکل ۲-۲۰

یک روش دیگر مطابق شکل زیر، استفاده از دو کارت شبکه در یک کامپیوتر برای اتصال به شبکه است. در این روش باید از سیستم عامل مناسب به منظور پشتیبانی از دو کارت شبکه استفاده نمود.



Alternate broadband network

شکل ۲-۲۱

فصل سوم

تاریخچه

۳-۱. مقدمه

در این فصل نگاهی به تاریخچه پیدایش اترنت، نقش محققان و شرکت های مختلف در توسعه اترنت و آینده این تکنولوژی خواهیم داشت.

۳-۲. تاریخچه

پروتکل Ethernet (تلفظ صحیح ایتِرنت) برای اولین بار به صورت تجاری در سال ۱۹۸۰ معرفی شد و سه سال بعد در سال ۱۹۸۳ استانداردی برای آن توسط سازمان IEEE به نام IEEE 802.3 نوشته شد. از زمان نوشتن این استاندارد، در این پروتکل تغییرات مختلفی حاصل شد مانند پشتیبانی از نرخ بیت (bit rate) های بالاتر (از 2.94 Mbit/s به 100Gbit/s)، افزایش پهنای باند، بهبود روش های دسترسی به رسانه (media access control)، واسط فیزیکی مورد استفاده و نوع توپولوژی شبکه از جمله این تغییرات است.

پس از ابداع و معرفی اترنت، این پروتکل به مرور زمان جایگزین تکنولوژی های دیگر مورد استفاده در شبکه های LAN مانند Token ring ، FDDI و ARCNET شد.

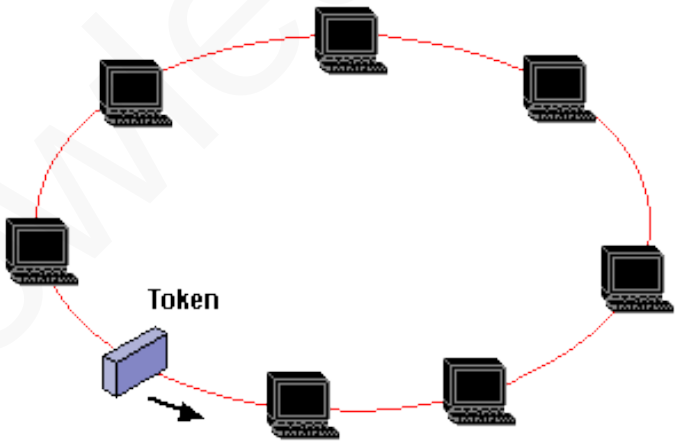
همچنین همزمان با پیشرفت های مهم در زمینه شبکه های کامپیوتری، تجهیزات و دستگاه های مربوط

به شبکه های اترنت نیز همگام با تحولات فوق شده و قابلیت های متفاوتی را در بطن خود ایجاد نمود.

قبل از ابداع اترنت، تکنولوژی های متفاوتی در شبکه های کامپیوتری استفاده می شد. یکی از متداول ترین آنها پروتکل ارتباطی Token Ring که توسط شرکت IBM عرضه شد بود. در شبکه های اترنت به منظور دستیابی به محیط انتقال (media) از فواصل خالی (Gap) تصادفی در زمان انتقال فریم ها استفاده می گردد. شبکه های Token Ring از یک روش پیوسته در این راستا استفاده می نمایند.

در شبکه Token Ring گره ها از طریق یک حلقه منطقی به یکدیگر متصل می گردند. فریم ها صرفا در یک جهت حرکت و پس از طی طول حلقه، کنار گذاشته خواهند شد. در این شبکه به جای استفاده از پروتکل CSMA/CD برای مدیریت دسترسی گره ها به شبکه، از روشی دیگر به نام Token Passing استفاده می شود. در این شبکه دراختیار گرفتن Token (نوع خاصی از یک فریم اطلاعاتی) شرط لازم برای ارسال اطلاعات است.

در روش فوق، در ابتدا یک Token ایجاد می گرد. Token فوق در طول حلقه می چرخد. زمانی که یک گره قصد ارسال اطلاعاتی را داشته باشد، می بایست Token را در اختیار گرفته و فریم اطلاعاتی خود را بر روی محیط انتقال ارسال نماید. زمانی که فریم ارسال شده مجددا به گره ارسال کننده برگشت داده شد (طی نمودن مسیر حلقه)، گره فریم خود را حذف و یک Token جدید را ایجاد و آن را بر روی حلقه قرار خواهد داد.



شکل ۳-۱

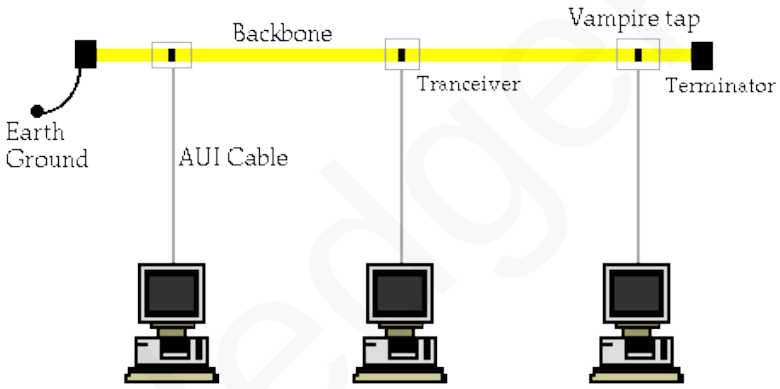
با توجه به توضیحات فوق مشخص می شود که سرعت در این شبکه ها نمیتواند خیلی بالا باشد. سرعت ارسال اطلاعات در شبکه های Token Ring بین ۴ تا ۱۶ مگابیت بر ثانیه است.

از جمله عواملی که باعث موفقیت اترنت بوده است می توان به امکان استفاده با تکنولوژی های جدید،

پایین بودن هزینه نصب و راه اندازی، استفاده ساده و امکان ترکیب با تکنولوژی های جدید اشاره کرد.

در اترنت اولیه از توپولوژی خطی (linear bus) استفاده می شد و ارتباط همه ی گره های شبکه از طریق کابل هم محور (coaxial) انجام می گرفت که توسط تمام گره ها به اشتراک گذاشته می شد. استفاده از تجهیزات حجیم و گران قیمت، سرعت پایین و کوتاه بودن حداکثر طول کابل شبکه از مشکلات اترنت های اولیه بود.

اولین نسخه از Ethernet به نام 10BASE5 از کابل هم محور با قطر 9.5mm به عنوان سیم ارتباطی استفاده می کرد که به نام thick Ethernet یا به طور خلاصه thicknet معرفی می شد. جانشین این ارتباط یعنی 10Base2 از کابل های زوج به هم تابیده شده و کانکتور RJ-58 استفاده می کرد که با نام thin Ethernet یا thinnet معرفی شد. (در فصل بعدی در مورد استانداردهای اترنت توضیح داده خواهد شد)



شکل ۳-۲

یکی از مهم ترین مشکلات استفاده از کابل هم محور برای اتصال تمام گره ها در نسخه های اولیه اترنت مانند 10Base5، قطع شدن کل شبکه در صورت به وجود آمدن مشکل در بخشی از کابل شبکه است. مشکل دیگر مشکل تصادم یا collision می باشد که هنگامی به وجود می آید که یک یا چند گره به شکل همزمان قصد ارسال در شبکه داشته باشند.

یکی از مهمترین تحولات ایجاد شده در شبکه های اترنت، امکان استفاده از تجهیزاتی مانند switch ها در شبکه است.

نسخه های جدیدتر اترنت از سیم های زوج های به هم تابیده شده (twisted pairs)، فیبر نوری و از تجهیزاتی مانند switch برای ایجاد شبکه استفاده می کنند و ارتباط به صورت نقطه به نقطه (point to point) بین گره ها و تجهیزات شبکه در یک توپولوژی ستاره برقرار می شود.

(در فصل بعدی در مورد مسئله تصادم و روش CSMA/CD که در نسخه های قدیمی اترنت برای

جلوگیری از تصادم استفاده می شد توضیح داده خواهد شد)

در ادامه تاریخچه مختصری از پیدایش اترنت و محققان و دانشمندانی که در توسعه اترنت نقش داشته اند آورده شده است.

دراواخر دهه ۱۹۶۰ دانشگاه هاوایی در آمریکا یک شبکه WAN بنام ALOHAnet طراحی کرد. همان طور که میدانیم WAN همان تکنولوژی LAN می‌باشد که در یک منطقه جغرافیایی وسیعتر گسترش یافته است. چون دانشگاه هاوایی وسعت زیادی داشت، مسئولین دانشگاه لازم دیدند که تمام کامپیوترهای پراکنده دانشگاه به هم مرتبط شوند. یکی از ویژگیهای کلیدی شبکه طراحی شده برای این دانشگاه استفاده از روش دسترسی CSMA/CD بود. این شبکه قدیمی پایه و اساس اترنت‌های امروزی شد.

Ethernet برای اولین بار توسط شرکت Xerox Palo Alto Research Center (PARC) در کالیفرنیا در سال های ۱۹۷۳ و ۱۹۷۴ توسعه یافت. این نسخه از اترنت دارای سرعت 2.94Mbit/s بود و برای اتصال ۱۰۰ کامپیوتر در ۱ کیلومتر مورد استفاده قرار می گرفت.

Robert Metcalfe (محقق در زمینه شبکه های رایانه‌ای اهل ایالات متحده آمریکا) و چندی دیگر از محققان شرکت Xerox ، به دنبال روشی برای ارتباط کامپیوترها با چاپگرهای شبکه بودند که یک طرح انتقال سیگنال و کابل‌کشی را اختراع کردند که مجرب به اختراع پروتکل اترنت شد.

در قدیم تصور می‌شد که تابش الکترومغناطیسی از طریق ماده luminiferous ether یا اتردرخشان انتشار می‌یابد. در قرن نوزدهم، وقتی جیمز کلارک ماکسول فیزیکدان انگلیسی، کشف کرد که الکترومغناطیس را میتوان توسط معادله موج توصیف کرد، دانشمندان تصور کردند فضا باید از ماده اتری پر شده باشد تا تابش یا نور بتواند از طریق آن انتشار یابد (پس از انجام آزمایش مایکلسون- مورلی در سال ۱۸۸۷ بود که فیزیکدانان دریافتند تابش الکترومغناطیس در خلا نیز انتشار می‌یابد).

نام Ethernet به پیشنهاد Metcalfe و برگرفته از ماده luminiferous ether که زمانی در علم فیزیک ماده منتشر کننده نور در جهان پنداشته می شد گذاشته شده است.

استفاده از ALOHAnet در پروتکل اترنت که توسط Metcalfe در بخشی از پایان نامه دوره دکترای او پیشنهاد شد، تحولی زیادی در این پروتکل ایجاد کرد.

ایده این کار برای اولین بار در یادداشتی غیررسمی توسط Metcalfe در روز ۲۲ ماه May سال ۱۹۷۳ ارائه شد. او در این یادداشت ابتدا نام این ارتباط را "luminiferous ether" (به معنای آسمان درخشان) گذاشت. در توضیح این نام آمده است :

"omnipresent, completely-passive medium for the propagation of electromagnetic waves"

(حاضر در همه جا، رسانای غیرفعال کامل به منظور انتشار امواج الکترومغناطیسی)

در سال ۱۹۷۶ شرکت Xerox فرم اختراع مربوط به پروتکل اترنت را که در آن نام افرادی مانند Robert Metcalfe ، David Boggs ، Chuck Thacker و Butler Lampson به عنوان مخترعان این ارتباط دیده می شد منتشر کرد.

در سال ۱۹۷۶ برای اولین بار این ارتباط در شرکت Xerox استفاده شد. در همین سال Metcalfe و Boggs مقاله ای رسمی در مورد این پروتکل منتشر کردند.

در ماه June سال ۱۹۷۹ Metcalfe شرکت Xerox را ترک و شرکت 3Com را تاسیس کرد.

Metcalfe شرکت های Digital Equipment Corporation (DEC) ، Intel و Xerox را برای همکاری مشترک برای توسعه اترنت به عنوان یک استاندارد متقاعد کرد. نتیجه ی این همکاری منجر به ارائه استاندارد DIX که مخفف "Digital/Intel/Xerox" شامل سرعت 10Mbit/s ، آدرس ۴۸ بیتی MAC برای فرستنده و گیرنده با مشخصه Ethertype ۱۶ بیتی بود گردید.

در روز ۳۰ ماه September سال ۱۹۸۰ ویژگی های این ارتباط به طور رسمی با عنوان:

"The Ethernet, A Local Area Network. Data Link Layer and Physical Layer Specifications"

(Ethernet ، شبکه ارتباط محلی. مشخصات های لایه ی پیوند داده و لایه ی فیزیکی) ارائه شد.

نسخه دوم اترنت دو سال بعد در ماه November سال ۱۹۸۲ ارائه شد.

همزمان به صورت رسمی نوشتن استاندارد برای این ارتباط شروع شد و در روز ۲۳ ماه June سال ۱۹۸۳ اولین استاندارد رسمی با عنوان IEEE 802.3 از سوی سازمان IEEE ارائه شد.

از همان ابتدا رقابت اترنت با دو نوع پروتکل که به طور گسترده مورد استفاده قرار می گرفت یعنی ارتباط Token Ring و Token Bus شروع شد اما به دلیل قابلیت تطبیق بالای اترنت با نیازهای بازار و شروع به کارگیری زوج سیم های به هم تابیده شده که ویژگی ارزان بودن را دارند، عملا در همان ابتدا اترنت برنده ی این رقابت شد و همه ی بازار را از آن خود کرد و در اواخر دهه ۱۹۸۰ اترنت به تکنولوژی برتر در شبکه ها تبدیل شد.

با بزرگ شدن شرکت 3Com ، این شرکت اولین محصول خود را در این زمینه که یک کارت شبکه مبتنی بر ارتباط اترنت با سرعت 10Mbit/s بود در ماه March سال ۱۹۸۱ ارائه کرد. در ادامه در همین سال شروع به ارائه آداپتورهایی برای محصولات PDP-11s و VAXes شرکت DEC و کامپیوترهای با

چند گذرگاه (Multibus-based) شرکت های Intel و Sun Microsystems کرد.

بلافاصله شرکت DEC برای ایجاد شبکه هایی یکپارچه با محصولات خود، شروع به ارائه گذرگاه Unibus مبتنی بر آداپتورهای اترنت کرد که تا سال ۱۹۸۶ تعداد گره های شبکه این شرکت به ده هزار گره رسید و در آن زمان به یکی از بزرگترین شبکه های کامپیوتری دنیا تبدیل گشت.

کارت ها شبکه مبتنی بر اترنت برای کامپیوترهای شرکت IBM در سال ۱۹۸۲ ارائه شد و تا سال ۱۹۸۵ شرکت 3Com توانست صد هزار عدد از این کارت ها را به فروش برساند.

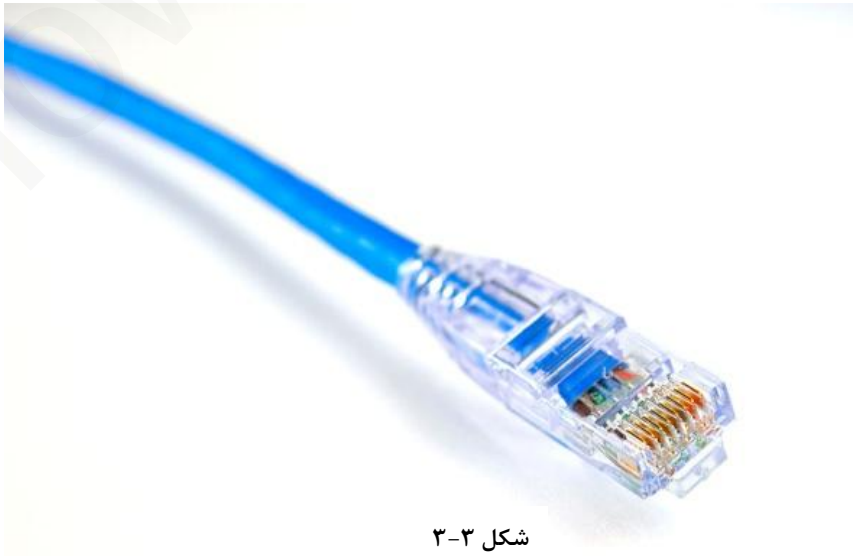
در اوایل دهه ۱۹۹۰ اترنت به قدری رایج شد که وجود آن برای کامپیوترهای مدرن به امری ضروری بدل گشت و به تدریج پورت های اترنت بر روی PC ها و workstation ها قرار گرفت. این روند با معرفی استاندارد 10BASE-T و کانکتورهای کوچک ماژولار (مانند RJ-45) سرعت زیادی گرفت و بعد از مدتی پورت های اترنت حتی روی مادربردهای ارزان قیمت نیز دیده می شد.

از این نقطه به بعد مشخصات اترنت برای پشتیبانی از پهنای باند بیشتر و نیازهای بازار شروع به تغییر کرد.

امروزه اترنت علاوه بر کامپیوترها، بر روی وسایل شخصی مانند تلفن های همراه، تبلت ها و ... استفاده می شود. استفاده از اترنت در کاربردهای صنعتی رو به گسترش است و در حال جایگزینی سیستم های ارتباطی قبلی است.

فروش تجهیزات مرتبط با اترنت از سال ۲۰۱۰ حدود ۱۶ بیلیون دلار در هر سال بوده است.

شکل زیر کانکتور ماژولار 8P8C را که اغلب به نام RJ-45 شناخته می شود همراه با کابل Cat 5 نشان می دهد.



شکل ۳-۳

شکل زیر واحد فرستنده/گیرنده (transceiver) و کانکتورهای مورد استفاده در نسخه های اولیه اترنت مانند 10Base5 و 10Base2 را نشان می دهد.

شکل ۳-۴

با گذر زمان، تراشه هایی با واحدهای کنترل کننده اترنت (Ethernet controller) و واحد فرستنده/گیرنده (transceiver) عرضه شدند که تحول زیادی در تجهیزات شبکه نسبت به تصویر قبلی ایجاد شد.

شکل زیر یک تراشه کنترل کننده اترنت را که در شبکه های اترنت امروزی استفاده می شود را نشان می دهد.

شکل ۳-۵

شکل زیر کارت شبکه ای را نشان می دهد که در دهه ۱۹۹۰ معرفی شد که هم از کابل های هم محور (کانکتور BNC سمت چپ) پشتیبانی می کرد هم از کابل های CAT .(کانکتور RJ-45 سمت راست)

شکل ۳-۶

همان طور که در ابتدای فصل اشاره شد یکی از مهم ترین تحولات به وجود آمده در اترنت، امکان استفاده از توپولوژی ستاره توسط تجهیزات مختلف شبکه مانند switch ها می باشد.

اولین دستگاه switch (به نام Ethernet Switch) برای ایجاد شبکه اترنت و توپولوژی ستاره توسط شرکت Kalpana در Silicon Valley طراحی و ساخته شد.

در نسخه های اولیه اترنت که با استفاده از repeater توپولوژی ستاره پیاده سازی می شد، همه ی گره های شبکه باید در سرعت یکسان کار می کردند اما در repeater های جدید تر و همچنین تجهیزات پیشرفته تر مانند switch ها، امکان استفاده از گره های با سرعت های مختلف (شبکه های Mixed-speed) به وجود آمده است.

اترنت با یک روند ثابت همچنان به رشد خود ادامه می دهد. امروزه شاهد ارائه استانداردهای اترنت با سرعت های چند ده گیگابیت بر ثانیه و مسافت های چند صد متر تا چند کیلومتر هستیم.

پس از گذشت حدود سی سال از عمر شبکه های اترنت، استانداردهای مربوطه ایجاد و برای عموم متخصصین شناخته شده هستند و همین امر نگه داری و پشتیبانی از شبکه های اترنت را آسان نموده است. اترنت با صلابت به سمت افزایش سرعت و بهبود کارائی و عملکرد گام برمی دارد.

فصل چهارم

اترنت

٤-١. مقدمه

در ابتدای این فصل واژگان پرکاربرد مورد استفاده در شبکه های کامپیوتری و اترنت آورده شده است. در بخش اول این فصل در مورد مدل های مرجع OSI و TCP/IP که دو مدل رایج برای توصیف مفهومی عملکرد شبکه های کامپیوتری و دسته بندی پروتکل ها با توجه به کارکرد هر یک در یک لایه می باشد می پردازیم. در بخش بعدی با توجه به کاربرد مدل TCP/IP در شبکه های اینترنت و اترنت، به بررسی پروتکل های رایج مورد استفاده در این مدل می پردازیم که پروتکل اترنت نیز در این بخش جایگاهش در مدل TCP/IP و چگونگی ارتباط آن با پروتکل های دیگر تا حدودی مشخص می شود. در ادامه ی این بخش در مورد مفهوم سرویس دهنده و سرویس گیرنده و همچنین مفهوم سوکت و پورت صحبت می شود که در برنامه نویسی و درک چگونگی فرایند برقراری ارتباط بین سرویس ها یا گره های مختلف در شبکه در پروتکل های لایه های بالایی مدل TCP/IP ضروری می باشد.

در بخش بعدی این فصل، به طور مشخص وارد بحث در مورد پروتکل اترنت می شویم. در بخش اول ابتدا معرفی کوتاهی در مورد اترنت برای آشنایی اولیه انجام می دهیم. در بخش بعدی در مورد استاندارد های مختلف منتشر شده اترنت تحت عنوان IEEE 802.3 و بررسی مشخصات نسخه های رایج اترنت توضیح می دهیم.

در ادامه کابل های رایج مورد استفاده برای ایجاد شبکه های اترنت یا متصل شدن به شبکه های موجود

را معرفی و در مورد انواع هریک از کابل ها و انتخاب مناسب کابل با توجه به مشخصات و سرعت استاندارد پیاده سازی شده توضیح می دهیم.

در بخش بعدی در مورد کانکتور متداول مورد استفاده در شبکه های اترنت، یعنی کانکتور RJ-45 توضیحاتی می دهیم و در بخش استاندارد های TIE/EIA در مورد نحوه ی اتصال کابل شبکه به کانکتور توضیح می دهیم.

در بخش بعدی به طور خلاصه و برای آشنایی، چند مثال در مورد پیاده سازی استانداردهای معرفی شده برای اترنت را بررسی می کنیم.

بخش بعدی که از مهم ترین بخش های این فصل می باشد و برای پیاده سازی اترنت دانستن آن لازم می باشد، ساختار فریم های اترنت است. در این بخش ساختار فریم های متداول که در شبکه های اترنت برای ارسال و دریافت داده مورد استفاده قرار می گیرند بررسی خواهد شد. در فصل هفتم نیز که در مورد تراشه ENC28J60 توضیح می دهیم، اشاره ای به ساختار فریم های اترنت خواهیم داشت که با توجه به توضیحات ارائه شده در این بخش، در آنجا تنها به چند نکته اشاره شده در برگه اطلاعاتی این تراشه بسنده می کنیم.

در ادامه این فصل نیز در مورد جزئیات بیشتری در مورد پروتکل اترنت مانند نوع ارتباط half-duplex و full-duplex در اترنت، توصیف زیر لایه های تعریف شده برای اترنت، کدهای مورد استفاده در استاندارد های مختلف اترنت و بحث زمان بندی فریم های اترنت و قابلیت Auto-Negotiation در اترنت و کاربرد آن می پردازیم.

در انتها نیز به معرفی دو نوع کاربرد رایج اترنت تحت عنوان Embeded Ethernet یا اترنت تعبیه شده و Industrial Ethernet یا اترنت صنعتی می پردازیم.

۲-٤. راهنمای واژگان مورد استفاده در شبکه های Ethernet

واژه	توضیحات
CRC	مخفف Cyclic Redundancy Check الگوریتمی برای بررسی بیت ها برای محاسبات FCS فریم های اترنت و کلید جدول ترکیبی (hash table) برای فیلتر کردن بسته های دریافتی
DA	مخفف Destination Address آدرس مقصد بسته های اترنت که شامل شش octet می باشد
ESD	مخفف End-of-Stream Delimiter (تعیین کننده پایان جریان اطلاعات) در سرعت 100Mb/s ESD بعد از FCS برای تعیین پایان فریم فرستاده می شود
FCS	مخفف Frame Check Sequence (ترتیب بررسی فریم) چهار octet که در آخر هر فریم قرار می گیرد و حاوی بیت های چک کردن اطلاعات خطایابی می باشد
IP	مخفف Internet Protocol ، دو نسخه IPv4 و IPv6
LAN	مخفف Local Area Network به معنای شبکه محلی یا Large Area Network به معنای شبکه گسترده
MAC	مخفف Media Access Control (کنترل دسترسی به رسانه) واحدی که وظیفه پیاده سازی توابع کنترل دسترسی به رسانه طبق مشخصات اترنت را دارد
MAC Address	یک شماره ی شامل شش octet برای تعیین آدرس لایه ی فیزیکی گره های متصل به شبکه اترنت، هر فریم در اینترنت حاوی آدرس منبع و آدرس مقصد است، به هر دوی این آدرس ها MAC address اطلاق می گردد.
MDI	مخفف Medium Dependent Interface به معنای واسط وابسته به رسانا یا مخفف Management Data Input به معنای مدیریت داده های ورودی
MDO	مخفف Management Data Output به معنای مدیریت داده های خروجی
MDIO	مخفف Management Data Input/Output به معنای مدیریت داده های ورودی/خروجی
MII	مخفف Media Independent Interface به معنای واسط بدون وابستگی به رسانا، واسط استاندارد ۴ بیتی بین MAC و PHY برای انتقال فریم های TX و RX ، در سرعت MII 10Mb/s در فرکانس 2.5MHz عمل می کند و در سرعت 100Mb/s در فرکانس 25MHz عمل می کند
MIIM	مخفف MII Management (مدیریت MII)، دسته ای از سیگنال های باند کناری (Sideband) MII برای دسترسی به رجیسترهای PHY
OUI	مخفف Organizationally Unique Identifier (شناسه منحصر به فرد سازمانی)، شامل سه octet بالایی MAC address که معمولا برای هر سازمان یا شرکتی متفاوت است، به عنوان مثال برای محصولات شرکت Microchip برابر 00-04-A3h می باشد.
Octet	در مجموعه واژگان اترنت به معنای ۸ بیت یا یک بایت است.
Packet Buffer	حافظه فیزیکی یا مجازی است که همه ی بسته های ارسالی و دریافتی در آن ذخیره می شود
PHY	واحدی که لایه ی فیزیکی اترنت را پیاده سازی می کند

RAM	Random Access Memory حافظه با دسترسی تصادفی (حافظه غیر دائم)
Receive Buffer	بخش منطقی packet buffer برای ذخیره سازی فریم های دریافتی
RX	مخفف Receive به معنای دریافت
SA	مخفف Source Address (آدرس منبع)، معادل آدرس MAC فرستنده
SFD	مخفف Start Frame Delimiter (تعیین کننده شروع فریم)، یک octet که مشخص کننده ی شروع فریم است
SPI	مخفف Serial Peripheral Interface یا واسط سریال سخت افزارهای جانبی
SSD	مخفف Start-of-Stream Delimiter (تعیین کننده شروع جریان اطلاعات)، در سرعت 100Mb/s اولین octet بخش preamble (آغاز) به عنوان SSD شناخته می شود و کدینگ آن از بقیه بخش های preamble متفاوت است
Station Address	معادل آدرس MAC
Transmit Buffer	بخش منطقی packet buffer برای ذخیره سازی فریم های ارسالی
TX	مخفف Transmit به معنای ارسال
RMII	مخفف Reduced Media Independent Interface به معنای واسط مستقل از رسانا ی کاهش یافته، نسخه ی ۲ بیتی MII
SMII	مخفف Serial Media Independent Interface به معنای واسط مستقل از رسانای سریال، نسخه یک بیتی MII
NRZI	مخفف Non-Return-to-Zero Inverted ، کد باینری که در آن سطح منطقی یک برابر گذر (transition) سیگنال و صفر نشان دهنده ی عدم گذر سیگنال است

جدول ۴-۱

٤-٣. مدل مرجع OSI و TCP/IP

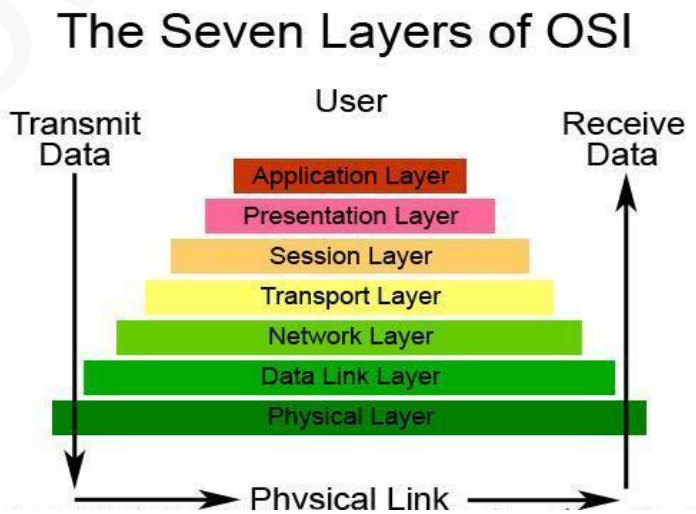
در بحث شبکه های کامپیوتری، یکی از بهترین روش ها به منظور تشریح مفهومی نحوه کارکرد و پیاده سازی شبکه، استفاده از مدل های مرجع ارائه شده در این زمینه مانند OSI و TCP/IP می باشد.

این مدل ها را می توان در اکثر کتاب ها یا مقالاتی که در مورد پروتکل های شبکه صبحت می شود مشاهده کرد. توسط این مدل های مفهومی، به راحتی و بدون توجه به جزئیات می توان سخت افزار و نرم افزار مورد نظر را طراحی کرد.

در این بخش ابتدا با این دو مدل و لایه های مختلف آن آشنا شده و در بخش های بعدی جایگاه اترنت در این مدل ها را بررسی می کنیم.

٤-٣-١. مدل OSI

مدل OSI (Open System Interconnection) یک مدل مرجع برای ارتباط بین دو کامپیوتر می باشد که در سال ۱۹۸۰ طراحی گردیده است. هر چند امروزه تغییراتی درآن به وجود آمده اما هنوز هم کاربردهای فراوانی در اینترنت و به خصوص در معماری پایه شبکه دارد. این مدل بر اساس لایه بندی قراردادهای برقراری ارتباط که همزمان روی دو سیستم مرتبط اجرا شده اند پایه ریزی شده است که این امر بسیار سرعت و دقت ارتباط را افزایش می دهد و این قراردادها به صورت طبقه طبقه در هفت لایه تنظیم شده اند که در ادامه بررسی خواهند شد .



شکل ۴-۱

این لایه ها بین لایه بالاتر و پایین تر خود قرار گرفته و به آنها سرویس می دهند.

لایه های OSI در حقیقت یک مدل کاملا فرضی برای درک فضای شبکه هستند و در حقیقت چنین لایه هایی وجود ندارند.

بالاترین لایه، لایه هفت (لایه ی کاربرد یا Application) و پائین ترین لایه، لایه یک است (لایه ی فیزیکی یا physical). در زمان ارسال داده از یک کامپیوتر به کامپیوتر دیگر، داده ها حرکت خود را از لایه هفتم آغاز نموده و پس از تبدیل به بخش (segment) ، داده نما (Datagram) ، بسته (Packet) و قالب (Frame)، در نهایت از طریق محیط انتقال (مثلا کابل) برای کامپیوتر مقصد ارسال می گردند.

هر لایه اطلاعات را از لایه ی بالاتر خود دریافت میکند و آن را مورد پردازش قرار میدهد و آن را به لایه ی پایین تر خود می‌دهد.

هر لایه قبل از ارائه ی بسته ی اطلاعات به لایه ی بعدی، اطلاعات دریافتی از لایه بالایی را توسط اضافه کردن سرآمد (Header) مربوط به خود، بسته بندی (Encapsulation) می کند و به لایه های پایین تر ارسال می کند. سرآمد ها معمولا شامل آدرس مبدا و مقصد، اطلاعات مربوط به کنترل خطا، شناسایی پروتکل و پارامترهای پروتکل می باشند.

نکته : به فرایند اضافه کردن سرآیند مخصوص هر پروتکل به اطلاعات دریافت شده از لایه های بالایی، بسته بندی (Encapsulation)گفته می شود. اطلاعات دریافت شده برای رفتن به لایه های بالاتر، بخش سرآیند مربوط به هر پروتکل جدا می شود و به لایه های بالاتر ارسال می شود که به این عمل از بسته خارج کردن (Decapsulation) گفته می شود. گاهی اوقات ممکن است از این دو عمل به عنوان Assemble و Disassemble نیز یاد شود. در این مورد در بخش های بعدی به همراه مثال توضیح بیشتری داده خواهد شد. همچنین در بخش عملی پروژه با نحوه پیاده سازی عملی فرایند بسته بندی آشنا خواهید شد.

اکنون به توضیح هر لایه میپردازیم.

۴-۳-۱-۱. لایه Application (کاربرد)

- ارائه سرویس های شبکه به برنامه ها (نظیر پست الکترونیکی ، ارسال فایل ها و....)
- تشخیص زمان لازم به منظور دستیابی به شبکه

لایه کاربرد شامل مجموعه پروتکل‌هایی مانند وب (HTTP) ، انتقال فایل (FTP) ، انتقال خبر (NNTP)، و پست الکترونیک (POP و قرارداد ساده نامه‌رسانی) است. برنامه‌های کاربردی نظیر مرورگرهای اینترنتی، برنامه‌های مدیریت ایمیل و... در این لایه قرار می‌گیرند و به صورت کلی واسط بین کاربر و دنیای شبکه می‌باشد.

این لایه با سیستم عامل و برنامه های کاربردی ارتباط دارد. کاربران با استفاده از نرم افزارهای کاربردی متفاوت قادر به انجام عملیات مرتبط با شبکه خواهند بود، به عنوان مثال میتوانند اقدام به ارسال فایل، پیام و.... نمایند.

بصورت خلاصه در تعریف کارایی این لایه می توان گفت که این لایه رابط بین کاربر و شبکه است.

۴-۳-۱-۲. لایه Presentation (ارائه)

- □ ایجاد اطمینان لازم در رابطه با قابل استفاده بودن داده برای سیستم دریافت کننده
- □ رمزنگاری داده

این لایه داده ها را از لایه ی کاربرد دریافت میکند و ساختار آنها را به گونه ای تبدیل خواهد کرد که توسط لایه های پایین تر قابل استفاده باشد.

همچنین برعکس این عمل را نیز انجام میدهد، یعنی زمانی که اطلاعاتی از لایه نشست یا Session به این لایه وارد می شود، این اطلاعات را به گونه ای تبدیل می کند که لایه کاربرد بتواند آنها را درک کرده و متوجه شود. دلیل اهمیت این لایه این است که نرم افزارهای اطلاعات را به شیوه ها و اشکال مختلفی نسبت به یکدیگر بر روی شبکه ارسال می کنند. برای اینکه ارتباطات در سطح شبکه ها بتوانند به درستی برقرار شوند، بایستی اطلاعات را به گونه ای ساختاردهی کنید که برای همه انواع شبکه ها و استانداردها قابل فهم باشد. بطور خلاصه وظیفه اصلی این لایه قالب بندی اطلاعات یا Formatting اطلاعات است. معمولا فعالیت هایی نظیر رمزنگاری و فشرده سازی از وظایف اصلی این لایه محصوب می شود.

۴-۳-۱-۳. لایه Session (جلسه)

- □ ایجاد، مدیریت و خاتمه ارتباط برقرار شده بین برنامه ها

□ برقراری جلسه

□ مدیریت جلسه

□ ارائه سرویس‌های کنترل دیالوگ، مدیریت نشانه، همگام سازی

□ خاتمه دادن به جلسه

در این لایه بر کارهایی از قبیل زمان ارسال و دریافت بسته ها، مقدار رسیده و مقدار مانده از بسته ها نظارت می شود که به مدیریت بسته ها بسیار کمک می کند.

وقتی داده ها به شکلی قابل درک برای ارسال توسط شبکه در آمدند، ماشین ارسال کننده بایستی یک Session با ماشین مقصد ایجاد کند. منظور از Session دقیقا شبیه ارتباطی است که از طریق تلفن انجام می شود، شما برای ارسال اطلاعات از طریق تلفن حتما بایستی با شخص مورد نظرتان تماس برقرار کنید. اینجا زمانی است که لایه نشست وارد کار می شود. این لایه وظیفه ایجاد، مدیریت و نگهداری و در نهایت خاتمه یک Session را با کامپیوتر مقصد بر عهده دارد. نکته جالب در خصوص لایه نشست این است که بیشتر با لایه کاربرد مرتبط است تا لایه فیزیکی. شاید این طور فکر شود که بیشتر Session ها بین سخت افزارها و از طریق لینک های شبکه ایجاد می شوند اما در اصل این نرم افزارهای کاربردی هستند که برای خود Session با نرم افزار مقصد ایجاد میکنند. اگر کاربری از تعدادی نرم افزار کاربردی استفاده میکند، هر کدام از این نرم افزاها به خودی خود می توانند یک Session با نرم افزار مقصد خود برقرار کنند که هر کدام از این Session ها برای خود یک سری منابع منحصر به فرد دارند.

۴-۳-۱-۴. لایه Transport (انتقال)

□ شکستن داده‌ها برای لایه‌های پایین تر

□ تعیین سرویس‌های لایه جلسه

□ ایجاد اطمینان لازم در رابطه با قابل استفاده بودن داده برای سیستم دریافت کننده

□ توافق در رابطه با گرامر انتقال داده برای لایه Application

□ رمزنگاری داده

در این لایه قبل از ارسال اطلاعات، یک بسته به سمت مقصد فرستاده می شود تا مقصد را برای دریافت اطلاعات آماده کند. همچنین این لایه وظیفه تکه تکه کردن بسته ها، شماره گذاری آنها و ترتیب و نظم

دهی آنها را بر عهده دارد که البته بسته ها در طرف گیرنده دوباره در همین لایه نظم دهی و قابل استفاده برای لایه های بالاتر خواهند شد.

لایه انتقال وظیفه نگهداری و کنترل ریزش اطلاعات یا Flow Control را بر عهده دارد. اگر به خاطر داشته باشید سیستم عامل ویندوز به شما این اجازه را میدهد که همزمان از چندین نرم افزار استفاده کنید. خوب همین کار در شبکه نیز ممکن است رخ بدهد، چندین نرم افزار بر روی سیستم عامل تصمیم میگیرند که بصورت همزمان بر روی شبکه اطلاعات خود را منتقل کنند. لایه انتقال اطلاعات مربوط به هر نرم افزار در سیستم عامل را دریافت و آنها را در قالب یک رشته تکی در می آورد. همچنین این لایه وظیفه کنترل خطا و همچنین تصحیح خطا در هنگام ارسال اطلاعات بر روی شبکه را نیز بر عهده دارد. به صورت خلاصه وظیفه لایه انتقال این است که از رسیدن درست اطلاعات از مبدا به مقصد اطمینان حاصل کند، انواع پروتکل های اتصال گرا یا Connection Oriented و غیر اتصال گرا یا Connection Less در این لایه فعالیت می کنند.

۴-۳-۱-۵. لایه Network (شبکه)

- ارائه ارتباط و مسیر انتخابی برای دو سیستم
- پاسخ به سوالات متعددی نظیر نحوه ارتباط سیستم های موجود در segment های متفاوت شبکه
- آدرس های مبداء، مقصد، Subnet و تشخیص مسیر لازم
- پروتکل های IP و IPX در این لایه استفاده می گردند
- کنترل عملکرد زیر شبکه
- مسیر یابی (Routing)
- کنترل گلوگاه‌ها
- کیفیت سرویس دهی
- پیوستن به شبکه‌های ناهمگن

و اما پیچیده ترین لایه یعنی لایه شبکه که در آن قراردادهای شبکه بندی تعریف شده است. وظیفه این لایه انتقال تکنولوژی برقراری ارتباط برای دیگر شبکه های مستقل است که این امر این امکان را به OSI می دهد که بتواند در زیر شبکه های مختلف فعالیت کند.

تصور کنید که قرار است برای یکی از دوستانتان یک نامه بلند بالا بنویسید، اما کاغذ به اندازه کافی ظرفیت برای درج مطالب شما ندارد، خوب شما میتوانید چندین کاغذ را استفاده کرده و متن خود را در قالب چندین نامه بنویسید و برای هر کدام از صفحات نامه شماره صفحه بگذارید. بعد از اینکه نامه در مقصد به دوستتان رسید، با استفاده از شماره صفحه ها میتواند نامه را به ترتیب خوانده و اطلاعات را به درستی دریافت کند. این دقیقا همان کاری است که لایه شبکه انجام می دهد.

وظیفه لایه شبکه این است که چگونگی رسیدن داده ها به مقصد را تعیین کند. این لایه وظایفی از قبیل آدرس دهی، مسیریابی و پروتکل های منطقی را عهده دار است.

لایه شبکه مسیرهای منطقی یا Logical Path بین مبدا و مقصد را ایجاد میکند که به اصطلاح مدارهای مجازی یا Virtual Circuits نامگذاری می شوند. این مدارها باعث می شوند که هر بسته اطلاعاتی بتواند راهی برای رسیدن به مقصدش پیدا کند. لایه شبکه همچنین وظیفه مدیریت خطا در لایه خود، ترتیب دهی بسته های اطلاعاتی و کنترل ازدهام را نیز بر عهده دارد.

ترتیب بسته های اطلاعاتی بسیار مهم است، زیرا هر پروتکلی برای خود یک حداکثر اندازه بسته اطلاعاتی تعریف کرده است.

برخی اوقات پیش می آید که بسته های اطلاعاتی از این حجم تعریف شده بیشتر می شوند و به ناچار اینگونه بسته های به بسته های کوچکتری تقسیم می شوند و برای هر کدام از این بسته های اطلاعاتی یک نوبت یا Sequence داده می شود که معلوم شود کدام بسته اول است و کدام بسته دوم و.... به این عدد به اصطلاح Sequence Number هم گفته می شود.

وقتی بسته های اطلاعاتی در مقصد دریافت شدند، در لایه شبکه این Sequence Number ها چک می شود و به وسیله همین Sequence Number است که اطلاعات به حالت اولیه باز میگردند و تبدیل به اطلاعات اولیه می شوند. در صورتی که یکی از این بسته های به درستی دریافت نشود، در لایه شبکه از طریق چک کردن این عدد مشخص می شود که کدام بسته اطلاعاتی دریافت نشده است و طبیعتاً مجدداً در خواست داده می شود.

۴-۳-۱-۶. لایه Data Link (ارتباط داده)

□ آدرس دهی فیزیکی و یا سخت افزاری (MAC) ، توپولوژی شبکه

□ Frame ها در این لایه قرار دارند

□ رفع خطاهای فیزیکی

□ قالب بندی داده‌ها

□ هماهنگی بین سرعت گیرنده و فرستنده

□ کنترل دسترسی به کانال مشترک

□ انتقال مطمئن داده از طریق محیط انتقال

□ تقویت کننده داده

در این لایه خطا در اطلاعات کشف و اصلاح می شوند و بدون خطا و به صورت مطمئن به سوی مقصد ارسال می شوند.

وظیفه دیگر این لایه مطمئن شدن از رسیدن اطلاعات به مقصد است که این کار توسط بخش های CheckSum, CRC و Parity Check انجام می پذیرد که در صورت بروز خطا مجددا اطلاعات ارسال خواهند شد.

لایه انتقال به خودی خود به دو زیر لایه به نام های MAC مخفف Media Access Control و LLC مخفف Logical Link Control تقسیم می شود. زیر لایه MAC همانطوری که از نامش پیداست شناسه سخت افزاری کامپیوتر که در واقع همان آدرس MAC کارت شبکه است را به شبکه معرفی میکند. آدرس MAC آدرسی سخت افزاری است که در هنگام ساخت کارت شبکه از طرف شرکت سازنده بر روی کارت شبکه قرار داده می شود و در حقیقت Hard Code می شود. این آدرس در حقیقت مهمترین فاکتور در آدرس دهی است که کامپیوتر از طریق آن بسته های اطلاعاتی را دریافت و ارسال می کند. زیر لایه LLC وظیفه کنترل Frame Synchronization یا یکپارچه سازی فریم ها و همچنین خطایابی در لایه دوم را بر عهده دارد.

۴-۳-۱-۷. لایه Physical (فیزیکی)

□ انتقال بیت‌ها از طریق کانال مخابراتی

□ کابل ها ، کانکتورها ، ولتاژها ، نرخ انتقال داده (bit rate)

□ ارسال اطلاعات به صورت مجموعه ای از بیت ها، سیگنال های الکتریکی و اینترفیس سخت افزاری

این لایه که تنها تشکیل شده از سخت افزار می باشد و قراردادهای سخت افزاری در آن اجرا می شود، وظیفه انتقال نهایی اطلاعات را دارد که این انتقال به صورت سیگنال و به صورت صفر و یک می باشد.

در واقع در این لایه با توجه به محیط انتقال استفاده شده، داده ها را به سیگنال های الکتریکی، پالس هایی از نور و یا سیگنال های رادیویی تبدیل و از طریق کابل برای کامپیوتر مقصد ارسال خواهد شد.

مسائل طراحی در این لایه عمدتاً از نوع فیزیکی، الکتریکی، زمان بندی (Timing) ، نرخ ارسال داده و رسانه فیزیکی انتقال است.

این لایه در واقع تعیین میکند که ما به چه شکل قرار است اطلاعات خود را و از طریق چه رسانه ای انتقال دهیم، برای مثال رسانه ما کابل فلزی است یا تجهیزات بی سیم؟ برای راحت کردن درک این لایه بهتر است بگوییم لایه فیزیکی تعیین میکند که اطلاعات چگونه در سطح رسانه دریافت و ارسال شوند. عملیات کدبندی یا Coding نیز نیز در این لایه انجام می شود.

۴-۳-۱-۸. خلاصه عملکرد هر لایه

به طور خلاصه میتوان کارکرد هر لایه را اینگونه خلاصه کرد :

لایه۱ (لایه فیزیکی) : تعریف الکتریکی و مکانیکی سیستم

لایه۲ (لایه پیوند داده) : قالب بندی و قالب تصحیح خطای داده

لایه۳ (لایه شبکه) : فرستادن بهینه پیام ها از یک شبکه به شبکه دیگر

لایه۴ (لایه انتقال) : کانالی برای ارسال پیام ها از یک فرایند کاربردی به دیگری

لایه۵ (لایه نشست) : سازمان دهی و همزمان سازی تبادل داده

لایه۶ (لایه ارائه) : قالب داده و یا دوباره ارائه دادن

لایه۷ (لایه کاربردی) : ارسال فایل، تبادل پیام ها

مثال هایی از چگونگی کاربرد این لایه ها :

RS-232 و RS-485 مثال هایی از لایه های فیزیکی

پروتکل Modbus مثالی از لایه ی data link است

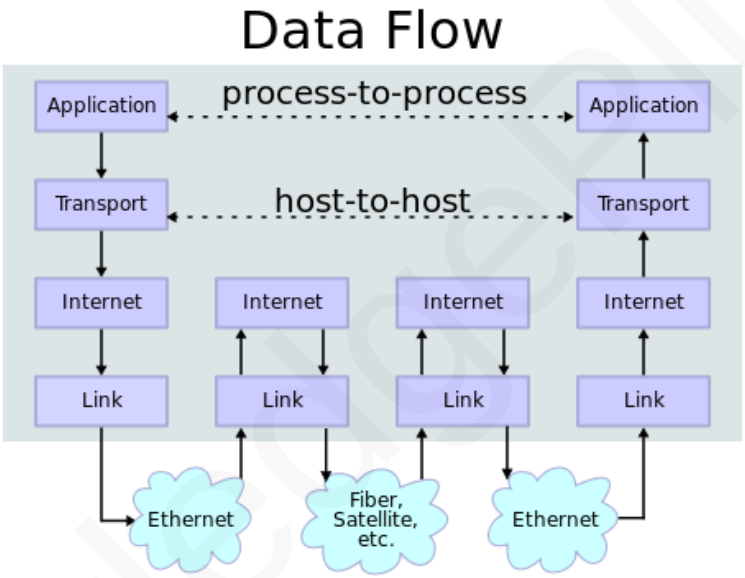
Ethernet شامل لایه های physical و data link است

پروتکل ابزار دقیق هوشمند HART و پروتکل های Profibus و Foundation Profibus در هر

سه لایه ی physical، data link و network کار میکنند.

٤-٣-٢. مدل TCP/IP

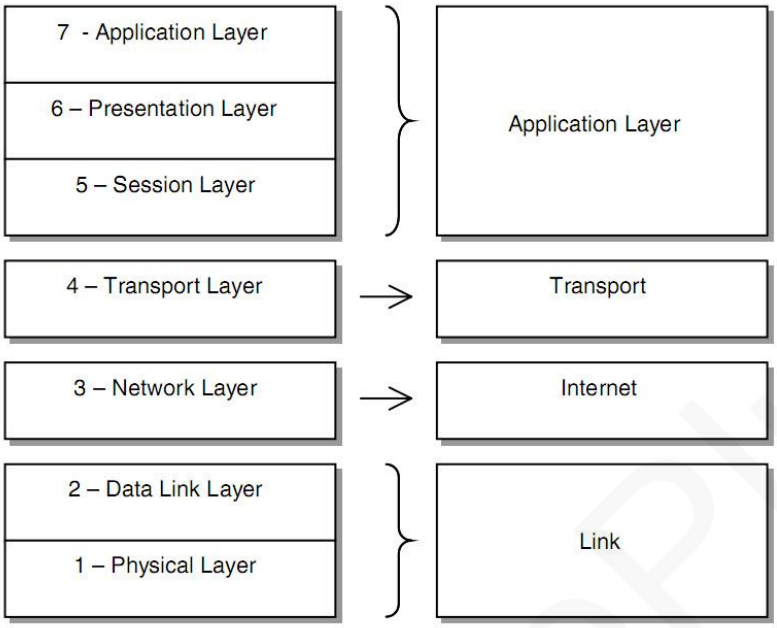
مدل TCP/IP (Transmission control protocol/Internet protocol) که به نام های دیگر مانند Internet protocol suit یا Protocol stack نیز معروف است، مدلی مفهومی و مجموعه ای از پروتکل ها می باشد که در شبکه های اینترنت و شبکه های کامپیوتری مورد استفاده قرار می گیرد.



شکل ۴-۲

علت نام گذاری این مدل به نام TCP/IP به دلیل دو پروتکل اصلی مورد استفاده در این مدل یعنی پروتکل TCP و پروتکل IP می باشد. این مدل شامل ۴ لایه کاربرد، انتقال، اینترنت و پیوند می باشد که عملکرد این لایه ها مشابه لایه های مدل OSI می باشد.

٤-٣-٣. مقایسه مدل OSI با مدل TCP/IP



TCP/IP protocols

شکل ۴-۳

مدل مرجع OSI و مدل مرجع TCP/IP نقاط مشترک زیادی دارند. هر دوی آنها مبتنی بر مجموعه‌ای از پروتکل های مستقل هستند و عملکرد لایه‌ها نیز تا حدی شبیه یکدیگر است. مدل OSI ثابت کرده که بهترین ابزار برای توصیف شبکه‌های کامپیوتری است، اما پروتکل های TCP/IP در مقیاس وسیعی مورد استفاده قرار می‌گیرد. این دو مدل تفاوت هایی با هم دارند که در زیر به برخی از آنها اشاره می کنیم.

قبل از ایجاد مدل OSI پروتکل های آن طراحی و ابداع شد، در نتیجه این مدل وابستگی و تعامل خاصی با هیچ مجموعه پروتکلی ندارد اما در TCP/IP مسئله برعکس بود و این خود باعث شده که مدل TCP/IP تنها برای شبکه‌های تحت خود مناسب باشد.

مدل OSI دارای هفت لایه است اما مدلTCP/IP چهار لایه دارد و از لایه ارائه (presentation) و لایه نشست (session) در آن خبری نیست.

شاید بزرگترین دستاورد مدل OSI روشن ساختن مفاهیم فوق و تفکیک آنها باشد. هر لایه سرویس هایی در اختیار لایه های بالاتر از خود قرار می دهد. تعریف این سرویس ها فقط می گوید که یک لایه چه کاری انجام می دهد و هیچ حرفی در مورد نحوه انجام آنها و چگونگی استفاده از سرویس ها نمی زند.

تعریف چگونگی دسترسی به سرویس های یک لایه بر عهده واسط است. واسط پارامتر های ورودی لازم

و نتیجه ای را که باید منتظر آن باشید تعریف می کند. حتی واسط هم نمی گوید که یک لایه کار خود را چگونه انجام می دهد. کاری را که یک لایه انجام می دهد را پروتکل های آن لایه تعریف می کنند. یک لایه مادامی که کار خود را درست انجام دهد، می تواند از هر پروتکلی استفاده کند. تغییر پروتکل های یک لایه هیچ تاثیری روی ارتباط آن با لایه های بالاتر نخواهد گذاشت.

ایده های فوق بسیار شبیه به مفاهیم مدرن برنامه نویسی شئ گرا هستند. هر شئ، مانند یک لایه، متدهایی (عملکردهایی) دارد که اشیا دیگر از آن استفاده می کنند. نحوه استفاده از این متدها در واقع همان سرویس هایی است که این شئ در اختیار دیگران می گذارد. ورودی ها و خروجی های شئ، واسط آن با دنیای خارج هستند. کد اجرایی شئ نیز شبیه همان پروتکل است که نحوه عملکرد آن از دید دیگران مخفی است.

همان طور که گفته شد، مدل OSI قبل از اختراع پروتکل های آن طراحی و ابداع شد، این بدان معناست که مدل OSI وابستگی و تمایل خاصی به هیچ مجموعه پروتکلی ندارد، این مشخصه یک نقطه ضعف نیز دارد و آن این است که طراحان تجربه چندانی در زمینه موضوع کار ندارند و واقعا نمی دانند کدام عملکرد را باید در کدام لایه قرار دهند. برای مثال، لایه پیوند داده در ابتدا فقط برای شبکه های نقطه به نقطه (point to point) طراحی شده بود و وقتی شبکه های بزرگتر طراحی شدند، مجبور شدند یک زیر لایه به آن اضافه کنند.

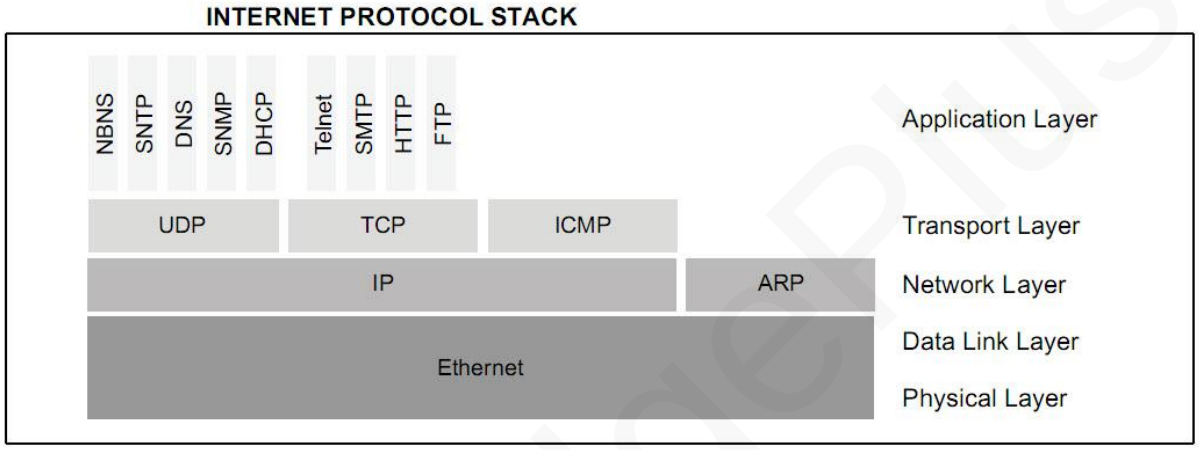
در مورد TCP/IP وضع بر عکس بود و اول پروتکل ها اختراع و توسعه داده شدند و سپس مدلی برای توصیف آنها ساخته شد.

تفاوت دیگر در زمینه اتصال گرا (Connection-Oriented) و غیر اتصال گرا (Connection-Less) می باشد. مدل OSI از هر دو نوع ارتباط اتصال گرا و غیر اتصال گرا در لایه شبکه پشتیبانی می کند ولی در لایه انتقال فقط از سرویس اتصال گرا پشتیبانی می کند. (چون این سرویس در معرض دید کاربران است)

مدل TCP/IP در لایه شبکه فقط سرویس غیر اتصال گرا دارد ولی درلایه انتقال از هر دو نوع ارتباط پشتیبانی می کند و دست کاربر را برای انتخاب باز می گذارد. (که به ویژه برای پروتکل های ساده درخواست و پاسخ (request-response) بسیار مهم است)

٤-٤. طبقه بندی پروتکل ها و جایگاه اترنت

در بخش قبلی در مورد دو مدل OSI و TCP/IP توضیح داده شد. مدلی که برای تشریح عملکرد پروتکل اترنت مناسب می باشد و بیشتر مورد استفاده قرار می گیرد، مدل TCP/IP است. در این بخش به توضیح بیشتر در مورد این مدل و پروتکل های آن و جایگاه پروتکل اترنت در این مدل می پردازیم.



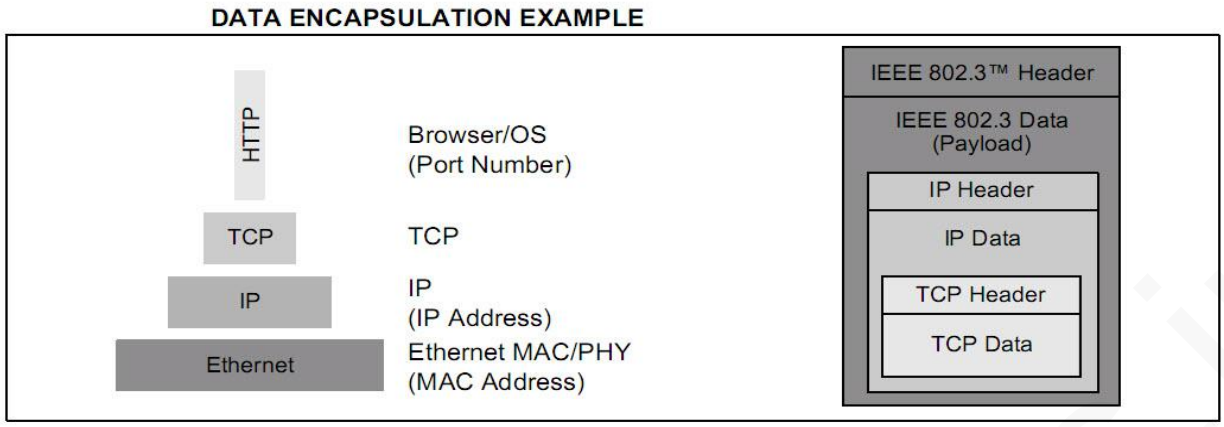
شکل ۴-۴

٤-٤-١. بسته بندی (encapsulation) فریم ها

برای یادگیری نحوه ی عملکرد اترنت باید ابتدا با مفهوم packet encapsulation یا فشرده سازی بسته ها که در بخش قبلی نیز به آن اشاره شد و نحوه به کارگیری آن در پروتکل های شبکه آشنا شویم.

هر لایه در طبقه بندی لایه ها، وظیفه و کارکردی مشخص دارد. به عنوان مثال لایه ی فیزیکی وظیفه انتقال الکتریکی بیت ها در طول رسانا را به عهده دارد. هر لایه ی بالایی از نتایج لایه ی پایین تر خود (بدون وابستگی به آن) استفاده می کند.

برای توضیح مفهوم بسته بندی بسته ها، می توان از مثالی که در ادامه آمده است و تطبیق دهنده ی لایه های مورد استفاده توسط مرورگرهای اینترنتی با مدل TCP/IP است استفاده نمود.



شکل ۴-۵

اگر از لایه ی کاربرد شروع کنیم، مرورگر یک درخواست HTTP طبق فرمان هایی که توسط نوع کاربرد مشخص می شود صادر می کند. این درخواست به لایه ی TCP منتقل می شود و یک بسته ی از نوع پروتکل TCP شامل یک بخش سرآیند (Header) و یک بخش داده (payload) تشکیل می شود. سرآیند شامل اطلاعات مربوط به پروتکل TCP مانند اطلاعات ترتیب بسته ها (packet sequencing information)، اطلاعات بررسی بیت ها (checksum) و آدرس پورت منبع و مقصد (که معمولا برای HTTP برابر ۸۰ است) می شود.

در لایه ی IP ، یک بسته اطلاعاتی IP (IP datagram) به منظور نگه داری بسته دریافت شده از لایه ی TCP تشکیل می شود. بسته اطلاعاتی IP شامل سرآیند IP و داده IP می شود. سرآیند IP شامل مواردی مانند نوع سرویس، اطلاعات بررسی بیت ها، نوع پروتکل (برای TCP برابر 06h) و آدرس IP منبع و مقصد می باشد. در بخش داده IP بسته ی دریافتی از TCP که باید ارسال شود قرار می گیرد.

در لایه ی پیوند داده و لایه ی فیزیکی، بسته اطلاعاتی IP برای شبکه طبق پروتکل IEEE 802.3 ارسال می شود. یک فریم از نوع MAC از بخش سرآیند و داده تشکیل شده است. سرآیند MAC شامل اطلاعاتی در مورد فریم MAC مانند آدرس MAC مقصد و طول فریم می باشد. در بخش داده، بسته اطلاعاتی IP که باید فرستاده شود قرار می گیرد.

باید به این نکته دقت شود که آدرسی که هر پروتکل در فرایند بسته بندی استفاده می کند با آدرس لایه های دیگر متفاوت است، به عبارت دیگر هر لایه آدرس مخصوص به خود را استفاده می کند. مثلا در این مثال لایه ی TCP از شماره پورت استفاده می کند که بستگی به نوع پروتکل لایه ی کاربردی (در این مثال HTTP) دارد. بسته های IP از آدرس IP استفاده می کنند که به صورت استاتیک یا داینامیک به آدرس های اینترنتی تخصیص پیدا می کند و فریم های MAC از آدرس سخت افزاری MAC استفاده می

کنند.

نکته : واژه های "packet" ، "frame" و "datagram" معادل یکدیگر هستند و در پروتکل های مختلف از هریک از آنها استفاده می شود.

در ادامه در مورد پروتکل های رایج مورد استفاده در هریک از لایه های مدل TCP/IP توضیح می دهیم.

٤-٤-٢. پروتکل های لایه کاربرد

لایه ی کاربرد فراهم کننده ی واسط کاربری است. معمولا زمانی که لایه ی کاربرد در بالای لایه های سطح پایین (مانند TCP و UDP) استفاده می شود، به آن یک شماره پورت اختصاص می دهند. به عنوان مثال همان طور که گفته شد معمولا به پروتکل HTTP شماره پورت ۸۰ تخصیص داده می شود.

لیست زیر شامل پروتکل های رایج لایه ی کاربرد که در اینترنت استفاده می شود می باشد.

HTTP	مخفف Hyper Text Transfer Protocol معمولا برای انتقال داده های مرورگر اینترنتی استفاده می شود
SMTP	مخفف Simple Mail Transfer Protocol برای انتقال ایمیل در اینترنت به کار می رود
FTP	مخفف File Transfer Protocol برای انتقال فایل در اینترنت به کار می رود
DNS	مخفف Domain Name System ، تبدیل نام دامنه های اینترنتی به آدرس IP مانند تبدیل knowledgeplus.ir به آدرس IP معادل آن
DHCP	مخفف Dynamic Host Configuration Protocol ، تخصیص داینامیک (خودکار) آدرس IP به گره های شبکه
Telnet	برای ساختن ارتباط TCP متقابل در هر گره
SNTP	مخفف Simple Network Time Protocol وظیفه مدیریت کلاک گره های شبکه برای سنکرون (همزمان) شدن به کلاک مرجع شبکه را دارد
SNMP	مخفف Simple Network Management Protocol وظیفه بررسی و نظارت بر روی گره های شبکه برای مداخله در شرایط های خاص مانند به وجود آمدن خطا در شبکه

جدول ۴-۲

٤-٤-٣. پروتکل های لایه انتقال

لیست زیر شامل پروتکل های رایج لایه ی انتقال که در اینترنت استفاده می شود می باشد.

TCP	مخفف Transmission Control Protocol ، فراهم کننده ی ارتباطی مطمئن با لایه ی کاربرد
UDP	مخفف User Datagram Protocol ، فراهم کننده ی ارتباطی سریع اما غیرمطمئن با لایه ی کاربرد
ICMP	مخفف Internet Control Message Protocol ، ارسال کننده ی پیام های خطا یا وضعیت به گره های شبکه

جدول ۴-۳

٤-٤-٤. پروتکل های لایه شبکه

لایه ی شبکه مشخص کننده ی چگونگی هدایت بسته ها در طول شبکه است که شامل QOS (Quality of Service یا کیفیت سرویس)، سرویس ها، قوانین آدرس شبکه برای لایه ی انتقال و... می باشد.

لیست زیر شامل پروتکل های رایج لایه ی شبکه که در اینترنت استفاده می شود می باشد.

ARP	مخفف Address Resolution Protocol ، ترجمه کننده ی آدرس های پروتکل به آدرس های واسط سخت افزاری، مانند تبدیل آدرس IP به آدرس MAC
RARP	مخفف Reverse Address Resolution Protocol ، ترجمه کننده ی آدرس های واسط سخت افزاری به آدرس های پروتکل ، مانند تبدیل آدرس MAC به آدرس IP
IP	مخفف Internet Protocol ، پروتکل بدون اتصال (Connectionless) لایه ی شبکه که توسط پروتکل های دیگر مانند TCP ، UDP و... استفاده می شود

جدول ۴-۴

٤-٤-٥. پروتکل های لایه پیوند داده و فیزیکی

لایه ی فیزیکی فراهم کننده ی انتقال جریان بیت ها از طریق اتصالات فیزیکی است و دارای وظایفی مانند کد کردن، مالتی پلکس (تسهیم) کردن، همزمان سازی (synchronization)، بازیابی کلاک (clock recovery)، تقسیم بندی (serialization) و... می باشد.

وظیفه ی لایه ی پیوند داده اطمینان از انتقال مطمئن فریم ها بدون وجود خطا است و وظایفی مانند ترتیب بندی فریم ها (frame sequencing)، کنترل جریان فریم ها و... می باشد.

Ethernet یکی از رایج ترین پروتکل های لایه های فیزیکی و پیوند داده است.

٤-٥. مفهوم Client/Server

همه ی پروتکل های لایه کاربرد که در بخش قبلی معرفی شدند (Telnet ، FTP ، HTTP و...) در یک رابطه Client/server مورد استفاده قرار می گیرند.

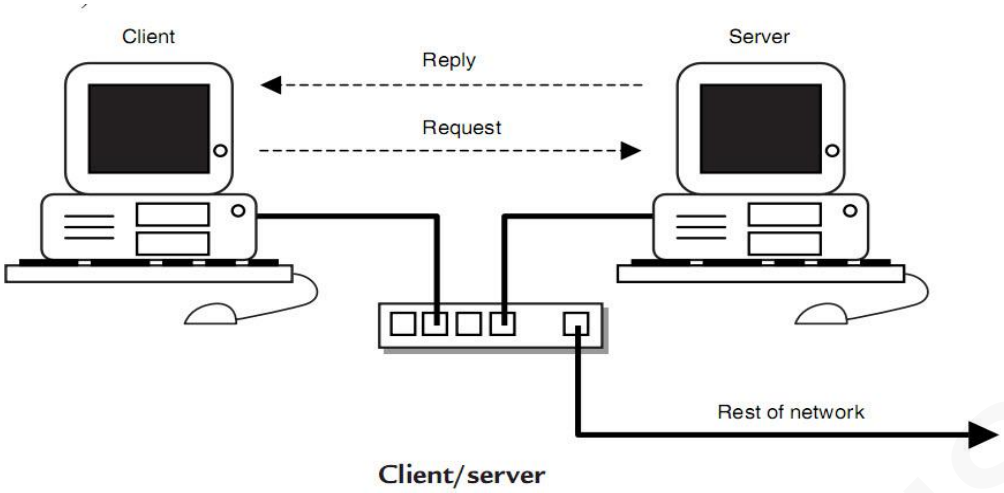
Client یا سرویس گیرنده، برنامه یا کامپیوتری است که سرویسی را از برنامه یا کامپیوتر دیگر درخواست می کند. مرورگر اینترنتی یکی از رایج ترین Client های مورد استفاده است.

Server یا سرویس دهنده، برنامه یا کامپیوتری است که به برنامه یا کامپیوتر دیگر سرویس می دهد. به عنوان مثال یک web server را می توان نام برد.

رابطه ی client/server توضیف کننده ی رابطه ای بین درخواست کننده ی سرویس (client) و ارائه کننده ی سرویس (server) می باشد.

٤-٦. مفهوم port و socket

کامپیوترها در یک رابطه client/server، از طریق socket ها و با زبان protocol با یکدیگر ارتباط برقرار می کنند.



شکل ۴-۶

سوکت (socket) یک اتصال منطقی (logical connection) برای کامپیوترها برای رد و بدل اطلاعات با دیگر کامپیوترها می باشد. به همین شکل یک پورت (port) سریال، یک اتصال فیزیکی (physical connection) برای ارتباط بین کامپیوتر و دیگر تجهیزات است.

سوکت در واقع نقطه ی انتهایی اتصال دو طرفه بین برنامه های در حال اجرا در شبکه است. ارتباط دو طرفه در شبکه توسط زوج client-server اجرا می شود.

سرور یک سوکت ایجاد کرده و منتظر می ماند تا کلاینت درخواستی برای او صادر کند. کلاینت با ایجاد یک سوکت درخواست خود را به سرور ارسال می کند. مکانیزم ارتباط دهنده ی بین سوکت های سرور و کلاینت، پورت نامیده می شود.

پورت یک شماره ۱۶ بیتی است که به پروتکل های مختلف لایه ی کاربرد اختصاص داده می شود.

ارتباط بین Client/server را میتوان اینگونه تشریح کرد :

سرور یک سوکت با شماره پورت سرویسی که می خواهد ارائه کند ایجاد می کند. کلاینت برای درخواست سرویس خود، یک سوکت با شماره پورت مربوط به سرویس مورد نظر ایجاد می کند. سرور درخواست ارسال شده را می پذیرد و با اطلاعات جواب می دهد.

شماره پورت های 1-1023 برای کاربردهای مشخص مانند ایمیل، صفحات وب و... رزرو شده هستند که جدول صفحه بعد شماره پورت های شناخته شده به همراه سرویس آنها را نشان می دهد.

سرویس	Port
Echo	7
Daytime	13
FTP Data	20
FTP Control	21
Telnet	23
SNMP	25
DNS	53
Gopher	70
Finger	79
http	80
NNTP	119

جدول ۴-۵

برای ارتباط با هر کدام از این سرویس ها، باید از شماره پورت مربوطه در کدهای خود استفاده کنیم.

از شماره پورت های 1024-65,535 می توان در کدنویسی برای پیاده سازی کاربردهای و سرویس های سفارشی استفاده کرد.

برای مشاهده لیست کامل سرویس ها و شماره پورت ها می توانید به برگه اطلاعاتی RFC1700 یا آدرس :

http://www.isi.edu/in-notes/iana/assignments/port-numbers

مراجعه فرمایید.

٤-٧. مشخصات و ویژگی های اترنت

نکته: این بخش شامل خلاصه ای کوتاه برای معرفی و آشنایی اولیه با پروتکل ارتباطی Ethernet است و بخش های بعدی در مورد هر کدام از نکات مطرح شده در این بخش مانند استانداردها، کابل های مورد استفاده و ... به طور کامل تر توضیح داده خواهد شد.

Ethernet (اترنت)، پروتکل، استاندارد یا تکنولوژی است از خانواده ی شبکه های کامپیوتری که معمولا در شبکه های LAN (Local area network) و MAN (Metropolitan area network) مورد استفاده قرار می گیرد.

تجهیزاتی که از طریق اترنت ارتباط برقرار می کنند داده ها را به قسمت های کوچکتری به اسم قالب (frame) تبدیل می کنند. هر استاندارد ممکن است ساختار خاصی را برای فریم تعریف کرده باشد. (به عنوان مثال ساختار فریم در استاندارد 10Base-T و 1000Base-T)

یک فریم از چندین بخش (field) تشکیل می گردد. هر فیلد نیز از مجموعه ای بایت تشکیل شده است. در بخش های بعدی در مورد ساختار فریم ها صحبت خواهیم کرد.

هر فریم شامل آدرس فرستنده و گیرنده و اطلاعات خطایابی (error-checking) است که توسط آن گیرنده متوجه قالب های خطا می شود و در ارتباطی که پروتکل های لایه های بالاتر نیز پیاده سازی می شود، میتوان درخواست ارسال مجدد (retransmission) قالب های خطا را از سمت گیرنده ارسال کرد.

یکی از ویژگی های اترنت از زمان عرضه تجاری آن در سال ۱۹۸۰ تا کنون، ویژگی backward compatibility است به معنای سازگاری و تطبیق با تکنولوژی ها و محصولات قبلی است.

اترنت یک پروتکل/واسط آسنکرون (غیر همزمان) می باشد که از روش CSMA/CD برای جلوگیری از ایجاد تداخل در شبکه استفاده می کند. ظرفیت انتقال اطلاعات 46-1500 octets (بایت) می باشد. علارغم پشتیبانی از سرعت های بالا تا چند صد مگابیت بر ثانیه ، برای کاربردهای با توان پایین مناسب نمی باشد.

اترنت یک پروتکل مبتنی بر لایه ی پیوند داده و لایه ی فیزیکی است که مشخصات آن در استاندارد IEEE 802.3 تعیین شده است.

اترنت بسته به سرعت، حالت انتقال و رسانای فیزیکی دارای انواع مختلفی است. سرعت های اترنت که

امروزه استفاده می شوند شامل 10Mbit/s ، 100Mbit/s ، 1000Mbit/s و... می شود.

رسانای انتقال فیزیکی مورد استفاده می تواند کابل Coaxial ، فیبر نوری، کابل UTP و... باشد.

برخی از ویژگی های اترنت عبارت است از :

- رایج ترین شبکه در دفاتر و ساختمان های تجاری
- قابلیت توسعه آسان
- قابلیت نظارت و کنترل تجهیزات متصل شده به شبکه از طریق اینترنت
- تاخیر پایین، عملکرد real-time

همه ی فریم ها به همراه بیت ها، بایت ها و فیلدهای مربوطه مستعد خطاء از منابع متعددی می باشند. بخش FCS (Frame check sequence) شامل یک مقدار عددی است که توسط گره مبداء و بر اساس داده موجود در فریم محاسبه می گردد. پس از محاسبه FCS ، مقدار استخراج شده به انتهای فریم ارسالی اضافه خواهد شد. زمانی که گره مقصد فریم را دریافت می نماید، مجددا مقدار FCS را محاسبه و با مقدار موجود در فریم مقایسه می گردد. در صورتی که دو عدد با یکدیگر متفاوت باشند، نشان دهنده بروز خطاء در زمان ارسال اطلاعات می باشد. در چنین مواردی فریم دورانداخته شده و از گره مبداء درخواست می شود که مجددا اطلاعات را ارسال نماید. (فرایند retransmission)

برای محاسبه FCS از سه روش عمده استفاده می گردد :

روش اول : Cyclic Redundancy Check (CRC)، محاسبات را بر روی داده انجام می دهد.

روش دوم : Two-dimensional parity ، در این روش با اضافه کردن بیت هشتم، زوج و یا فرد بودن تعداد یک های موجود در فریم مشخص می گردد.

روش سوم : Internet checksum ، در این روش مقدار تمامی بیت های داده با یکدیگر جمع می گردد.

هر گره زمانی که فریم را دریافت می کند، آدرس MAC آن را بررسی می کند تا مطمئن گردد که پیام برای وی ارسال شده است یا خیر؟ در صورتی که پیام برای گره مورد نظر ارسال نشده باشد، گره فریم را بدون بررسی محتویات آن کنار خواهد گذاشت. یکی از نکات قابل توجه در رابطه با آدرس دهی فریم های اترنت، پیاده سازی یک آدرس Broadcast است. زمانیکه آدرس مقصد یک فریم از نوع Broadcast باشد، تمام گره های موجود در شبکه آن را دریافت و پردازش خواهند کرد.

٤-٨. استانداردهای اترنت

در نسخه های جدیدتر اترنت، همه ی گره ها از یک کابل یا رسانای مشترک استفاده نمی کنند و هر گره در یک توپولوژی ستاره، فقط با تجهیزات شبکه مانند switch ارتباط برقرار می کنند، در این حالت تصادم (collision) تنها در صورتی اتفاق می افتد که به عنوان مثال گره شبکه و switch به طور همزمان بخواهند داده ای را به یکدیگر ارسال کنند.

در شبکه های اترنت اولیه وضعیت ارسال و دریافت اطلاعات بصورت یکطرفه (half-duplex) بود.در شبکه های مبتنی بر سوئیچ، گره ها صرفا با سوئیچ ارتباط برقرار کرده و قادر به ارتباط مستقیم با یکدیگر نمی باشند.

در نسخه های جدیدتر مانند 10Base-T که از حالت Full-Duplex پشتیبانی می کنند، سوییچ و گره های شبکه به طور همزمان به یکدیگر داده ارسال می کنند و عملا در این شبکه ها دیگر مسئله تصادم وجود نخواهد داشت.

٤-٨-١. تاریخچه

اولین شبکه مبتنی بر کابل های UTP در اواسط دهه ۱۹۸۰ به نام StarLan با سرعت 1Mbit/s طراحی و تحت عنوان استاندارد 1Base-5 معرفی شد.

شبکه SynOptics (که بعدها به نام LattisNet معروف شد) اولین شبکه اترنت با سرعت 10Mbit/s و توپولوژی ستاره تشکیل شده از repeater ها بود که منجر به ارائه استاندارد 10BASE-T شد.

در ماه February سال ۱۹۸۰ سازمان IEEE پروژه ای را برای استانداردسازی شبکه های LAN در لایه های فیزیکی و پیوند داده مدل OSI شروع کرد. موسسه IEEE نام گروه فوق را 802 قرار داد. (عدد 802 نشان دهنده ی سال و ماه تشکیل کمیته استانداردسازی است)

گروه DIX ، Gary Robinson (از شرکت DEC)، Phil Arst (از شرکت Intel) و Bob Printis (از شرکت Xerox) مشخصات پیشنهادی خود را برای شبکه های LAN در کتاب معروف به کتاب آبی (Blue Book) CSMA/CD ارائه دادند.

علاوه بر CSMA/CD ، Token Ring (با حمایت IBM) و Token Bus (با حمایت General Motors) هم جزو کاندیده ها برای استاندارد شبکه LAN بودند.

پروپزال های ارائه شده بسیار به هم نزدیک بودند و رقابت شدید برای انتخاب استاندارد ایجاد شد. در ماه December سال ۱۹۸۰ گروه تشکیل شده برای استاندارد سازی به سه گروه مجزا تقسیم شدند و کار نوشتن استاندارد برای هر بخش را به صورت جداگانه ادامه دادند.

موسسه IEEE برای تمایز هر یک از کمیته های جانبی از روش نام گذاری 802.x استفاده کرد که x یک عدد منحصر به فرد بوده که برای هریک از کمیته ها در نظر گرفته شده بود.

گروه 802.3 مسئول استانداردسازی عملیات درشبکه های CSMA/CD (مانند اترنت) را برعهده داشتند.

سازمان IEEE پیش نوشت اولیه ای از استاندارد 802.3 را در سال ۱۹۸۳ منتشر کرد و در سال ۱۹۸۵ این استاندارد را به صورت رسمی ارائه داد.

استاندارد ISO 802.3 نیز در سال ۱۹۸۹ از سوی سازمان ISO معرفی شد.

٤-۸-۲. قوائد نام گذاری

زمانی که لازم است اترنت به منظور اضافه کردن یک رسانه انتقال داده جدید و یا قابلیتی خاص توسعه یابد، موسسه IEEE یک ضمیمه جدید را برای استاندارد IEEE 802.3 ارائه می نماید . ضمیمه فوق دارای یک و یا دو حرف تکمیلی است. به عنوان مثال استاندارد IEEE 802.3a برای شبکه های با نام Thinnet و با سرعت 10Mbit/s می باشد یا استاندارد IEEE 802.3ae برای شبکه های اترنت با سرعت 10Gbit/s می باشد.

در مورد نام روش ها و نسخه های مختلف پیاده سازی، از یک نام کوتاه شده بر اساس مجموعه قوانین زیر به ضمیمه نسبت داده می شود :

- عددی که نشان دهنده نرخ داده یا سرعت ارتباط بر حسب Mbit/s می باشد. (مثلا 10 برای 10Mbit/s)
- حرف B که نشان دهنده استفاده از روش سیگنالینگ Baseband می باشد.
- یک و یا چندین حرف الفبائی که نوع رسانه انتقال داده را مشخص می نماید (مثلا حرف F برای فیبر

نوری و یا T برای کابل های مسی به هم تابیده شده (Twisted-Pairs))

به عنوان مثال منظور از استاندارد 10Base-T این است که سرعت این استاندارد 10Mbit/s ، روش سیگنالینگ آن Baseband و کابل مورد استفاده از نوع کابل زوج به هم تابیده شده (کابل های CAT) می باشد.

در روش سیگنالینگ Baseband ، از تمامی پهنای باند کابل شبکه استفاده نموده و سیگنال داده مستقیما بر روی رسانه انتقال داده ارسال می گردد. در سیگنالینگ Broadband که توسط اترنت استفاده نمی گردد، سیگنال داده هرگز مستقیما بر روی محیط انتقال داده قرار نمی گیرد و یک سیگنال آنالوگ (Carrier Signal) با سیگنال داده مدوله شده و سیگنال فوق ارسال می گردد. همچنین در سیستم Baseband از تمامی پهنای باند به منظور انتقال اطلاعات استفاده می گردد ولی در سیستم Broadband به منظور استفاده همزمان، پهنای باند به کانال های متعددی تقسیم می گردد. شبکه های رادیوئی و شبکه های کابلی تلویزیون از روش Broadband استفاده می نمایند .

از سال ۱۹۸۳ تاکنون، استانداردهای متفاوتی ارائه شده است که یکی از اهداف مهم آنها، تامین پهنای باند مناسب به منظور انتقال اطلاعات است. ما امروزه شاهد رسیدن به مرز گیگابیت در شبکه های کامپیوتری می باشیم.

پهنای باند ارائه شده توسط اترنت در نسخه های اولیه 10Mbit/s بود و برای کامپیوترهای شخصی دهه ۱۹۸۰ که دارای سرعت پائین بودند، کافی به نظر می آمد ولی در اوایل سال ۱۹۹۰ که سرعت کامپیوترهای شخصی و اندازه فایل ها افزایش یافت، مشکل پائین بودن سرعت انتقال داده بهتر نمایان شد. اکثر مشکلات فوق مربوط به کم بودن پهنای باند موجود مربوط می گردید. در سال ۱۹۹۵ موسسه IEEE استانداردی را برای اترنت با سرعت 100Mbit/s معرفی نمود. این روال ادامه یافت و در سال های ۱۹۹۸ و ۱۹۹۹ استانداردهائی برای سرعت های گیگابیت نیز ارائه گردید .

در بخش بعدی به ارائه جزئیات بیشتری از استاندارد های اترنت، از گذشته تا کنون می پردازیم.

٤-٨-٣. اترنت 10Mbit/s (Standard Ethernet)

جدول صفحه بعد خلاصه ای از استانداردهای اولیه ارائه شده برای اترنت را نشان می دهد. در استانداردهای اولیه سرعت 10Mbit/s می باشد و از کد Manchester برای ترکیب سیگنال کلاک و سیگنال اصلی استفاده می شود.(خاصیت self-clocking)

توپولوژی مورد استفاده در استانداردهای قدیمی تر مانند 10Base-5 و 10Base-2 خطی (bus) است. همان طور که در فصل دوم نیز اشاره شد، در توپولوژی خطی نگهداری و عیب یابی شبکه سخت است. استانداردهای جدیدتر اترنت از توپولوژی ستاره ای که توسط تجهیزاتی مانند hub ، switch یا router ساخته می شوند پشتیبانی می کنند.

نکته : امروزه همچنان اتصال دو گره اترنت توسط توپولوژی خطی انجام می شود. (توسط کابل Cross-Over که در ادامه توضیح داده خواهد شد)

نام پیاده سازی	استاندارد	کانکتورهای رایج	توضیحات
Xerox experimental Ethernet	Proprietary	-	سرعت 2.94Mbit/s ، کابل کواکسیال ۵۰ اهم
10BASE5	802.3 (8)	AUI	سرعت 10Mbit/s ، معروف به Thick-Ethernet ، مورد استفاده در دهه ۱۹۸۰، کابل کواکسیال RG-8X ۵۰ اهم، توپولوژی خطی، حداکثر طول کابل ۵۰۰ متر، حداکثر ۲۰۸ گره در هر سگمنت
10BASE2	802.3 (10)	BNC	سرعت 10Mbit/s ،کابل کواکسیال مدل RG-58 ۵۰ اهم، کانکتور T شکل ، تکنولوژی غالب در اواسط دهه ۱۹۸۰ ، معروف به Thin Ethernet ، ThinNet یا Cheapernet ، توپولوژی خطی، حداکثر طول کابل ۱۸۵ متر، حداکثر ۳۰ گره در هر سگمنت
10BROAD36	802.3 (11)	F	سرعت 10Mbit/s ، اولین اترنت برای مسافت های بلند، کابل کواکسیال، توپولوژی خطی ، استفاده از کد NRZ و مدولاسیون PSK
1BASE5	802.3 (12)	8P8C	سرعت 1Mbit/s ، کابل زوج به هم تابیده شده، معروف به StarLAN ، توپولوژی ستاره ای
StarLAN10	Proprietary	8P8C	سرعت 10Mbit/s ، کابل زوج به هم تابیده شده، توپولوژی ستاره
UTPLattisNet	Proprietary	8P8C	سرعت 10Mbit/s ، کابل زوج به هم تابیده شده، توپولوژی ستاره
10BASE-T	802.3 (14)	8P8C	کابل زوج به هم تابیده شده دو تایی (۴ سیمه) کابل CAT3 یا CAT5 ، توپولوژی ستاره
FOIRL	802.3 (9.9)	ST	استاندارد اصلی اترنت توسط فیبر نوری
10BASE-F	802.3 (15)	–	سرعت 10Mbit/s ، فیبر نوری
10BASE-FL	802.3 (15&18)	ST	نسخه جدیدتر استاندارد FOIRL

10BASE-FB	802.3 (15&17)	–	برای اتصال hub ها و switch ها، منسوخ شده
10BASE-FP	802.3 (15&16)	–	توپولوژی ستاره بدون استفاده از repeater ، هیچ وقت استفاده نشده است

جدول ۴-۶

در ادامه در مورد برخی از استانداردهای پرکاربرد این جدول توضیح می دهیم.

۴-۸-۳-۱. 10Base-5

این استاندارد در نسخه IEEE 802.3 با عنوان 10Base-5 یا Thicknet ارائه شده است. عدد 5 حداکثر طول کابل در ۵۰۰ متر (۵ قطعه (segment) ۱۰۰متری) می باشد.

ویژگی	توضیحات
حداکثر طول یک قطعه کابل	۵۰۰ متر
ترنسیورها	به کابل متصل می‌شوند
حداکثر فاصله بین کامپیوتر تا ترنسیور	۵۰ متر (۱۶۴ فوت)
حداقل فاصله میان ترنسیورها	۲/۵ متر (۸ فوت)
تعداد ستونهای اصلی وتکرارکننده‌ها	۵ ستون اصلی با ۴ تکرارکننده می‌توانند متصل شوند
تعداد شاخه‌های متصل به کامپیوتر	۳ شاخه از ۵ شاخه
حداکثر طول شاخه‌های متصل به هم	۲۵۰۰ متر (۸۲۰۰ فوت)
حداکثر تعداد کامپیوترهای هر شاخه	۱۰۰ کامپیوتر مطابق مشخصات

جدول ۴-۷

۴-۸-۳-۲. 10Base-2

این استاندارد در نسخه IEEE 802.3a با عنوان 10Base-2 یا Thinnet ارائه شده است. عدد ۲ نشان دهنده ی حداکثر طول کابل می باشد که ۱۸۵ متر یا حدود ۲۰۰ متر است. حداقل فاصله بین هر

کامپیوتر برابر ۰/۵ متر (۲ اینچ) می باشد.

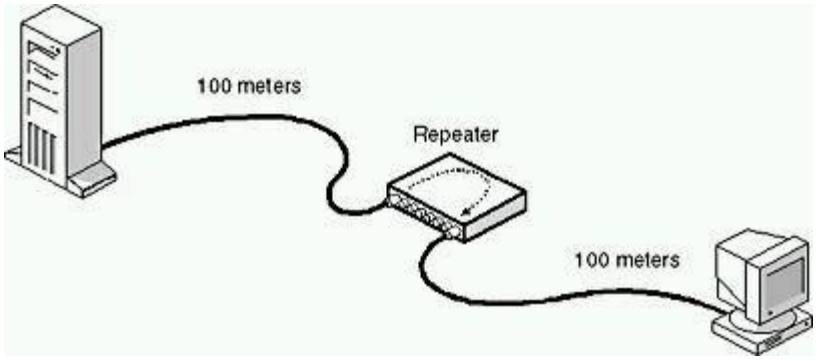
شبکه Thinnet حداکثر می‌تواند از ۳۰ گره (کامپیوتر و تکرارکننده) درهر سگمنت پشتیبانی کند.

ویژگی	توضیحات
حداکثر طول هر شاخه کابل	۱۸۵ متر (۶۰۷ فوت)
نحوه اتصال به کارت شبکه	T کانکتور BNC
تعداد شاخه‌های کابل و تکرارکننده‌ها	۵ شاخه کابل را میتوان با استفاده از ۴ تکرارکننده به هم متصل کرد
تعداد کامپیوترهای هر شاخه	مطابق مشخصات ۳۰ کامپیوتر برای هر شاخه
تعداد شاخه‌هایی که میتوانند کامپیوتر داشته باشند	۳ شاخه از ۵ شاخه
طول حداکثر شبکه	۹۲۵ متر (۳۰۳۵ فوت)

جدول ۴-۸

۴-۸-۳-۳. 10Base-T

یکی از کاربردی ترین استانداردهای اترنت، استاندارد 10Base-T می باشد. این استاندارد در نسخه IEEE 802.3i ارائه شده است. حداکثر طول هر سگمنت در این پیاده سازی ۱۰۰ متر می باشد و از repeater می توان برای افزایش مسافت استفاده کرد. طول کابل میان ۲ کامپیوتر حداقل ۲/۵ متر (حدود ۸ فوت) است. تعداد کامپیوترهای قابل اتصال در این شبکه ۱۰۲۴ عدد می باشد. حداقل کابل مورد استفاده باید CAT3 باشد.



ویژگی	توضیحات
کابل	UTP ۵ یا ۴، ۳ گروه
کانکتورها	RJ-45 در دو انتهای کابل
ترنسیور	هر کامپیوتر به یک ترنسیور نیاز دارد و روی برخی از کارت ها ترنسیور وجود دارد
فاصله بین ترنسیور و هاب	حداکثر ۱۰۰ متر
ستون اصلی هاب‌ها	کابل کواکسیال یا فیبر نوری برای پیوند شبکه‌های بزرگتر
حداکثرتعدادکامپیوترهای هرLANبدون اتصالات	۱۰۲۴ طبق مشخصات

جدول ۴-۹

۴-۸-۳-۴. 10Base-FL

این استاندارد در نسخه IEEE 802.3j ارائه شده است. 10BaseFL از کابل فیبر نوری برای اتصال گره های مختلف شبکه استفاده می کند.

یکی از مهمترین دلایل برای استفاده از فیبر نوری در شبکه اترنت، امکان استفاده از کابل های طولانی تر و ایزوله سازی کابل در مقابل نویزهای محیطی می باشد. حداکثر طول هر شاخه کابل در این استاندارد برابر ۲۰۰۰ متر می باشد.

٤-٨-٤. اترنت 100Mbit/s (Fast Ethernet)

در جدول زیر استانداردهای مربوط به اترنت سریع (Fast Ethernet) با سرعت 100Mbit/s نشان داده شده است. این استانداردها توسط گروه IEEE 802.3u تهیه شده است. در همه ی استاندارد های اترنت سریع از توپولوژی ستاره استفاده می شود.

از کاربردهای این استاندارد با توجه به سرعت بالای آن می توان در ارسال و دریافت فایل های تصویر و فیلم اشاره نمود.

نام پیاده سازی	استاندارد	کانکتورهای رایج	توضیحات
100BASE-T	802.3 (21)		کابل CAT
100BASE-TX	802.3 (24&25)	8P8C	استفاده از کد MLT-34B5B ، کابل CAT5 (۴ سیمه)، طول کابل ۱۰۰ متر
100BASE-T4	802.3 (23)	8P8C	استفاده از کد PAM-38B6T ، کابل CAT3 (۴ سیمه)، ارتباط half-duplex
100BASE-T2	802.3 (32)		استفاده از کد PAM-5 ، کابل CAT3 (۴ سیمه)، ارتباط full-duplex
100BASE-FX	802.3 (24&26)	ST/SC	استفاده از کد 4B5B NRZI ، حداکثر طول کابل برای حالت half-duplex برابر ۴۰۰ متر و برای حالت full-duplex برابر ۲ کیلومتر، فیبر نوری
100BASE-SX	-785TIA	ST/SC	حداکثر طول کابل ۳۰۰ متر، فیبر نوری
100BASE-BX10	802.3 (58)	ST/SC/LC	فیبر نوری، استفاده از مالتی پلکسر برای جداسازی سیگنال های ارسال و دریافت، حداکثر طول کابل ۱۰ کیلومتر
100BASE-LX10	802.3 (58)	ST/SC/LC	فیبر نوری، حداکثر طول کابل ۱۰ کیلومتر
100Base-VG	802.12	8P8C	کابل CAT3 (۴ سیمه)، منسوخ شده

جدول ۴-۱۰

در ادامه در مورد برخی از استانداردهای پرکاربرد این جدول توضیحاتی را ارائه می دهیم.

سه استاندارد پر کاربرد دیگر این جدول 100Base-T4 ، 100Base-TX و 100Base-FX می باشند.

استاندارد 100Base-T4 از چهار زوج (یکی برای ارسال و دیگری برای برگشت و دو زوج دیگر bi-Directional یا دو طرفه) برای انتقال سیگنال استفاده می کند، با استفاده از این روش می توان از کابل های ارزان تر قیمت (voice grade) مانند CAT3 استفاده کرد. پیاده سازی 100Base-T4 (با توجه به نوع کابل) ارزان تر است.

یکی از ویژگی های 100Base-TX قابلیت Full Duplex یا ارتباط دو طرفه می باشد که در

استاندارد 100Base-T4 وجود ندارد.

برخی از مزایای استاندارد 100Base-FX عبارت اند از پشتیبانی از حالت Full Duplex ، مسافت های طولانی تر (تا ۱۸۵ متر و در حالت Full Duplex تا ۴۰۰ متر و حتی ۲ کیلومتر)، مشخصات EMC بهتر و امنیت بیشتر. کابل مورد استفاده در این استاندارد فیبر نوری (با مشخصات multimode fiber with a 62.5 micron core and 125 micron cladding) می باشد و مشابه 10Base-FL فقط یک زوج از زوج های فیبر نوری استفاده می شود.(یک سیم برای ارسال و دیگری برای دریافت)

کانکتور پیشنهادی توسط EIA کانکتور SC هست.

کاربرد 10Base-T در پهنای باند کم و کاربردهای ساده است. کاربرد 100Base-T در کاربردهای با پهنای باند بالاتر مانند Voice over Ethernet یا Remote secure monitoring می باشد.

در جدول زیر برخی از استانداردهای رایج اترنت و مشخصات آنها مانند نرخ داده، روش سیگنالینگ، حداکثر طول کابل هر سگمنت، رسانا و توپولوژی مورد استفاده آمده است.

100BaseT	10BaseFL	10BaseT	10Base2	10Base5	Ethernet Value	Characteristics
10	10	10	10	10	10	Data rate (mbps)
Baseband	Baseband	Baseband	Baseband	Baseband	Baseband	Signaling method
500	500	500	500	500	500	Maximum segment length
UTP cable	Fiber-optic	UTP cable	50-ohm coaxial	50-ohm coaxial	50-ohm coaxial	Media
Bus	Point-to-Point	Star	Bus	Bus	Bus	topology

جدول ۴-۱۱

٤-۸-۵. اترنت 1000Mbit/s (Gigabit Ethernet)

جدول زیر استاندارد های مربوط به اترنت گیگابیت (Gigabit Ethernet) را که از سرعت 1000Mbit/s یا 1Gbit/s پشتیبانی می کند را نشان می دهد. این استاندارد توسط گروه IEEE 802.3z تهیه شده است. در همه ی استانداردها از توپولوژی ستاره استفاده می شود.

نام پیاده سازی	استاندارد	کانکتورهای رایج	توضیحات
1000BASE-T	802.3 (40)	8P8C	استفاده از کد PAM-5 ، کابل

			CAT5 یا CAT5e (۴ سیمه)
1000BASE-TX	TIA-854		کابل CAT6
1000BASE-SX	802.3 (38)	ST/SC/LC	استفاده از کد NRZ8B10B ، فیبر نوری، حداکثر طول کابل ۵۵۰ متر
1000BASE-LX	802.3 (38)	SC/LC	استفاده از کد NRZ8B10B ، فیبر نوری، حداکثر طول کابل ۵۵۰ متر با فیبر نوری multi-mode و ۲ تا ۱۰ کیلومتر با فیبر نوری single-mode
1000BASE-LH	multi-vendor	SC/LC	حداکثر طول کابل ۱۰۰ کیلومتر
1000BASE-CX	802.3 (39)	CX4	استفاده از کد 8B10B NRZ ، کابل STP و حداکثر طول کابل ۲۵ متر
1000BASE-BX10	802.3 (59)	SC/LC	فیبر نوری single-mode و حداکثر ۱۰ کیلومتر
1000BASE-LX10	802.3 (59)	SC/LC	فیبر نوری single-mode و حداکثر ۱۰ کیلومتر
1000BASE-PX10-D	802.3 (60)	SC/LC	فیبر نوری single-mode و حداکثر ۱۰ کیلومتر
1000BASE-PX10-U	802.3 (60)		فیبر نوری single-mode و حداکثر ۱۰ کیلومتر
1000BASE-PX20-D	802.3 (60)		فیبر نوری single-mode و حداکثر ۱۰ کیلومتر
1000BASE-PX20-U	802.3 (60)		فیبر نوری single-mode و حداکثر ۱۰ کیلومتر
1000BASE-ZX	Unknown	SC/LC	فیبر نوری single-mode و حداکثر ۱۰۰ کیلومتر
1000BASE-KX	802.3 (70)		حداکثر ۱ متر برای backplane

جدول ۴-۱۲

در جدول زیر دو استاندارد مربوط به سرعت های 2.5Gbit/s و 5Gbit/s نشان داده شده است. در این دو سرعت فقط از کابل های CAT استفاده می شود.

نام پیاده سازی	استاندارد	کانکتورهای رایج	توضیحات
2.5GBASE-T	802.3bz	8P8C	کابل CAT5e حداکثر ۱۰۰ متر
5GBASE-T	802.3bz	8P8C	کابل CAT6 حداکثر ۱۰۰ متر

جدول ۴-۱۳

جدول زیر استانداردهای مربوط به سرعت 10Gbit/s را نشان می دهد. سهم اصلی بازار در سرعت های 10Gbit/s مربوط به دو استاندارد 10GBASE-LR و 10GBASE-ER می باشد. این استاندارد توسط گروه IEEE 802.3ae تهیه شده است.

نام پیاده سازی	استاندارد	کانکتورهای رایج	توضیحات
10GBASE-SR	802.3 (52)	SC/LC	فیبر نوری multi-mode حداکثر طول ۲۶ متر تا ۸۲ متر بسته به کیفیت کابل، فیبر نوری multi-mode حداکثر طول کابل ۳۰۰ یا ۴۰۰ متر
10GBASE-LX4	802.3 (53)	SC/LC	حداکثر طول ۲۴۰ متر تا ۳۰۰ متر در فیبر نوری multi-mode و طول ۱۰ کیلومتر در فیبر نوری single-mode
10GBASE-LR	802.3 (52)	SC/LC	حداکثر طول ۱۰ کیلومتر در فیبر نوری single-mode
10GBASE-ER	802.3 (52)	SC/LC	حداکثر طول ۴۰ کیلومتر در فیبر نوری single-mode
10GBASE-SW	802.3 (52)		پشتیبانی از لایه فیزیکی WAN
10GBASE-LW	802.3 (52)		پشتیبانی از لایه فیزیکی WAN
10GBASE-EW	802.3 (52)		پشتیبانی از لایه فیزیکی WAN
10GBASE-CX4	802.3 (54)	CX4	استفاده از کانکتور infiniBand 4X و کابل CX4 ، حداکثر طول ۱۵ متر
10GBASE-T	802.3 (55)	8P8C	کابل UTP
10GBASE-LRM	802.3 (68)	SC/LC	استفاده از کابل multimode حداکثر طول کابل ۲۲۰ متر
10GBASE-KX4	802.3 (71)		طول ۱ متر برای ۴ backplane
10GBASE-KR	802.3 (72)		طول ۱ متر برای backplane

جدول ۴-۱۴

در جدول زیر استانداردهای مربوط به دو سرعت 40Gbit/s و 100Gbit/s نشان داده شده است.

40 gigabits/second	100 gigabits/second	استاندارد	کانکتورهای رایج	توضیحات
40GBASE-KR4		802.3 (84)		حداقل ۱ متر برای backplane
40GBASE-CR4	100GBASE-CR10	802.3 (85)	QSFP+/CX4	حداکثر طول کابل حدود ۷ متر
40GBASE-SR4	100GBASE-SR10	802.3 (86)	MPO	۱۰۰ متر برای فیبر نوری multi-mode و ۱۵۰ متر برای فیبر نوری multi-mode
40GBASE-LR4		802.3 (87)	SC/LC	حداقل طول کابل ۱۰ کیلومتر برای فیبر نوری single-mode
	100GBASE-LR4	802.3 (88)	SC/LC	
	100GBASE-ER4	802.3 (88)	SC/LC	حداقل ۴۰ کیلومتر برای فیبر نوری single-mode
40GBASE-FR		802.3 (89)	SC/LC	بیش از ۲ کیلومتر برای فیبر نوری single-mode

جدول ۴-۱۵

٤-٨-٦. اترنت First mile

در جدول ۴-۱۶ استانداردهای اترنت تحت عنوان First mile که به منظور ارائه از سوی ISP ها برای منازل و شرکت های کوچک ارئه شده است، نشان داده شده است.

نام پیاده سازی	استاندارد	توضیحات
10BaseS	Proprietary[22]	اترنت در بستر VDSL
2BASE-TL	802.3 (63)	در بستر سیم های تلفن
10PASS-TS	802.3 (62)	
100BASE-LX10	802.3 (58)	در بستر فیبر نوری Single-mode
100BASE-BX10		
1000BASE-LX10	802.3 (59)	
1000BASE-BX10		
1000BASE-PX10	802.3 (60)	در بستر Passive optical network یا شبکه نوری غیرفعال
1000BASE-PX20		

جدول ۴-۱۶

در حال حاضر استانداردهای مربوط به سرعت های 25Gbit/s و 50Gbit/s در گروه IEEE 802.3cd در حال بررسی می باشد. همچنین نسل جدید اترنت تحت عنوان Terabit Ethernet با سرعت ها بیش از 1000Gbit/s یا 1Tbit/s در حال توسعه می باشند.

نکته : برخی از استانداردهای شبکه جزو استانداردهای IEEE نمی باشند اما از قالب های شبکه اترنت پشتیبانی می کنند و قابلیت کار در شبکه های اترنت را دارا می باشند مانند LattisNet و 100BaseVG ، همچنین استانداردهای 802.11 و 802.16 که مربوط به شبکه بی سیم (Wireless) هستند، با این که از قالب های شبکه اترنت پشتیبانی نمی کنند، با این حال با استفاده از تجهیزاتی مانند Bridge های مبتنی بر آدرس MAC می توانند به شبکه اترنت متصل شوند.

مشخصات اترنت در طول سال های مختلف به منظور پشتیبانی از سرعت ها بالاتر و کاربرد های جدیدتر توسعه یافته است. این توسعه در قالب ارائه مکمل یا الحاقیه (Supplement) های جدید ارائه می شود. جدول زیر روند این توسعه را که به تعدادی از آنها در بخش های قبلی اشاره شد را در طول سال های ۱۹۸۵ تا ۲۰۰۳ نشان می دهد.

Supplement	Year	Description
IEEE 802.3a	1985	10Base-2 Thin Ethernet
IEEE 802.3c	1985	10 Mb/s Repeater Specification
IEEE 802.3d	1987	Fiber Optic Inter-Repeater Link
IEEE 802.3i	1990	10Base-T Twisted Pair
IEEE 802.3j	1993	10Base-F Fiber Optic
IEEE 802.3u	1995	100Base-T Fast Ethernet and Auto-Negotiation
IEEE 802.3x	1997	Full-Duplex Standard
IEEE 802.3z	1998	1000Base-X Gigabit Ethernet (SX, LX, CX)
IEEE 802.3ab	1999	1000Base-T Gigabit Ethernet over Twisted Pair
IEEE 802.3ac	1998	Frame Size Extension to 1522 Octets for VLAN Tagging
IEEE 802.3ad	2000	Link Aggregation for Parallel Links
IEEE 802.3af	2003	Power Over Ethernet (PoE)

جدول ۴-۱۷

٤-٩. کابل های اترنت

در بخش قبلی در جداول ارائه شده برای استانداردهای مختلف اترنت، توضیحاتی در مورد کابل های مورد استفاده در هر استاندارد داده شد.

در این بخش و بخش بعدی در مورد کابل ها و کانکتورهای رایج مورد استفاده در استانداردهای اترنت امروزی توضیح می دهیم. علاوه بر این استانداردها، شرکت های مختلف برای محصولات خود ممکن است از کانکتورها و کابل های دیگر نیز استفاده کنند که علت این مسئله اغلب افزایش مسافت توسط فیبر نوری است.

به طور کلی امروزه برای شبکه های اترنت از دو نوع کابل استفاده می شود، کابل های زوج به هم تابیده شده تحت عنوان CAT (مخفف category به معنای دسته بندی) و فیبر نوری.

به طور معمولا در سرعت های پایین (مانند 10Mbit/s و 100Mbit/s) از کابل های CAT و در سرعت های بالاتر و مسافت های طولانی از فیبر نوری استفاده می شود.

٤-٩-١. کابل های CAT

کابل های Category با توجه به سرعت انتقال و کاربرد به چند گروه تقسیم می شوند و با نماد CAT به همراه شماره گروه مشخص می گردند. در جداولی که در ادامه آورده می شود انواع کابل های Category و کاربرد هریک را نشان می دهند.

سرعت	رسانای فیزیکی
1 Mb/s	1Base5 : زوج به هم تابیده شده (twisted pair) مورد استفاده برای تلفن
10 Mb/s	10Broad36 : یک کابل broadband
	10Base2 : کابل کواکسیال RG 58
	10Base5 : یک کابل کواکسیال
	10Base-F : یک فیبر نوری
	10Base –T : کابل UTP مدل CAT3 یا بهتر، ارتباط Full-Duplex
100 Mb/s	100Base-FX : دو زوج سیم فیبر نوری، ارتباط Full-Duplex
	100Base-T2 : دو زوج سیم UTP مدل CAT3 یا بهتر، ارتباط Full-Duplex
	100Base-T4 : چهار زوج سیم UTP مدل CAT3 یا بهتر، ارتباط Half-Duplex
	100Base-TX : دو زوج سیم UTP مدل CAT5 یا بهتر، ارتباط Full-Duplex

1000Base-CX : کابل Copper jumper	**1 Gb/s**
1000Base-LX : فیبر با حالت چندگانه/تکی (Multi/Single) با طول موج بلند	
1000Base-SX : فیبر با حالت تکی با طول موج کوتاه	
1000Base-T : چهار زوج سیم CAT5e ، CAT6 یا کابل های بهتر	

جدول ۴-۱۸

ردیف	حداکثر سرعت انتقال اطلاعات	موارد کاربرد
CAT1	1Mbps	سیستم های قدیمی تلفن و مودم
CAT2	4Mbps	شبکه های Token Ring
CAT3	10Mbps	شبکه های Token Ring و 10BaseT
CAT4	16Mbps	شبکه های Token Ring
CAT5	100Mbps	Ethernet در سرعت های 10Mbps و 100Mbps(اترنت سریع) و شبکه های Token Ring
CAT5e	1Gbps	شبکه ها Gigabit Ethernet
CAT6	1Gbps	شبکه ها Gigabit Ethernet
CAT6a	10Gbps	شبکه ها Gigabit Ethernet
CAT7	10Gbps	شبکه ها Gigabit Ethernet

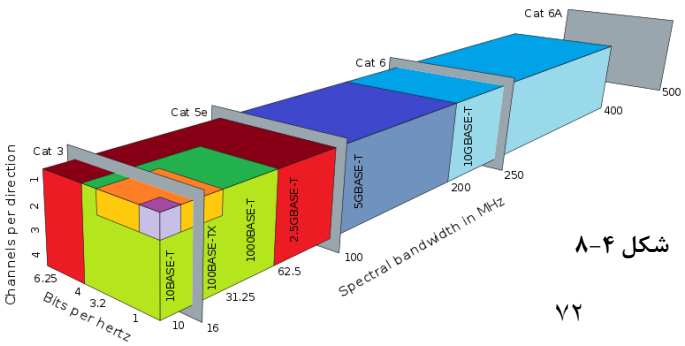
جدول ۴-۱۹

در جدول زیر به طور ویژه در مورد کابل های CAT 5 تا CAT 7 ، فرکانس کاری، نوع شیلدار یا بدون شیلد، نوع کانکتور مورد استفاده و تعداد زوج سیم ها اطلاعاتی ارائه شده است.

	Frequency Supported*	Ethernet Signal Supported	Shielded	Connector	Conductor Pairs
Cat 5	1 - 100MHz	10/100Base T	Optional	8p8c, RJ45	4
Cat5e	1 - 100MHz	10/100Base T, Gigabit Ethernet	Optional	8p8c, RJ45	4
Cat6	1 - 250MHz	10/100Base T, Gigabit Ethernet	Optional	8p8c, RJ45	4
Cat6a	1 - 500MHz	10/100Base T, Gigabit Ethernet, 10Gig Ethernet	Optional	8p8c, RJ45	4
Cat7	1 - 600MHz	10/100Base T, Gigabit Ethernet, 10Gig Ethernet	Individual Pair & Overall Cable Shield	GG45 TERA	4
Cat7a	1 - 1000MHz	10/100Base T, Gigabit Ethernet, 10Gig Ethernet	Individual Pair & Overall Cable Shield	GG45, TERA	4

جدول ۴-۲۰

شکل زیر رابطه ی سرعت و پهنای باند مورد استفاده توسط هر استاندارد و کابل های مناسب برای هر یک را به صورت گرافیکی نشان می دهد.



شکل ۴-۸

نکته ۱ : کابل های CAT3 و سیم های مسی تلفن ها قابل جا به جایی هستند.

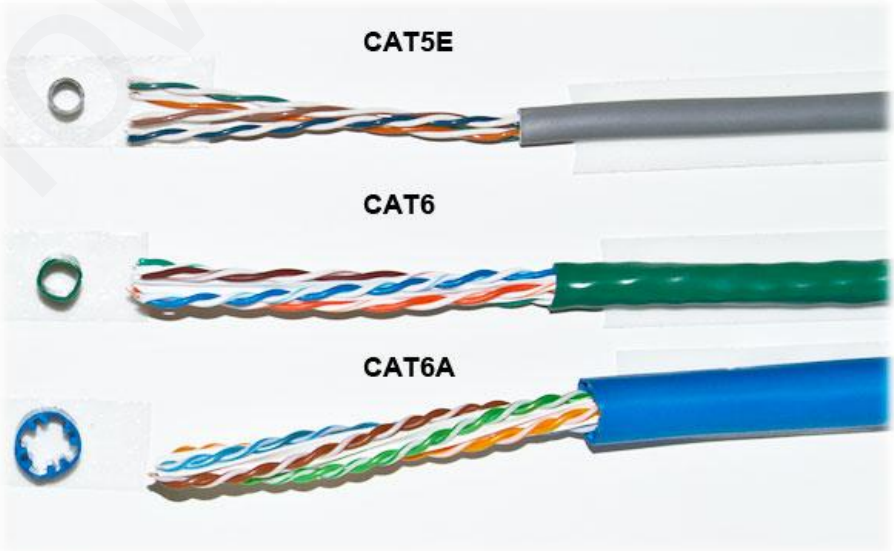
کابل های CAT6 مشابه کابل های CAT5 بوده ولی بین هر کدام از زوج سیم ها از یک جداکننده فیزیکی به منظور کاهش پارازیت های الکترومغناطیسی استفاده شده است.

نکته ۲ : استفاده از کابل های CAT باعث حذف نویز Cross-talk می شود و هرچه پیچش سیم ها در واحد طول بیشتر باشد، امکان استفاده در مسافت های طولانی تر بیشتر می شود (مثلا CAT5 از CAT3 بیشتر دارای پیچش است و در مسافت های طولانی تر استفاده می شود)

نکته ۳ : مشخصات کابل های CAT توسط سازمان های EIA/TIA ارائه می شود.

در کابل های CAT ، دو رشته سیم که در حقیقت یک مسیر رفت و برگشت برای سیگنال محسوب می گردند، طبق قوائد خاصی به هم تابیده می شوند و زوج سیم به هم تابیده را می سازند. این کار برای حذف تداخل الکترومغناطیسی (EMI) حاصل از منابع خارجی مثل سیم های بدون شیلد و از بین بردن شُنود (cross-talk) در سیم هایی که از کنار هم عبور می کند انجام می شود و در نهایت رفتار EMC شبکه را بهبود می بخشد. علت این مسئه این است که به دلیل تابیده شدن سیم ها به یکدیگر، نویز القاء شده بر روی هر دو سیم تقریبا یکسان است و در گیرنده هایی که به شکل تفاضلی (differential) سیگنال این دو سیم را مورد پردازش قرار می دهند، نویز القاء شده بر روی سیم ها حذف خواهد شد.

شکل زیر نمونه ای از کابل های category را نشان می دهد.



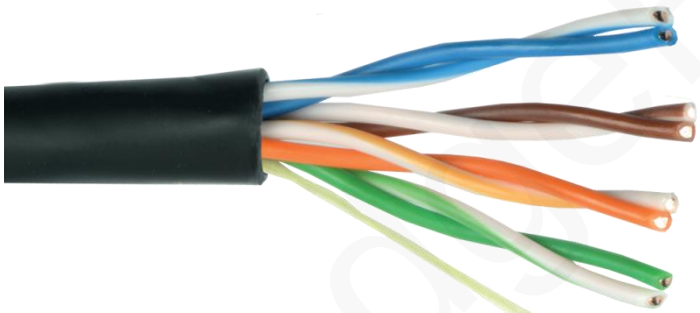
شکل ۴-۹

٤-٩-٢. انواع کابل های محافظ دار

همان طور که گفته شد، حالت تابیده شدن سیم ها به شکل دو به دو با هم باعث کاهش نویز القاء شده و تاثیر سیگنال های ناخواسته خارجی بر روی کابل می شود.

اگر محیطی که در آن کابل ها عبور می کنند جنبه صنعتی داشته باشد و نویزهای منابع خارجی بیشتر باشد، میتوان از کابل های شیلددار (محافظ دار) استفاده نمود. به طور کلی کابل های شیلددار و بدون شیلد دارای تقسیم بندی های زیر می باشند :

١- کابل UTP : ساده ترین و رایج ترین نوع کابل است که در آن جز به هم تابیدن زوج سیم ها، تدبیر دیگری برای کاهش نویز اندیشیده نشده است.

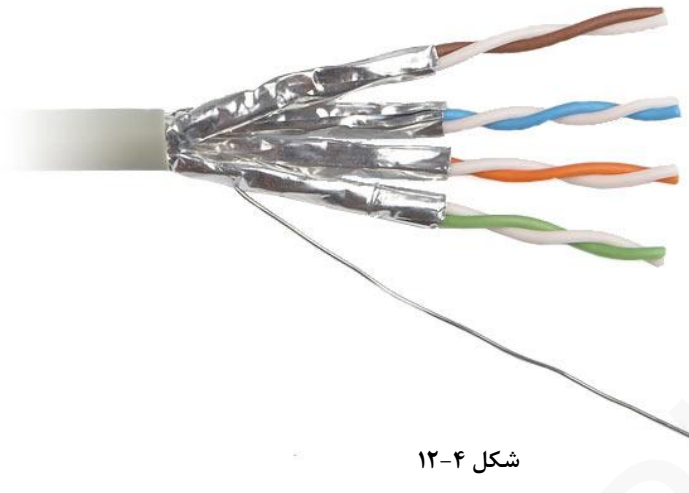


شکل ۴-۱۰

٢- کابل S/UTP : مجموعه ای از چند زوج به هم تابیدن که دارای یک روکش فلزی مشترک می باشند.

شکل ۴-۱۱

۳- کابل STP : مجموعه ای از چند زوج به هم تابیده که هر کدام روکش مجزایی دارند.



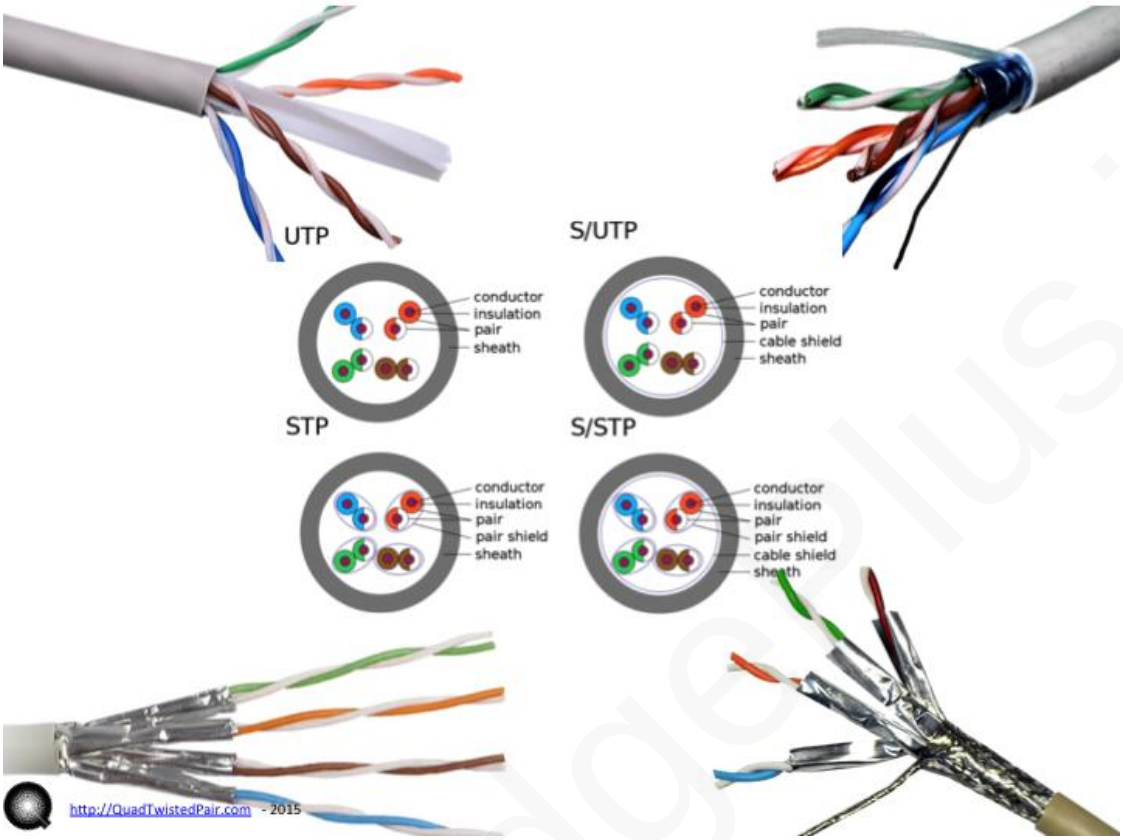
شکل ۴-۱۲

۴- کابل S/STP : مجموعه ای از چند زوج به هم تابیده که هر کدام روکش جداگانه ای دارند و کابل مجموعه نیز دارای یک روکش فلزی مشترک است.

شکل ۴-۱۳

نکته : بدیهی است هرچه تعداد شیلدهای کابل بیشتر باشد، در مقابل نویزهای محیطی مقاومت بیشتری وجود دارد.

شکل زیر این چهار نوع کابل را نشان می دهد.



شکل ۴-۱۴

٤-۹-۳. کد رنگی استاندارد کابل ها

رنگ های سیم های زوج به هم تابیده شده در کابل های CAT از کد رنگی ۲۵ زوج (25-pair color code) تبعیت می کند. در این کد، استاندارد رنگ های مورد استفاده برای ایجاد تمایز میان سیم های موجود درون کابل تعریف شده است.

این کد شامل ۵ رنگ اصلی (major) و ۵ رنگ غیر اصلی (minor) است که ترکیب این رنگ ها ۲۵ کد رنگی را مطابق شکل ۴-۱۵ تشکیل می دهند.

شکل ۴-۱۵

٤-۹-٤. کابل های Patch

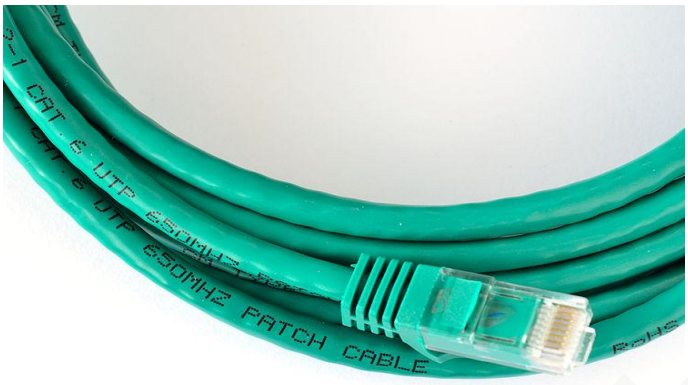
کابل های CAT به دو صورت در بازار وجود دارند :

- کابل flat یا solid (تخت) : برای مسافت های طولانی استفاده می شود، غیر قابل انعطاف
- کابل braided : قابل انعطلاف و برای مسافت های کوتاه

کابل هایی که به عنوان patch cable عرضه می شوند، کابل هایی از جنس braided هستند که اغلب دارای مسافت کوتاه (حداکثر ۲ متر) هستند و برای اتصال دستگاه های مختلف به یکدیگر مورد استفاده قرار می گیرند. (مثلا اتصال کامپیوتر به switch های شبکه)

فرق کابل های patch با کابل های معمولی در انعطاف پذیر بودن آنها می باشد که برای استفاده در مسافت های کوتاه لازم می باشد اما میزان تضعیف سیگنال در این کابل ها از کابل های معمولی بیشتر است، به همین دلیل کاربرد آنها در مسافت های کوتاه می باشد.

در شکل زیر یک کابل CAT6 از نوع patch به همراه کانکتور RJ-45 را که توسط استاندارد T568B به آن متصل شده است را نشان می دهد.



شکل ۴-۱۶

شکل زیر استفاده از کابل های patch را برای اتصال به switch های شبکه نشان می دهد.

شکل ۴-۱۷

نکته : باتوجه به توضیحات ارائه شده، استفاده از کابل های patch که در بازار با اندازه های بلند ساخت کشور چین وجود دارد باعث ایجاد اختلال و مشکل در شبکه می شود. به عنوان مثال اخیرا یک نمونه از همین کابل ها با طول دوازده متر استفاده شد که باعث افت شدید سرعت و به وجود آمدن نویزهای محیطی و قطع و وصل شدن ارتباط شد.

٤-٩-٥. فیبر نوری

همان طور که در بخش های قبلی اشاره شد، از فیبر نوری در سرعت های 100Mbit/s و بالاتر استفاده می شود. مهم ترین دلایل استفاده از فیبر نوری عبارت است از امکان استفاده در مسافت های طولانی، ضریب تضعیف کمتر و ایزولاسیون الکتریکی بهتر که باعث جلوگیری از انتشار نویز در محیط و تاثیرپذیری سیگنال های کابل از نویزهای محیطی می باشد. استفاده از فیبر نوری از کابل های شبکه در برابر نویزهای تولید شده توسط آسانسورها در ساختمان ها جلوگیری می کند.

در استفاده از فیبرهای نوری، مقداری حداقلی برای مسافت وجود دارد که به علت سطح ولتاژ های دریافت شده تعریف شده، نمیتوان از کابل های کوتاه تر استفاده کرد. در صورتی که از فیبرهایی که به منظور مسافت های طولانی تر طراحی شده است برای مسافت های کوتاه استفاده شود، باید از تضعیف کننده سیگنال یا signal attenuator استفاده شود.

شکل زیر کانکتور مربوط به استاندارد 100Base-FX را که باید به فیبر نوری متصل گردد را نشان می دهد.

شکل ۴-۱۸

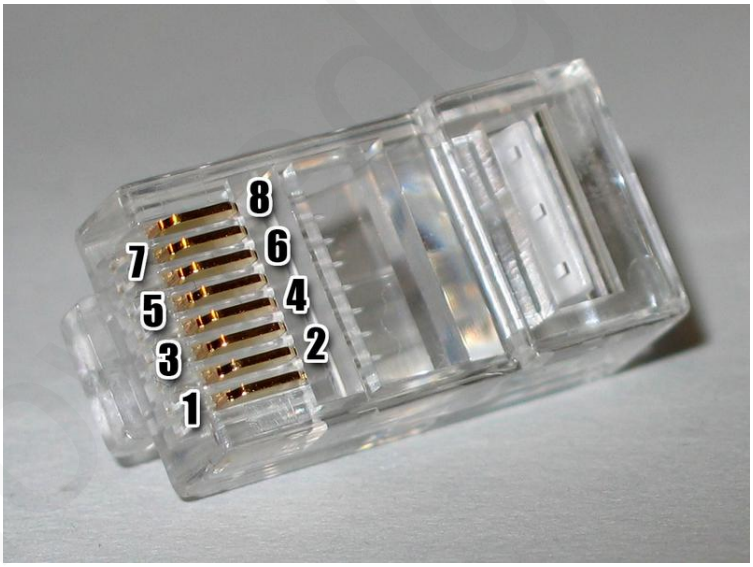
نکته : علاوه بر کابل های CAT و فیبر نوری، می توان از کابل های متفرقه مانند کابل ها مدل Type ساخت شرکت IBM استفاده نمود. (مثلا برای سرعت 100Mbit/s می توان از Type 1 STP استفاده نمود)

٤-١٠. کانکتورهای اترنت

با توجه به نوع کابل مورد استفاده، کانکتوری که باید استفاده شود نیز متفاوت است.

نوع کانکتور را سازمان های EIA/TIA در استاندارد EIA-568 برای کابل های CAT کانکتور RJ-45 و IBM برای کابل های Type نوع DB9 معرفی نموده اند.

شکل زیر یک نمونه کانکتور RJ-45 که یکی از رایج ترین کانکتورهای مورد استفاده در اترنت می باشد را به همراه شماره پایه ها نشان می دهد.



شکل ۴-۱۹

استفاده از نام RJ-45 برای این کانکتور اشتباه است، در واقع اسم اصلی این کانکتور که در شبکه های کامپیوتری و اترنت استفاده می شود 8P8C (8-position 8-contact) می باشد که علت نام گذاری آن تعداد ۸ محل اتصال است. این کانکتور زیر مجموعه ای از کانکتورهای ماژولار (modular connectors) می باشد. (مانند 6P6C ، 4P4C)

مهم ترین کاربرد کانکتورهای ماژولار در سیم های تلفن و در شبکه های اترنت است.

واژه 8P8C عنوان کننده 8 Position و 8 Contact است به این معنا که ۸ محل اتصال وجود دارد و هر ۸ اتصال استفاده می شود.

یک کانکتور 6P2C شامل ۶ محل اتصال است که دو اتصال از آنها در وسط قرار دارد و به آنها سیم متصل می شود ولی ۴ محل اتصال دیگر در گوشه قرار دارند و استفاده نمی شوند.

اولین استفاده از این کانکتورها در استاندارد Registered Jack بود.

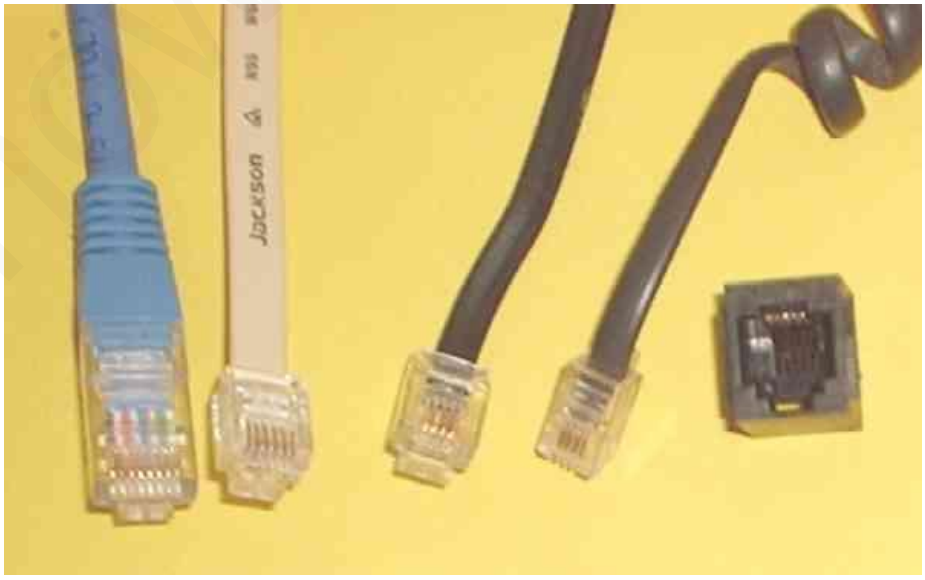
استفاده از واژه RJ-45 در واقع به نوع کاربرد کانکتور در مجموعه استانداردهای RJ که توسط سازمان FCC در سال ۱۹۷۶ تعیین شده است اشاره دارد.

در استانداردهای RJ مانند RJ-45 ، حروف RJ مخفف Registered Jack می باشد که استانداری تعریف شده در حوزه telecommunication به عنوان مدارات واسط می باشد.

استاندارد Registered JacK مشخص کننده ی نوع سیم بندی کانکتورها است و در مورد شکل و اندازه کانکتور یا نوع مادگی و نری آن مشخصاتی ارائه نکرده است.

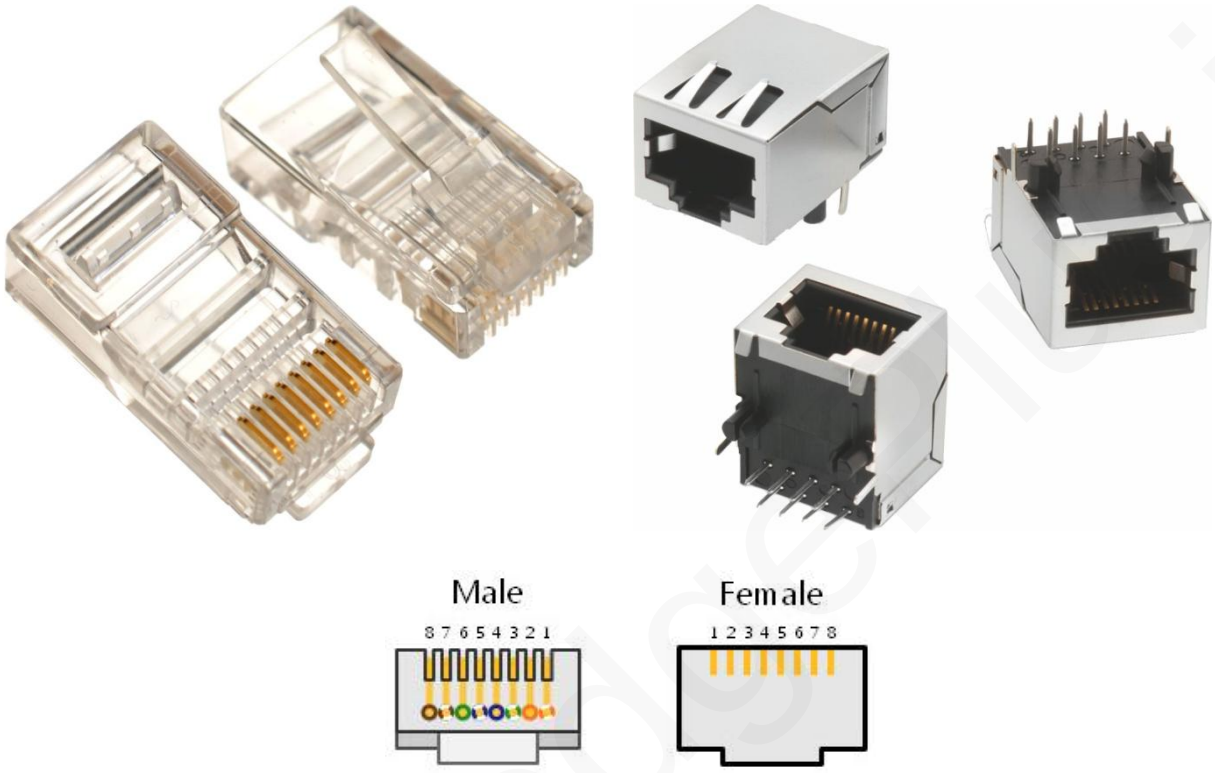
به عنوان مثال کانکتور 8P8C در استانداردهای RJ-45s، RJ-49 ، RJ-61 و دیگر استانداردها کاربرد دارد.

ظاهر این کانکتورها شبیه هم هستند و باید دقت شود که آنها را اشتباه نگیریم. به عنوان مثال در شکل زیر به ترتیب از چپ به راست کانکتورهای ماژولار 8P8C (رایج شده با نام RJ-45) ، 6P6C ، 6P4C ، 4P4C (رایج شده با نام های RJ-9 ، RJ-10 یا RJ-22) و کانکتور مادگی مدل 6P6C نشان داده شده است.



شکل ۴-۲۰

کانکتورهای 8P8C دارای دو مدل نری (plug) و مادگی (Jack) می باشد. مدل مادگی بر روی برد نصب می شود و مدل نری بر روی کابل. در شکل زیر یک مدل کانکتور RJ-45 نری و یک مدل مادگی به همراه شماره پایه های هرکدام را مشاهده می کنید.



شکل ۴-۲۱

برای اتصال کانکتور RJ-45 نری به کابل باید از ابزار آچار پرس (crimper) مناسب استفاده کرد.

شکل ۴-۲۲

نکته ۱ : اگرچه کانکتور RJ-45S (که در کاربردهای تلفن استفاده می شود) دارای شباهت زیادی با کانکتور 8P8C است اما به دلیل تفاوت در بخشی از بدنه این دو، نمی توانند به جای هم استفاده شوند. (در واقع کانکتور 8P8C مدل un-keyed یا بدون دکمه کانکتور RJ-45S می باشد.)

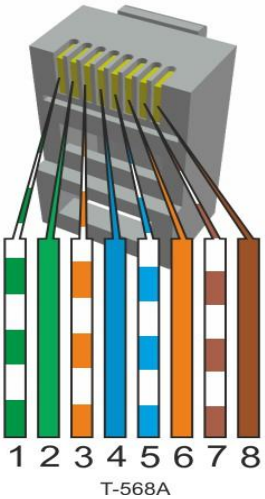
نکته ۲ : در اغلب کاتالوگ ها و یا دیتاشیت ها نیز علارغم اشتباه بودن، از همان نام RJ-45 استفاده می شود و در بازار نیز به همین نام شناخته می شود.

روش های اتصال کانکتور به کابل در بخش بعدی توضیح داده خواهد شد.

٤-١١. استانداردهای TIA/EIA 568A/B ، کابل های Cross-Over و Straight-Thtough

بخشی از مشخصات لایه فیزیکی اترنت توسط سازمان EIA/TIA (Electronic Industries Association and Telecommunications Industry Association) تعیین شده است.

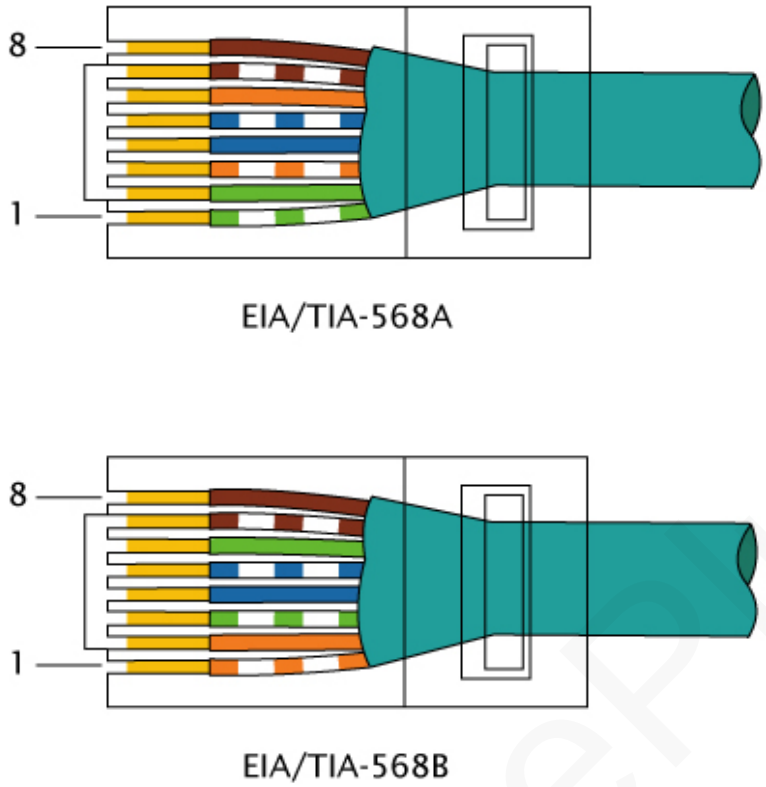
برای اتصال سیم های کابل CAT به کانکتور RJ-45 دو استاندارد TIA/EIA 568A و TIA/EIA 568B وجود دارد. اگرچه هر دو شبیه هم هستند ولی امروزه استاندارد TIA/EIA 568B بیشتر استفاده می شود. شکل های ۴-۲۳ و ۴-۲۴ ترتیب اتصال سیم های کابل CAT را به کانکتور RJ-45 مطابق این دو استاندارد و کاربرد هر یک ار سیم ها را در نسخه های مختلف اترنت نشان می دهند.



شکل ۴-۲۳

Pin	Description	10base-T	100Base-T	1000Base-T
1	Transmit Data+ or BiDirectional	TX+	TX+	BI_DA+
2	Transmit Data- or BiDirectional	TX-	TX-	BI_DA-
3	Receive Data+ or BiDirectional	RX+	RX+	BI_DB+
4	Not connected or BiDirectional	n/c	n/c	BI_DC+
5	Not connected or BiDirectional	n/c	n/c	BI_DC-
6	Receive Data- or BiDirectional	RX-	RX-	BI_DB-
7	Not connected or BiDirectional	n/c	n/c	BI_DD+
8	Not connected or BiDirectional	n/c	n/c	BI_DD-

PinoutsGuide.com



شکل ۴-۲۴

طبق استاندارد TIA/EIA-568 (مجموعه استانداردهای تدوین شده توسط سازمان TIA و EIA به منظور ساخت کابل های مورد استفاده در کاربردهای telecommunication یا ارتباط از راه دور)، دو روش برای اتصال کانکتور های RJ-45 به کابل شبکه وجود دارد :

اتصال straight-through : از این کابل برای اتصال گره های مختلف شبکه به تجهیزات شبکه مانند سوییچ استفاده می شود.

اتصال Cross-over : از این کابل برای اتصال دو گره اترنت بدون واسط استفاده می شود، مثلا اتصال دو کامپیوتر، اتصال دو مدار میکروکنترلری، اتصال یک مدار میکروکنترلری به کامپیوتر، اتصال دو هاب به یکدیگر (بدون استفاده از پورت uplink)

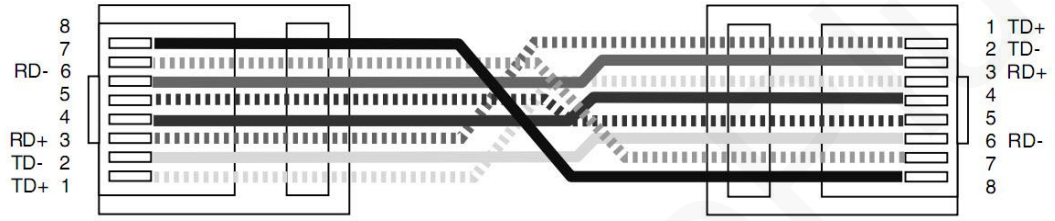
تنها تفاوت این دو روش جا به جایی رنگ های نارنجی و سبز (سیم فرستنده (TX) و سیم گیرنده (RX)) است. (در واقع پایه فرستنده (TX) یک طرف به پایه گیرنده (RX) طرف دیگر متصل می شود، مشابه ارتباط USART)

برای ساخت کابل straight-through باید در هر دو سر کابل از یک استاندارد استفاده کرد.

برای ساخت کابل Cross-over باید در یک سر کابل از استاندارد 568 A و در سر دیگر کابل از استاندارد 568 B استفاده شود. (جا به جایی سیم های TX و RX)

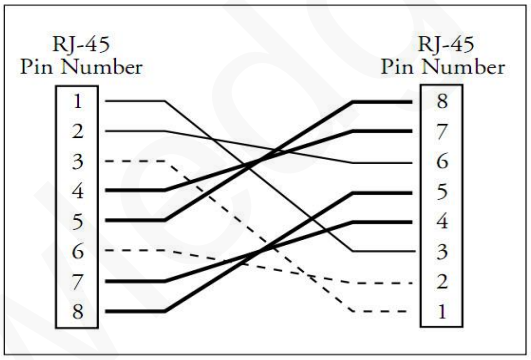
شکل ها و جدول زیر نحوه ساخت کابل های straight-through و Cross-over را نشان می دهند.

10/100Base-T Straight-through cable



10/100Base-T Crossover cable

شکل ۴-۲۵



Cross Connect Cable (Repeater Eliminator) for 10BASE-T, 100BASE-T, and 100BASE-T4.

شکل ۴-۲۶

10/100 Base-T cable wire color code

Straight-through Cable (Both Ends)	Crossover Cable (One End)
1 – White/Orange	1 – White/Green
2 – Orange	2 – Green
3 – White/Green	3 – White/Orange
4 – Blue	4 – Blue
5 – White/Blue	5 – White/Blue
6 – Green	6 – Orange
7 – White/Brown	7 – White/Brown
8 – Brown	8 – Brown

جدول ۴-۲۱

نکته : برای تشخیص این مسئله که از کدام نوع کابل استفاده کنیم، اگر یکی از پوروت های سوییچ یا هاب حرف X داشت، از کابل straight-through و اگر هر دو پورت X داشت یا هیچ کدام نداشت از کابل Cross-over استفاده شود.

علاوه بر این دو نوع کابل، نوع دیگری از کابل ها به نام Rolled over وجود دارد. در این کابل ها سیم ها در دو جهت به شکل مخالف قرار داده شده اند یعنی مثلا پایه ۱ در یک طرف به پایه ۸ در طرف دیگر متصل می شود، پایه ۲ در یک طرف به پایه ۷ در طرف دیگر و به همین ترتیب. از کاربردهای این کابل ها برای اتصال سوییچ ها یا روترها با کانکتور RJ-45 به پورت سریال کامپیوتر با کانکتور DB9 می باشد.

AUTO-CROSSOVER .۱۲-٤

همان طور که در بخش قبل توضیح داده شد، در اتصالات کابل های شبکه اترنت، پایه ی TX یک گره به پایه RX گره دیگر می خورد و بعکلس.

در شبکه های اترنت که از توپولوژی ستاره استفاده می کنند، این جا به جایی در switch ، hub یا router اتفاق می افتد، در نتیجه در چنین شرایطی اتصالات دو سر کابل به شکل یک به یک هست و از کابل straight-through استفاده می شود.

کابل های دیگری وجود دارد به نام کابل cross-over که این اتصالات درون کابل انجام شده است. از این کابل برای اتصال دو گره مستقیما به یکدیگر یا اتصال دو switch ، hub یا router استفاده می شود. البته اگر اتصالات اشتباه باشد به گره های شبکه آسیبی وارد نمی شود و فقط ارتباط برقرار نمی شود.

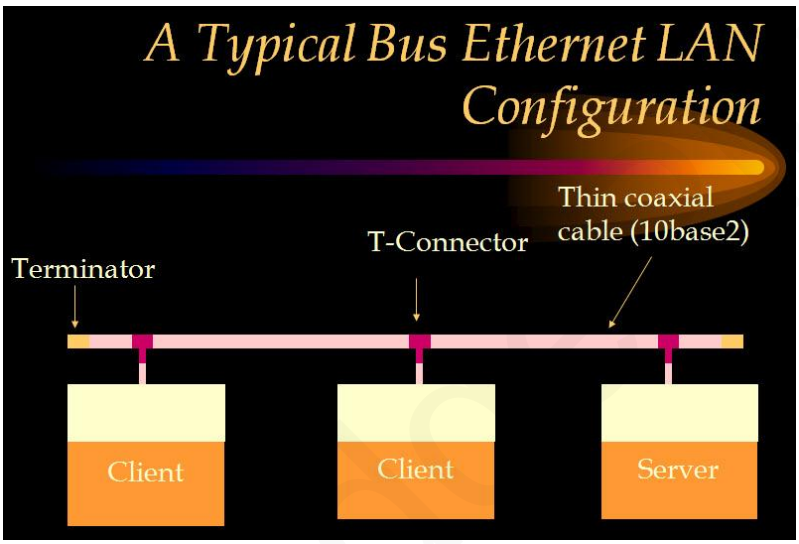
امروزه در اغلب کارت های شبکه و تجهیزات شبکه مانند switch ها و router ها و... قابلیتی قرار گرفته است به نام Auto-Crossover یا Auto-MDIX که به شکل خودکار اتصالات را تغییر می دهد تا ارتباط برقرار شود و در این صورت می توان از هر دو کابل برای اتصال استفاده نمود، در اینصورت کافی است یکی از طرفین ارتباط از این قابلیت پشتیبانی کند.

نکته : توجه شود که این قابلیت با auto-polarity که در آن گره پلاریته ی سیگنال TX یا RX را تغییر می دهد اشتباه گرفته نشود. این دو قابلیت با یکدیگر فرق دارند و کاربردهای متفاوتی دارند.

٤-١٣. پیاده سازی استانداردها

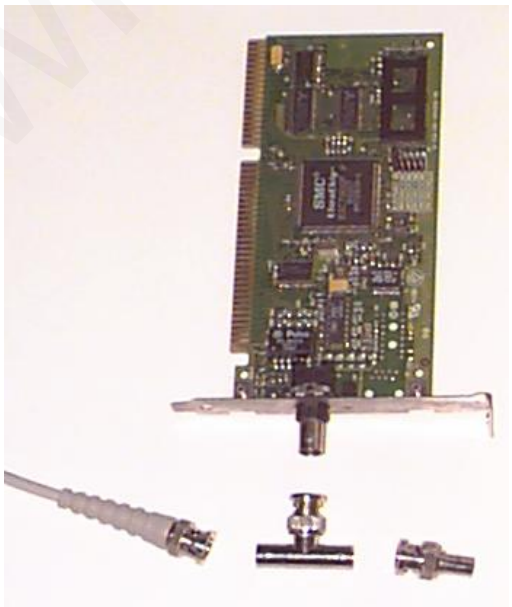
در این بخش طبق استانداردها، کابل ها و کانکتورهایی که در بخش های قبل توضیح داده شد، به توضیح چند نوع پیاده سازی ساده از استانداردهای ارائه شده می پردازیم.

پیاده سازی استاندارد 10Base-2 توسط توپولوژی خطی صورت می پذیرد. برای اتصال گره های شبکه از کابل هم محور Thin ، کانکتورهای T شکل و خاتمه دهنده (Terminator) استفاده شده است.



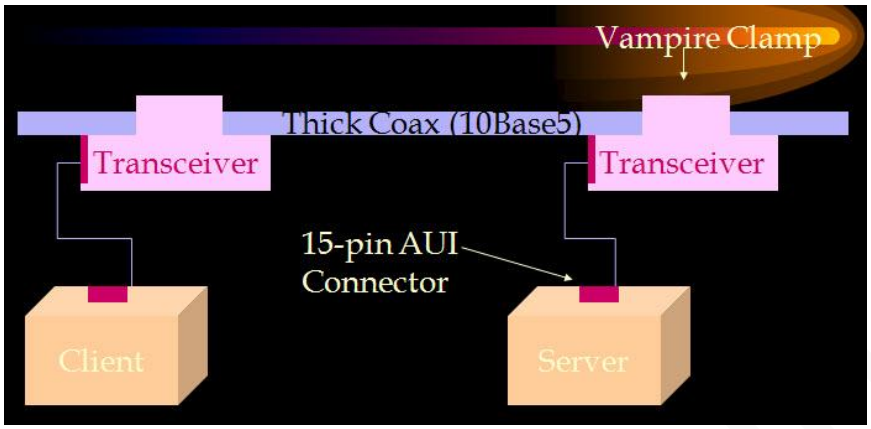
شکل ۴-۲۷

شکل زیر اتصال کارت شبکه را به کانکتور T شکل و کابل شبکه نشان می دهد.



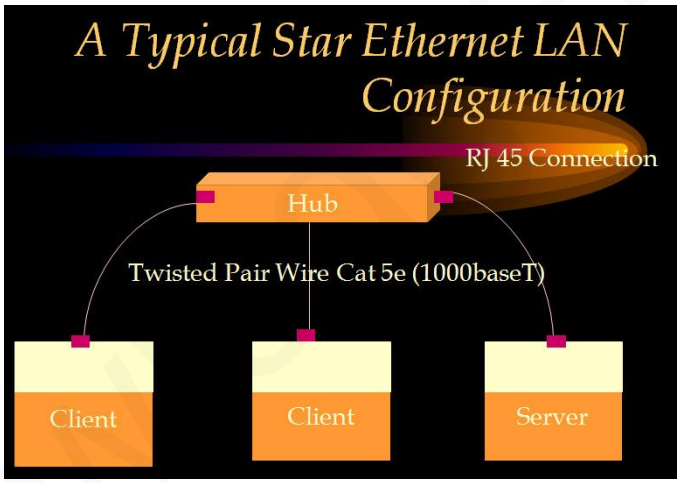
شکل ۴-۲۸

پیاده سازی استاندارد 10Base-5 توسط توپولوژی ستاره صورت می پذیرد. کابل مورد استفاده هم محور Thick می باشد و از بست های Vampire (Vampire Clamp) استفاده می شود.



شکل ۴-۲۹

پیاده سازی استاندارد 1000Base-T توسط توپولوژی ستاره صورت می پذیرد. برای اتصال گره های شبکه از کابل CAT5e و کانکتورهای RJ-45 می شود.

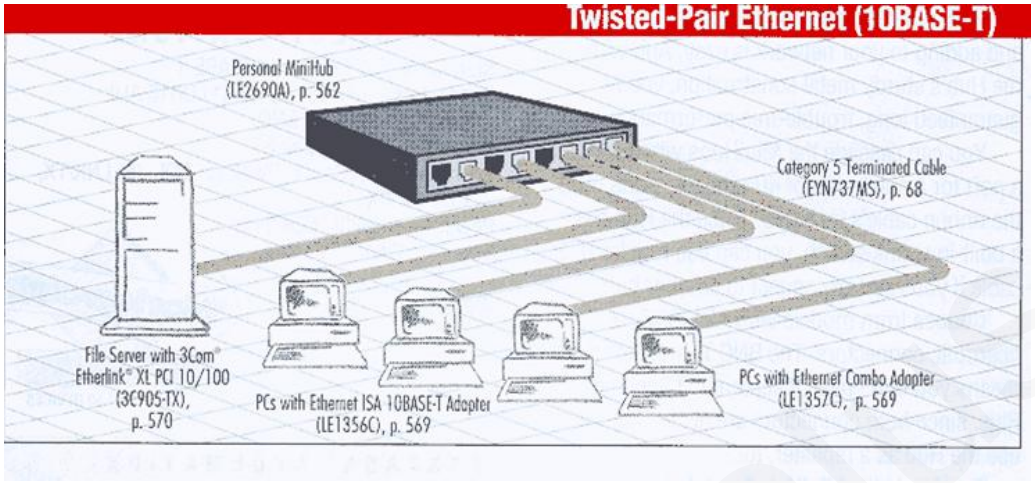


شکل ۴-۳۰

پیاده سازی استاندارد 10Base-T توسط توپولوژی ستاره، کابل های STP یا UTP و کانکتور RJ-45 صورت می پذیرد.

شکل ۴-۳۱

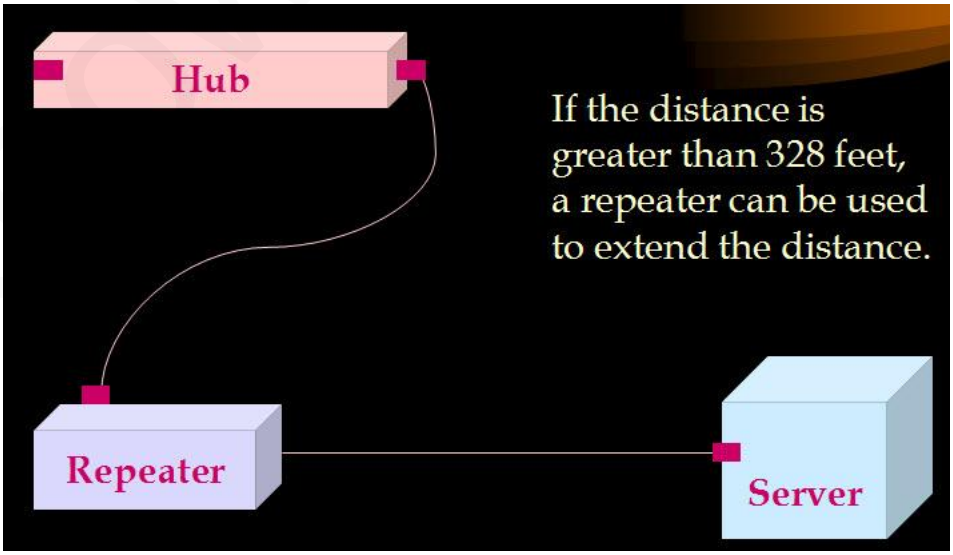
شکل زیر پیاده سازی عملی این شبکه را نشان می دهد.



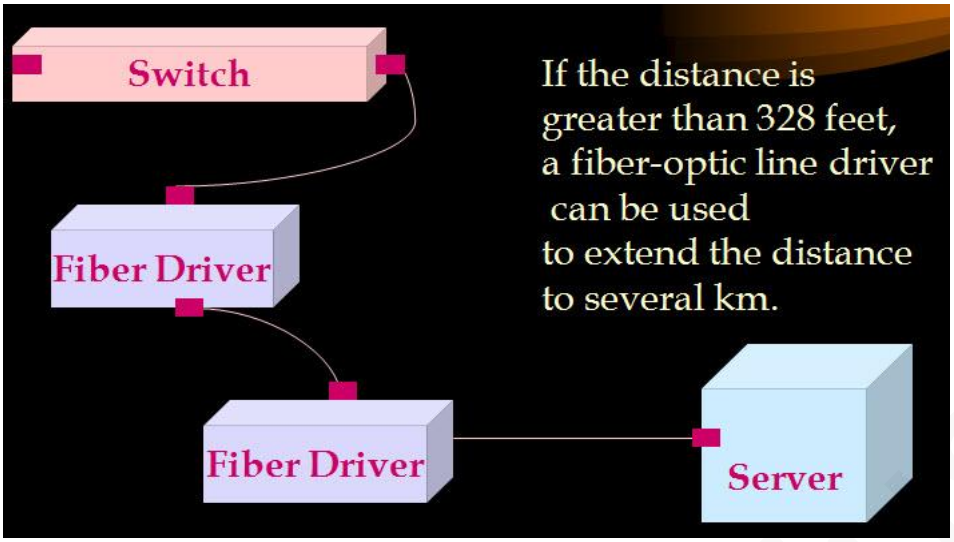
شکل ۴-۳۲

در این مثال، شبکه دارای ۵ گره می باشد. ۴ تا از گره ها کامپیوتر شخصی (PC) می باشد که توسط کارت شبکه شرکت Combo و ISA به هاب مرکزی متصل شده اند. گره پنجم یک کامپیوتر با نقش سرور در شبکه است که توسط کارت شبکه از شرکت 3Com به Hub متصل شده است. کابل مورد استفاده مدل CAT5 همراه با termination (خاتمه سازی) می باشد.

با توجه به این که حداکثر طول کابل میتواند ۳۲۸ متر باشد، برای افزایش مسافت و تقویت دامنه سیگنال (که در اثر افزایش طول کابل و مقاومت مسیر از مقدار مجاز کمتر شده است) می توان از repeater یا کابل های فیبر نوری و fiber driver استفاده نمود.

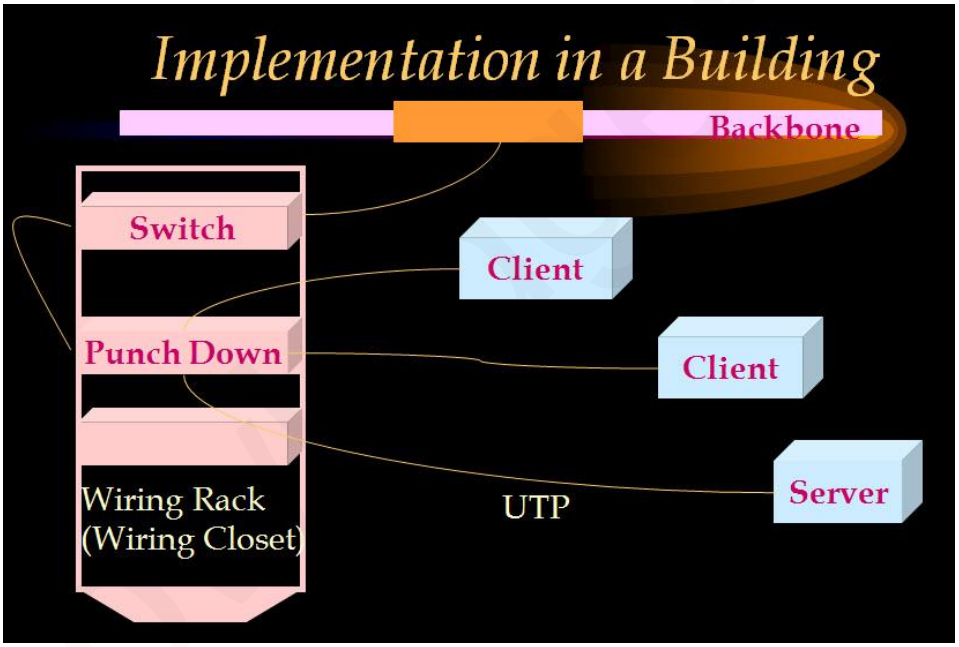


شکل ۴-۳۳



شکل ۴-۳۴

شکل زیر پیاده سازی شبکه را در ساختمان نشان می دهد.



شکل ۴-۳۵

نکته : برای پیاده سازی شبکه با توپولوژی ستاره در ساختمان، اجزای شبکه مانند Hub ، Switch ، Repeater و بقیه تجهیزات درون کشوهای rack و درون یک کمد مخصوص قرار می گیرند و تمام گره ها توسط رابط سیم مناسب به این تجهیزات متصل می شوند.

٤-١٤. فریم های اترنت

اندازه یک فریم اترنت محدود بوده و دارای یک حداقل و حداکثر است و مجموعه ای از اطلاعات ضروری و مورد نیاز می بایست در فریم وجود داشته باشد. یک فریم باید دارای آدرس مبدا و مقصد باشد.

دو دستگاه متفاوت در شبکه نمیتوانند دارای آدرس مشابه باشند. تمام گره های موجود در شبکه، فریم های ارسالی در طول شبکه را دریافت می کنند و ابتدا آدرس MAC مقصد آن فریم را بررسی میکنند، اگر آدرس مربوط به آنها باشد، آن را بررسی می کنند و در غیر اینصورت فریم را بدون بررسی محتوایات آن کنار خواهند گذاشت.

یکی از نکات قابل توجه در آدرس دهی در شبکه های اترنت، قابلیت استفاده از آدرس نوع Broadcast (پخش عمومی) است. زمانی که آدرس MAC مقصد یک گره از نوع Broadcast باشد، تمامی گره های موجود در شبکه فریم را دریافت و مورد پردازش قرار می دهند.

٤-١٤-١. ساختار پایه فریم ها

قالب پایه فریم ها برای اترنت نسخه های 10Mb/s و 100Mb/s در شکل زیر نشان داده شده است که در ادامه به توضیح بخش های مختلف آن می پردازیم.

BASIC FRAME FORMAT

	10/100 IEEE 802.3™ Frame
7 octets	Preamble
1 octet	Start Frame Delimiter (SFD)
6 octets	Destination Address (DA)
6 octets	Source Address (SA)
2 octets	Length (≤ 1500) Type (≥ 1536)
46 octets to 1500 octets	Client Data (Payload)
	Pad (if necessary)
4 octets	Frame Check Sequence (FCS)

شکل ۴-۳۶

نکته : فریم های اترنت در سرعت ها و استانداردهای مختلف اترنت با هم تفاوت دارند که در ادامه توضیح

داده خواهد شد.

Preamble (مقدمه) : شامل ۷ بایت است. در سرعت 100Mb/s اولین بایت شامل 4B/5B کد شده به صورت /J/K/ می باشد که به تعیین کننده شروع جریان اطلاعات (SSD) شناخته می شود. کاربرد Preamble برای گیرنده ایجاد قابلیت قفل کردن روی جریان داده قبل از رسیدن فریم ها می باشد.

SFD : برابر عدد '10101011b' می باشد. گاهی اوقات این بخش جزئی از Preamble شناخته می شود و به همین دلیل Preamble را ۸ بایت درنظر می گیرند.

DA : شامل ۶ بایت برای آدرس MAC سخت افزاری مقصد می باشد.

SA : شامل ۶ بایت برای آدرس MAC سخت افزاری فرستنده می باشد.

Length/Type : اگر عدد موجود در این بخش کوچکتر یا مساوی عدد ۱۵۰۰ (دسیمال) باشد نشان دهنده ی تعداد بایت های بخش Payload (داده کاربر) می باشد و اگر این عدد بزرگتر یا مساوی عدد ۱۵۳۶ باشد، نشان دهنده ی نوع Ether یا نوع داده های کاربر می باشد. رایج ترین انواع Ether در جدول زیر نشان داده شده است.

0800h	IPv4
86DDh	IPv6
0806h	ARP
8035h	RARP

جدول ۴-۲۲

(Payload) Client Data : شامل داده های کاربر و بخش های سرآیند پروتکل می باشد. حداقل طول این بخش برابر ۴۶ بایت است و حداکثر آن برابر ۱۵۰۰ بایت است که البته ممکن است مقادیری خارج از این محدوه را برخی تولیدکنندگان درنظر بگیرند که در این صورت خارج استاندارد IEEE 802.3 می باشد.

Pad : از آنجایی که حداقل طول بخش Payload برابر ۴۶ بایت است، در صورتی که از این مقدار کمتر باشد، باید از Pad برای تکمیل کردن آن استفاده کرد.

FCS : این بخش طبق اطلاعات موجود در بخش های دیگر و کدهای خطایابی مانند کد ۳۲ بیتی CRC محاسبه می شود.

ESD (End-of-Stream Delimiter) : در سرعت 100 Mb/s ، لایه ی فیزیکی (PHY) بعد از FCS یک سمبل /T/R/ به منظور نشان دادن پایان یافتن فریم ارسال می کند.

در سرعت 10 Mb/s یک سیگنال ویژه TP_IDL و سکوت در شبکه (network silence به معنای عدم ارسال اطلاعات در شبکه) نشان دهنده ی پایان یافتن فریم است.

نکته ۱ : سمبل /T/R و سیگنال TP_IDL جزء فریم محسوب نمی شوند و به همین دلیل در شکل بالا نمایش داده نشده اند.

علاوه بر فریم پایه توضیح داده شده در سرعت های 10 Mb/s و 100 Mb/s ، دو نوع فریم رایج دیگر به نام های فریم کنترلی (Control Frame) و فریم مجازی LAN (VLAN tagged) وجود دارد که در ادامه توضیح می دهیم.

٤-١٤-٢. فریم های کنترلی (Control Frames)

فریم های با عدد 8808h در قسمت نوع Ether ، به عنوان فریم های کنترلی MAC شناخته می شوند و برای کنترل جریان فریم ها در ارتباط به کار می روند. پیاده سازی ویژگی های کنترلی MAC در گره های شبکه اترنت دلخواه است.

اولین بایت از فریم های کنترلی MAC شامل opcode است. در حال حاضر تنها فریم کنترلی استاندارد، فریم pause یا توقف است که دارای opcode و آدرس مقصد زیر می باشد :

Opcode : 0001h
Address : 01-80-c2-00-00-01 (Multicast آدرس)

فریم pause باعث ایجاد درخواست از گره ی فرستنده اطلاعات برای توقف موقت ارسال اطلاعات به مدت دو بایت پس از opcode می شود.

ارسال فریم pause با عدد 0000h باعث لغو درخواست توقف ارسال اطلاعات قبلی می شود.

٤-١٤-٣. فریم های VLAN Tagged

شبکه ی محلی مجازی یا VLAN (Virtual Local Area Network) باعث ایجاد اطلاعات اضافه بر روی فریم ها برای ایجاد شبکه های مبتنی بر توپولوژی های منطقی به جای توپولوژی های فیزیکی می شود.

٤-١٤-٤. ظرفیت فریم ها

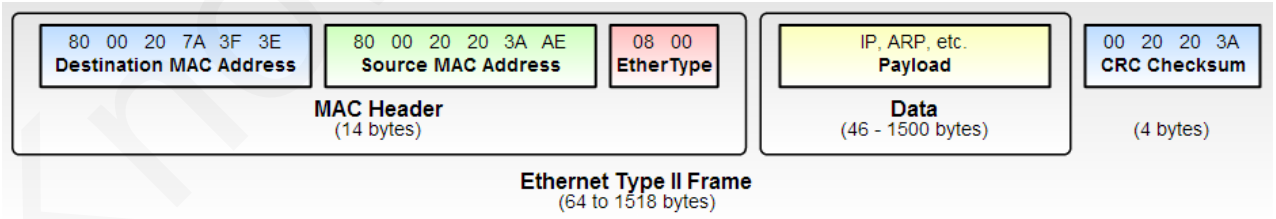
در استاندارد IEEE 802.3 معمولا بخش Preamble و SFD جزو ظرفیت فریم محسوب نمی شود، بنابراین حداقل طول برای فریم های پایه و کنترل ۶۴ بایت و حداکثر طول فریم برابر ۱۵۱۸ بایت است. حداکثر اندازه فریم های VLAN tagged برابر ۱۵۲۲ بایت است.

به فریم های با اندازه کمتر از ۶۴ بایت به اصطلاحا فریم های "runt" می گویند.

به فریم های با اندازه بیش از ۱۵۱۸ بایت، فریم های "long" یا "huge" گفته می شود. در 10/100Base-T به این فریم ها و همچنین به فریم های با اندازه بیش از ۹۰۰۰ بایت در gigabit Ethernet ، اصطلاح "Jumbo" داده می شود.

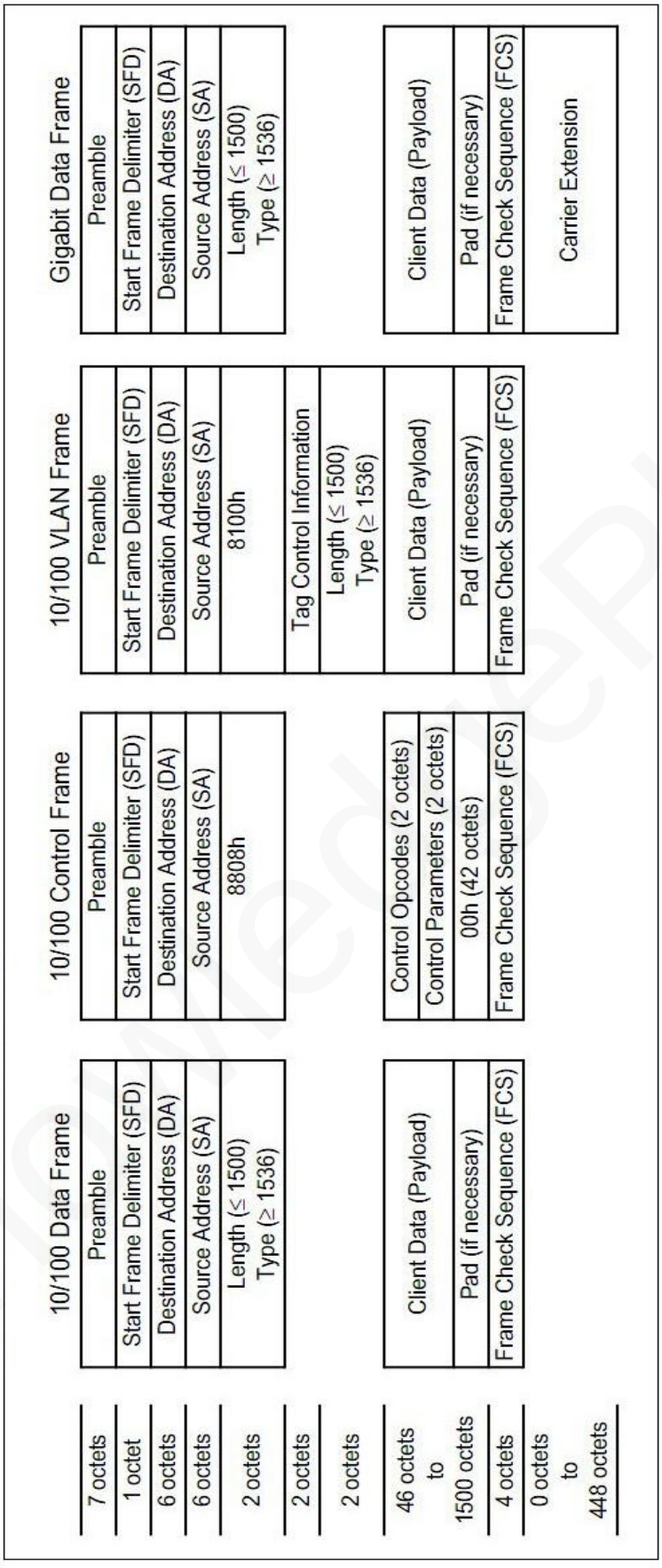
به فریم های بیشش از ۶۰۰۰ بایت اصلاح "giant" گفته می شود.

گاهی اوقات در برخی مقالات منظور از طول فریم، اندازه بخش payload است، بنابراین مثلا ممکن است به فریم های با ظرفیت ۱۵۰۰ بایت فریم Jumbo گفته شود.



شکل ۴-۳۷

در شکل ۴-۳۸ مقایسه ای بین سه نوع فریم معرفی شده و فریم های gigabit Ethernet انجام شده است.


COMMON ETHERNET FRAME TYPES
10/100 Data Frame
Preamble
Start Frame Delimiter (SFD)
Destination Address (DA)
Source Address (SA)
Length (≤ 1500)
Type (≥ 1536)
Client Data (Payload)
Pad (if necessary)
Frame Check Sequence (FCS)
10/100 Control Frame
Preamble
Start Frame Delimiter (SFD)
Destination Address (DA)
Source Address (SA)
8808h
Control Opcodes (2 octets)
Control Parameters (2 octets)
00h (42 octets)
Frame Check Sequence (FCS)
10/100 VLAN Frame
Preamble
Start Frame Delimiter (SFD)
Destination Address (DA)
Source Address (SA)
8100h
Tag Control Information
Length (≤ 1500)
Type (≥ 1536)
Client Data (Payload)
Pad (if necessary)
Frame Check Sequence (FCS)
Gigabit Data Frame
Preamble
Start Frame Delimiter (SFD)
Destination Address (DA)
Source Address (SA)
Length (≤ 1500)
Type (≥ 1536)
Client Data (Payload)
Pad (if necessary)
Frame Check Sequence (FCS)
Carrier Extension
7 octets
1 octet
6 octets
6 octets
2 octets
2 octets
2 octets
46 octets to 1500 octets
4 octets
0 octets to 448 octets

شکل ۴-۳۸

۱۵-٤. ارتباط Half-Duplex و پروتکل CSMA/CD

در اترنت برای جلوگیری از ایجاد تداخل در دسترسی همزمان گره ها به شبکه، از پروتکل CSMA/CD یا CSMA/CA استفاده می شود.

تکنولوژی CSMA/CD (Carrier sense multiple access with collision detection) به معنی دستیابی چندگانه با قابلیت شنود سیگنال حامل/پیشگیری از تصادم، مسئولیت تشریح و تنظیم نحوه ارتباط گره ها با یکدیگر را به عهده دارد.

با اینکه واژه فوق پیچیده به نظر می آید ولی با تقسیم نمودن واژه فوق به بخش های کوچکتر، می توان با نقش هر یک از آنها سریعتر آشنا گردید. به منظور شناخت تکنولوژی فوق مثال زیر را در نظر بگیرید.

فرض کنید سگمنت اترنت، مشابه یک میز ناهارخوری باشد. چندین نفر دور تا دور میز نشسته و به گفتگو مشغول می باشند. واژه multiple access (دستیابی چندگانه) بدین مفهوم می باشد که زمانی که یک ایستگاه اترنت اطلاعاتی را ارسال می دارد، تمام ایستگاههای دیگر موجود در محیط انتقال نیز از انتقال اطلاعات آگاه خواهند شد. (نظیر صحبت کردن یک نفر در میز ناهار خوری و گوش دادن سایرین)

فرض کنید که شما نیز بر روی یکی از صندلی های میز ناهار خوری نشسته و قصد حرف زدن را داشته باشید، در همان زمان فرد دیگری در حال سخن گفتن است، در این حالت می بایست شما در انتظار اتمام سخنان گوینده باشید. در پروتکل اترنت وضعیت فوق carrier sense (شنود سیگنال) نامیده می شود. قبل از اینکه گره ای قادر به ارسال اطلاعات باشد می بایست گوش خود را بر روی محیط انتقال گذاشته و بررسی نماید که آیا محیط انتقال آزاد است؟ در صورتیکه صدایی از محیط انتقال به گوش ایستگاه متقاضی ارسال اطلاعات نرسد، گره مورد نظر قادر به استفاده از محیط انتقال و ارسال اطلاعات خواهد بود.

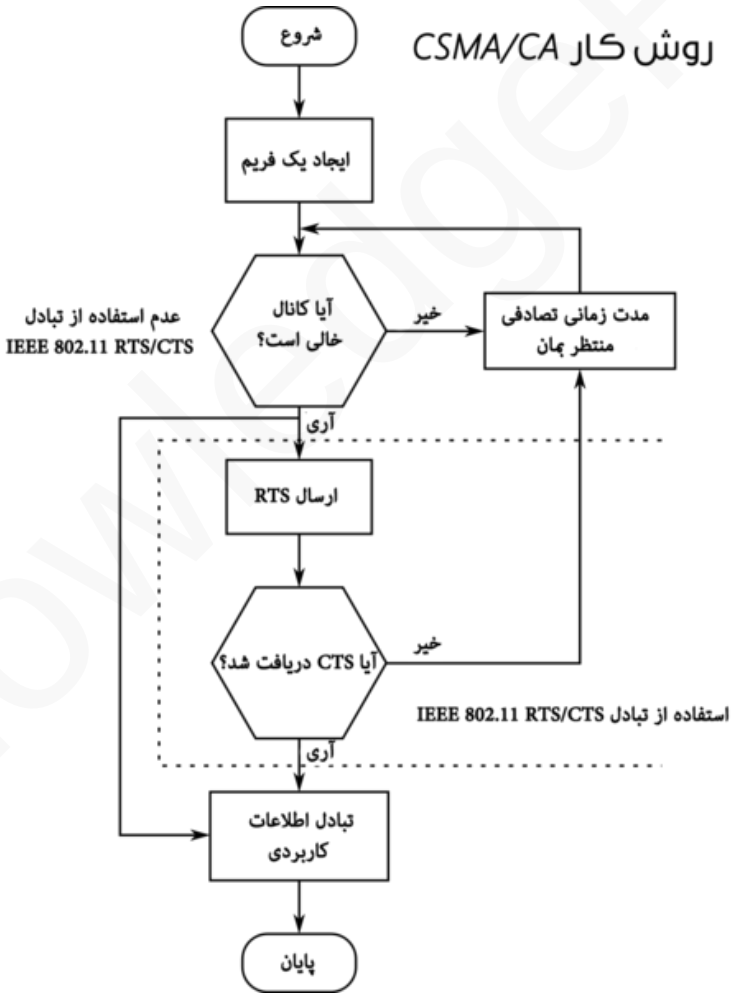
Carrier-sense multiple access شروع یک گفتگو را قانونمند و تنظیم می نماید ولی در این رابطه یک نکته دیگر وجود دارد که می بایست برای آن نیز راهکاری اتخاذ شود. فرض کنید در مثال میز ناهار خوری در یک لحظه سکوتی حاکم شود و دو نفر نیز قصد حرف زدن را داشته باشند. در چنین حالتی در یک لحظه سکوت موجود توسط دو نفر تشخیص و بلافاصله هر دو تقریبا در یک زمان یکسان شروع به حرف زدن می نمایند. چه اتفاقی خواهد افتاد؟

در اترنت پدیده فوق را تصادم (Collision) می گویند و زمانی اتفاق خواهد افتاد که دو گره قصد استفاده از محیط انتقال و ارسال اطلاعات را بصورت همزمان داشته باشند. در گفتگوی انسان ها، مشکل

فوق را می توان بصورت کاملا دوستانه حل نمود. ما سکوت خواهیم کرد تا این شانس به سایرین برای حرف زدن داده شود، همانگونه که در زمان حرف زدن من، دیگران این فرصت را برای من ایجاد کرده بودند. گره های اترنت زمانی که قصد ارسال اطلاعات را داشته باشند، به محیط انتقال گوش فرا داده تا به این اطمینان برسند که تنها گره موجود برای ارسال اطلاعات می باشند. در صورتی که گره های ارسال کننده اطلاعات متوجه نقص در ارسال اطلاعات خود گردند، از بروز یک تصادم در محیط انتقال آگاه خواهند گردید.

در زمان بروز تصادم ، هر یک از گره های مربوطه به مدت زمانی کاملا تصادفی در حالت انتظار قرار گرفته و پس از اتمام زمان انتظار می بایست برای ارسال اطلاعات شرط آزاد بودن محیط انتقال را بررسی نمایند. توقف تصادفی و تلاش مجدد، یکی از مهمترین بخش های پروتکل است.

فلوچارت زیر الگوریتم کار این پروتکل را نشان می دهد.



شکل ۴-۳۹

با توجه به توضیحاتی که داده شد، در ادامه به بررسی دقیق تر مسائل زمان بندی در پروتکل اترنت می

پردازیم.

اترنت اساسا به عنوان پروتکلی طراحی شد که در یک توپولوژی با رسانه ی مشترک مانند کابل هم محور و توپولوژی خطی عمل کند. در این توپولوژی همه ی گره ها دسترسی یکسانی به شبکه دارند اما در هر لحظه فقط یک گره می تواند در شبکه داده های خود را ارسال کند. (Half-duplex)

ارسال همزمان داده ها روی شبکه باعث ایجاد تداخل در داده ها و از بین رفتن آنها می شود. با توجه به این توضیحات الزامات زیر را برای پروتکل اترنت می توان در نظر گرفت :

- گره های مختلف باید بتوانند به شبکه دسترسی داشته باشند (Multiple Access)
- هر گره باید بتواند تشخیص دهد که آیا گره ی دیگری در حال انتقال داده روی شبکه است یا نه؟ (Carrier Sense)
- هر گره باید بتواند تشخیص دهد که آیا زمان ارسال داده های خود متقارن با ارسال داده های گره ی دیگری در شبکه شده است یا خیر (Collision Detect)
- هر گره باید بتواند زمانی را منتظر بماند تا گره های دیگر انتقال داده های خود را انجام دهند و شبکه آزاد شود (Backoff)

این الزامات در اترنت به واسطه ی استفاده از روش CSMA/CD رعایت شده است.

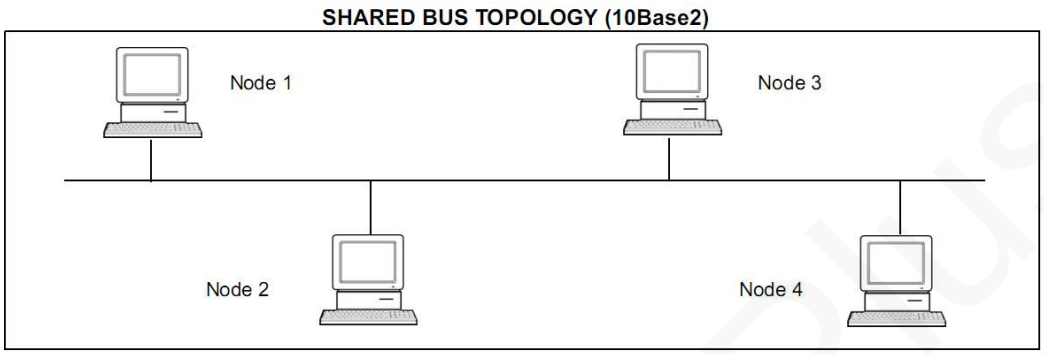
هر گره در شبکه قبل از شروع ارسال باید اطمینان حاصل کند که شبه در وضعیت آزاد (Idle) قرار دارد، اگر اینگونه نبود باید صبر کند تا شبکه آزاده شود و پس از آزاد شدن شبکه، باید به اندازه ی مدت زمانی که از پیش تعریف شده، منتظر بماند. به این مدت زمان IPG (مخفف Inter-Packet Gap یا شکاف بین بسته ها) یا IFG (مخفف Inter-Frame Gap یا شکاف بین فریم ها) گفته می شود و مطابق جدول ۴-۲۳ وابسته به سرعت مورد استفاده درشبکه است.

در واقع IPG زمانی است که به گره های گیرنده داده می شود تا خود را برای دریافت فریم های بعدی آماده کنند.

همان طور که قبلا نیز گفته شد، اگر دو گره به طور همزمان اقدام به ارسال اطلاعات کنند، یک تصادم رخ می دهد. بدترین شرایط (worst case) زمانی رخ می دهد که دو گره که از نظر فیزیکی در مقابل یکدیگر قرار دارند، همزمان اقدم به ارسال داده کنند، به شکلی که یکی از گره ها قبل از این که داده گره ی دیگر را دریافت کند اقدام به عمل ارسال داده می کند.

به عنوان مثال فرض کنید برای دو گره ی ۱ و ۴ در شکل زیر این اتفاق بیفتد. گره ی ۱ اقدام به ارسال داده در شبکه می کند. این داده مدت زمانی طول می کشد تا به گره ی ۴ برسد و گره ۴ قبل از رسیدن این

داده و اطلاع از این که گره ۱ شروع به ارسال داده کرده است، شروع به ارسال داده می کند. گره ۴ بلافاصله تشخیص یک تصادم در شبکه را می دهد و یک سیگنال به نام سیگنال Jam را در شبکه ارسال می کند. این سیگنال باید بلافاصله به گره ۱ برسد تا قبل از این که این گره به مقایسه داده های ارسالی و داده های شبکه بپردازد، از طریق این سیگنال متوجه تصادم در شبکه بشود.



شکل ۴-۴۰

این موضوع در شبکه های 10Base2 ، 10Base5 و 10Base-F که دارای یک رسانه ی مشترک هستند اتفاق می افتد.

به مدت زمانی که طول می کشد تا تمام گره ها از رخ دادن تصادم مطلع شوند و گره فرستنده اطمینان پیدا کند که قابلیت ارسال را دارد، Slot time گفته می شود. Slot time مشخص کننده ی حداکثر تاخیر سیگنال (signal propagation delay) مجاز شبکه است.

به تصادم توضیح داده شده در بالا، تصادم in-window گفته می شود، چون درون زمان slot time رخ داده است. اگر طول کابل شبکه از مقدار مجاز بیشتر باشد، تصادم های out-of-window یا late رخ می دهد. این تصادم ها مانند تصادم های in-window به عنوان خطای انتقال محسوب نمی شوند اما به عنوان خطا در توپولوژی شبکه شناخته می شوند. برخلاف تصادم های in-window ، این تصادم ها توسط لایه ی فیزیکی و پیوند داده مورد بررسی قرار نمی گیرند و باید توسط نرم افزار بررسی شوند.

KEY ETHERNET TIMING PARAMETERS

	Bit Time	IPG Time	Slot Time	Network Diameter (without repeaters)	Network Diameter (with repeaters)
10Base-T	100 ns	9.6 µs	512 Bit Times = 51.2 µs	100m	2500m
100Base-TX	10 ns	960 ns	512 Bit Times = 5.12 µs	100m	205m

جدول ۴-۲۳

با توضیحات بالا مشخص می شود که collision window برابر حداقل اندازه ی فریم است.

با این اندازه ی فریم، حداکثر طول شبکه های gigabit Ethernet حدود ۲۰ متر خواهد بود. برای افزایش این طول در gigabit Ethernet ، طول فریم ها از طریق اضافه کردن بخشی به نام Carrier Extension که در انتهای فریم قرار داده شده است را به ۴۰۹۶ بیت افزایش داده شده است.

از آنجایی که سرعت انتقال در 100Base-T ده برابر 10Base-T است، بنابراین زمان انتقال یک فریم 1/10 است و باعث کاهش slot time از 50us به 5us می شود. بنابراین اندازه شبکه از 2500m به 200m می رسد.

آخرین الزام برای هر پروتکل تعیین زمان ارسال مجدد در صورت ایجاد تصادم است. این مسئله در اترنت توسط روشی به نام binary exponential backoff algorithm انجام می شود که مانند زیر عمل می کند :

هر گره تاخیری تصادفی (در محدوده ی ۰ تا ۱) قبل از اقدام به ارسال مجدد انتخاب می کند.

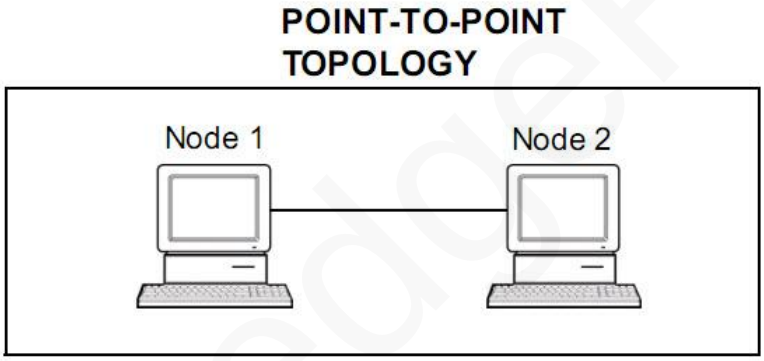
اگر باز هم تصادم رخ داد، اینبار یک تاخیر تصادفی بین محدوده ی ۰ تا ۳ انتخاب می کند.

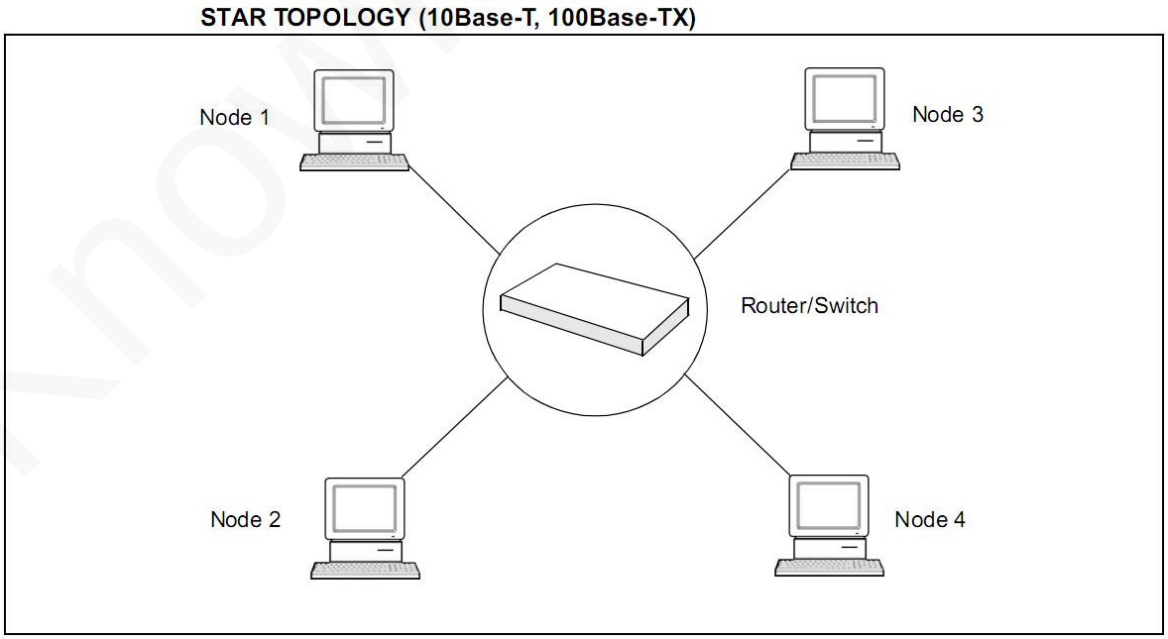
این فرایند ادامه پیدا می کند (از ۰ تا ۷، از ۰ تا ۱۵ و...) تا این که هیچ تصادمی رخ ندهد یا تعداد دفعات تلاش مجدد برای ارسال به ۱۰ بار برسد. در این حالت محدوده ی تاخیر از ۰ تا ۱۰۲۳ می باشد. به این روش، مقدار زمان backoff به صورت نمایی افزایش پیدا می کند و احتمال ایجاد تصادم نیز کاهش پیدا می کند.

۶ تلاش دیگر برای ارسال مجدد انجام می شود (در مجموع ۱۶ تلاش)، اگر باز هم تصادم رخ داد، گره گیرنده فریم را رها می کند و خطای excessive collision را گزارش می دهد؛ در این شرایط نرم افزار باید از این شرایط مطلع شود و اگر لازم است به ارسال مجدد فریم دریافت نشده اقدام کند.

٤-١٦. ارتباط Full-Duplex

در حالی که شبکه های اترنت قدیمی دارای یک رسانه ی مشترک بودند و نیاز به روش CSMA/CD برای جلوگیری از تداخل در ارتباطات شبکه داشتند، نسخه های جدید اترنت مانند شکل های زیر یا دارای ارتباط نقطه به نقطه (point to point) هستند (مانند اتصال دو کامپیوتر به یکدیگر) و یا دارای توپولوژی ستاره (با استفاده از تجهیزاتی مانند switch ها) هستند که در هر دو حالت هر کدام از گره ها دارای یک رسانه ی مجزا هستند و نوع ارتباط آنها Full-Duplex است و احتمال به وجود آمدن تصادم وجود ندارد و در نتیجه نیازی به استفاده از روش CSMA/CD وجود ندارد. هر گره در هر لحظه ای که بخواهد می تواند داده های خود را طبق زمان بندی IPG به شبکه بفرستد.





شکل ۴-۴۱

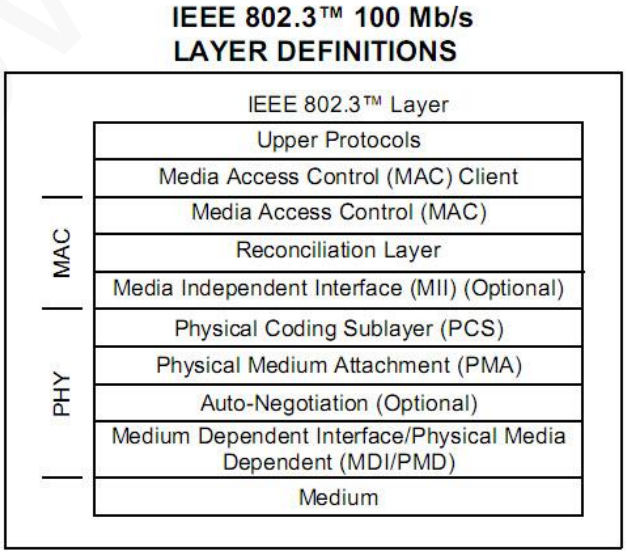
ارتباط Full-Duplex همچنین محدودیت در طول شبکه به دلیل زمان های slot time را از بین می برد. توجه شود که همه ی نسخه های اترنت از ارتباط Full-Duplex پشتیبانی نمی کنند. به عنوان مثال لیست زیر نسخه هایی از اترنت می باشد که از ارتباط Full-Duplex پشتیبانی نمی کنند :

- 10Base2
- 10Base5
- 10Base-FP
- 10Base-FB
- 100Base-T4

٤-١٧. زیرلایه های MAC و PHY

در بخش های اول این فصل در مورد جایگاه پروتکل اترنت در لایه های پیوند داده و فیزیکی در مدل TCP/IP توضیح دادیم. در اترنت اصطلاحاً به لایه پیوند داده، لایه MAC گفته می شود. از استاندارد اترنت سریع (Fast Ethernet) به بعد، برای تشریح بهتر چگونگی عملکرد اترنت و پیاده سازی عملی آن، عملکرد اترنت در این دو لایه به زیر لایه ها (sub-layer) مختلف تقسیم بندی شد. در این بخش می خواهیم در مورد هر یک از زیر لایه های MAC و PHY اترنت صحبت کنیم.

لایه های MAC و PHY در استاندارد IEEE 802.3 و سرعت 100 Mb/s در شکل زیر نشان داده شده است.



شکل ۴-۴۲

واسط فیزیکی برای اتصال به رسانه ی انتقال، MDI (Media Dependent Interface) نامیده می شود و با توجه به نوع رسانای مورد استفاده (کابل CAT ، فیبر نوری و...) تغییر می کند.

واسط بین لایه MAC و PHY را MII (Media Independent Interface) می نامند و دارای یک مسیر فرستنده و یک مسیر گیرنده و یک مسیر برای مدیریت است که برای خواندن و نوشتن در رجیسترهای PHY استفاده می شود. عرض مسیر فرستنده و گیرنده یکسان است و با توجه به سرعتی که لایه های MAC و PHY پیاده سازی می کنند متفاوت است مانند :

10 Mb/s : عرض ۲ بیت در فرکانس 2.5MHz

100 Mb/s : عرض ۴ بیت در فرکانس 25MHz

همچنین از خطوط Reduced MII و Serial MII نیز استفاده می شوند که به ترتیب دارای عرض ۲ بیت و ۱ بیت هستند.

Reconciliation Layer (لایه ی مصالحه) : نشان دهنده ی وضعیت های بخش فیزیکی (مانند تصادم) به لایه ی MAC می باشد.

Media Independent Interface (واسط مستقل از رسانه) : فراهم کننده ی مسیر فرستنده و گیرنده بین لایه های MAC و PHY می باشد.

Physical Coding Sublayer (زیر لایه ی کد کردن فیزیکی) : کد کردن، مالتی پلکس کردن و همزمان سازی جریان سمبل های خروجی (روش کد کردن 4B/5B و...) بر عهده این بخش می باشد.

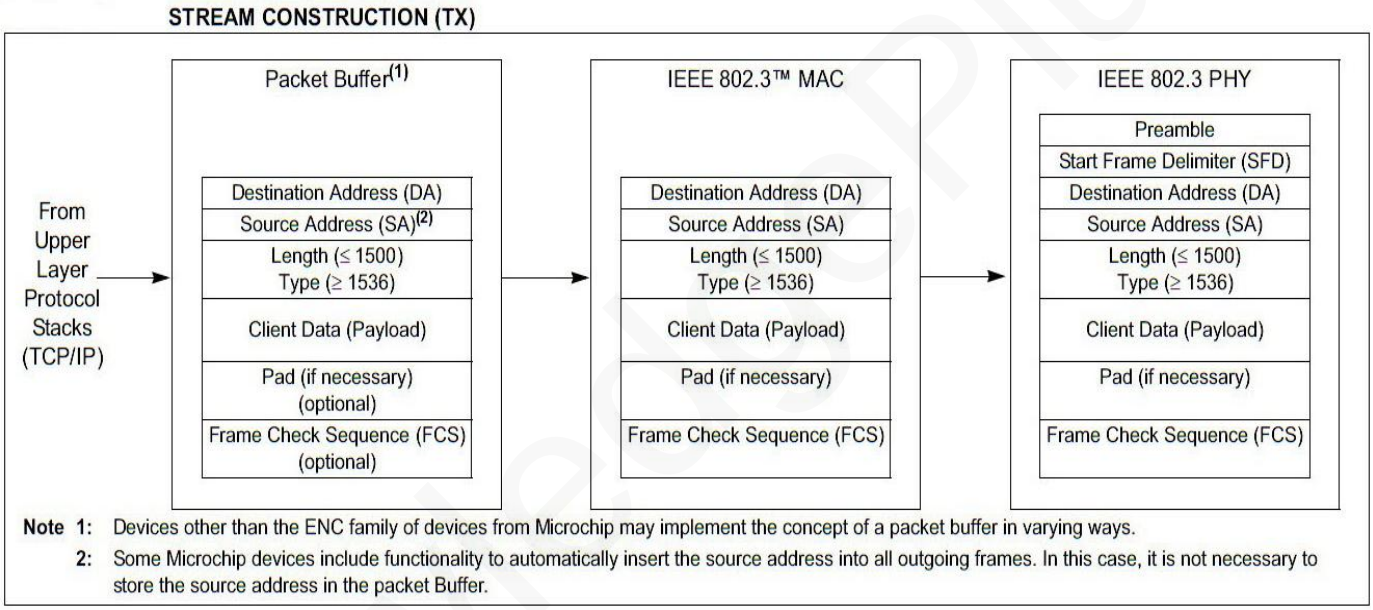
Physical Medium Attachment (الحاق رسانه ی فیزیکی) : انجام اعمالی مانند تقسیم بندی (serialization) ، deserialization ، بازیابی کلاک و ... بر روی جریان سمبل های ورودی و خروجی بر عهده این بخش می باشد.

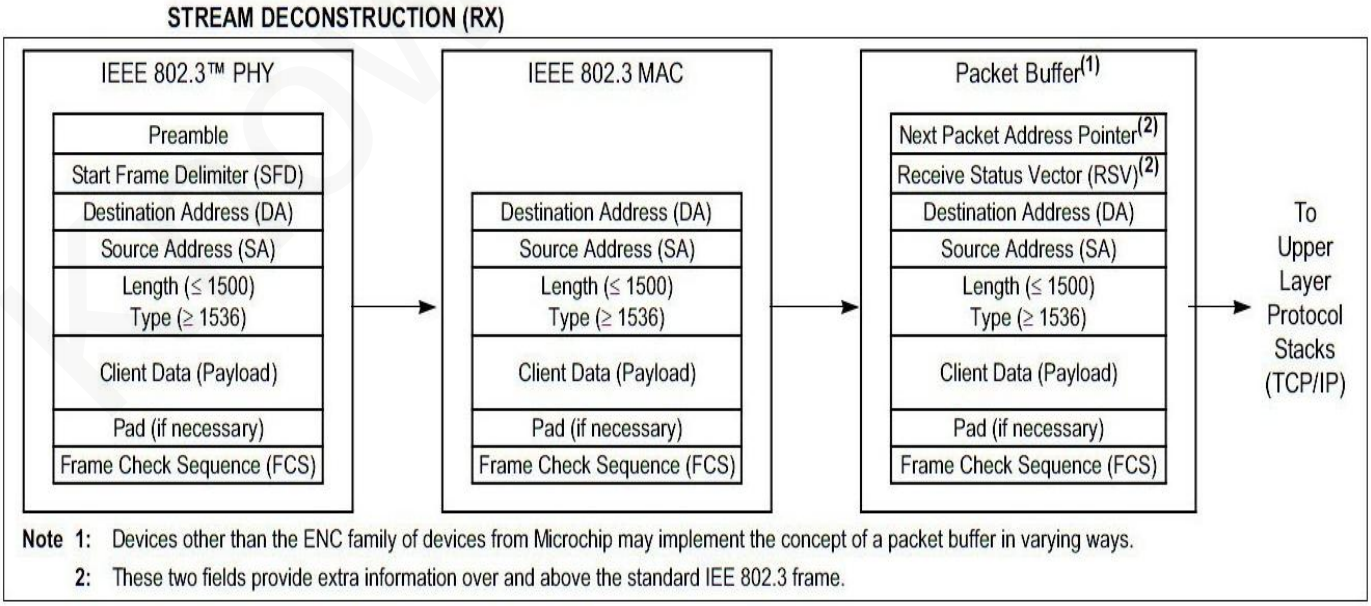
Auto-Negotiation : تنظیم مشخصات ارتباط بر عهده این بخش می باشد.

Medium Dependent Interface/Physical Media Dependent (MDI/PMD) : شامل تراشه transceiver برای پیاده سازی ارتباط در شبکه و کانکتور RJ-45 می باشد.

Medium : مانند کابل های UTP ، فیبر نوری و...

شکل اول فرایند تشکیل بسته های داده برای ارسال آنها در شبکه اترنت (Stream Construction) و شکل دوم فرایند دریافت بسته ها از شبکه و ارسال بسته ها به پروتکل های لایه های بالاتر برای پردازش (Stream Deconstruction) را نشان می دهند.





شکل ۴-۴۳

۱۸-٤. سیگنال های اترنت 10Mbit/s

با توجه به تفاوت های جریان اطلاعات در سرعت های 10 Mb/s و 100 Mb/s ، در این بخش به توضیح محتویات و سیگنال ها در این دو سرعت و همچنین نحوه ی انتقال فریم ها توسط رسانه ی فیزیکی می پردازیم.

مرحله اول برای انتقال داده ها، کد کردن آنها توسط کد Manchester است. در کد منچستر مورد استفاده در اترنت، گذر سیگنال از سطح منطقی High به سطح منطقی Low توسط عدد صفر کد شده است و برعکس آن یعنی گذر سیگنال از Low به High توسط عدد یک کد می شود.

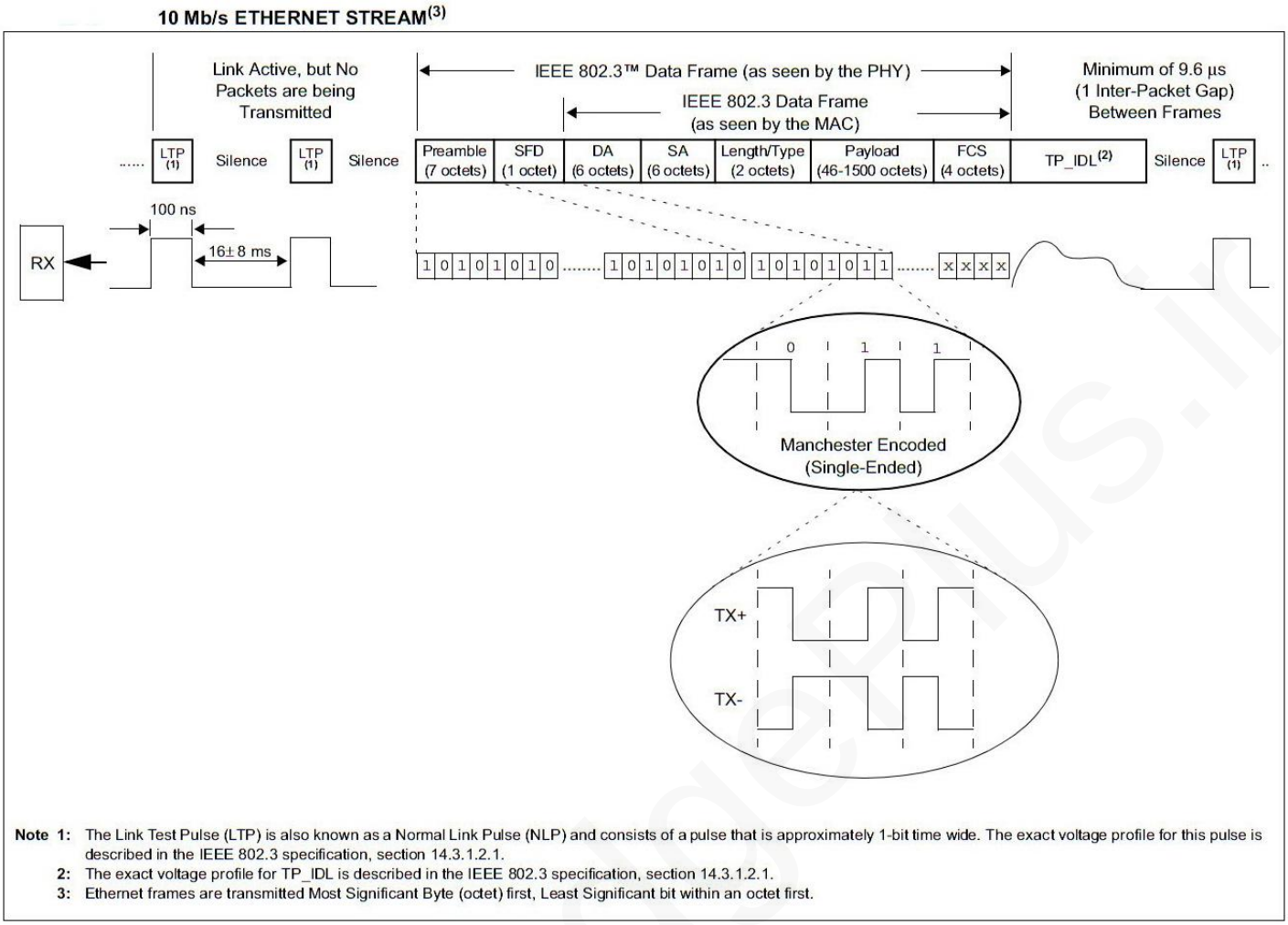
علت استفاده از کد منچستر، اطمینان بالای این کد و همچنین قابلیت بازیابی سیگنال از روی جریان داده ها می باشد که البته این مسئله به پهنای باند دو برابر نیاز دارد.

از آنجایی که 10Base-T از سیگنال های تفاضلی استفاده می کند، کد منچستر بر روی سیگنال های تفاضلی مطابق شکل صفحه بعد پیاده شده است.

مرحله دوم در انتقال داده ها، شکل دهی به شکل موج خروجی به منظور تطابق با مشخصات استاندارد IEEE 802.3 است که تعیین کننده ی تاخیر انتشار سیگنال به منظور تطابق با طول کابل شبکه و همچنین به حداقل رساندن EMI می باشد.

در نهایت سیگنال توسط درایور ولتاژ یا درایور جریان (این که کدام استفاده بشود به نوع پیاده سازی لایه PHY دارد) در طول رسانه ی فیزیکی شبکه انتقال می یابد. محدوده ی ولتاژ تفاضلی در گیرنده 350mv تا 3.1v می باشد.

شکل ۴-۴۴ جریان اطلاعاتی فریم های اترنت را در استاندارد 10Mbit/s نشان می دهد.



شکل ۴-۴۴

به دلیل این که کابل های UTP دارای رفتاری مشابه فیلترهای پایین گذر می باشند، از روش کد کردنی که در 10Base-T استفاده شد نمیتوان در سرعت های بالاتر مانند 100 Mb/s استفاده نمود.

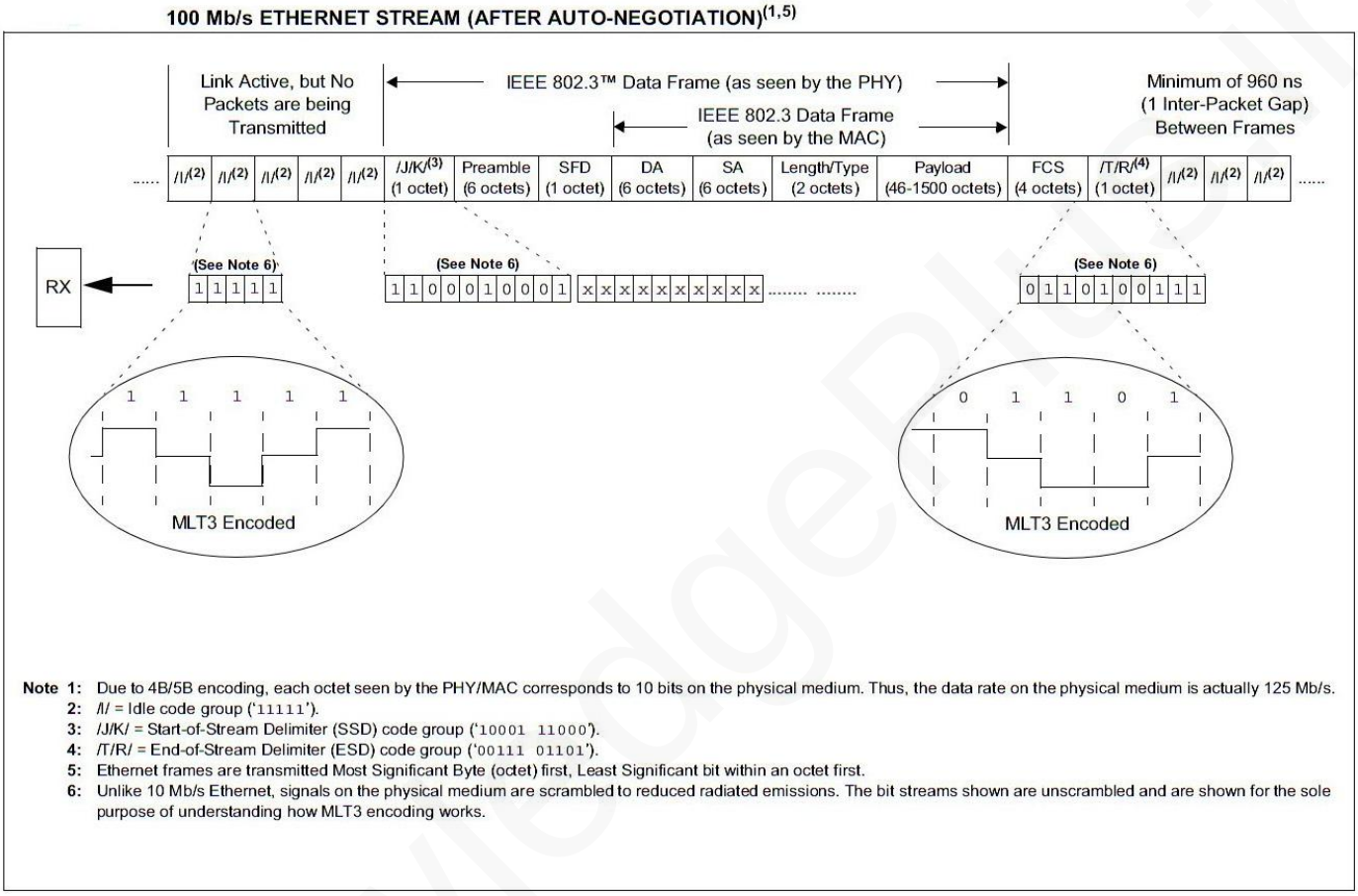
در پیاده سازی 100Base-TX از روش کدگذاری MLT3 (مخفف Multi-Level Transition 3 یا گذارچندگانه ۳) استفاده می شود که در شکل صفحه بعد نشان داده شده است. هر یک از سطوح منطقی صفر و یک توسط یکی از سه نوع گذار (transition) سیگنال کد می شود. گذار سیگنال همیشه در ولتاژهای نزدیک صورت می گیرد (... 1-,0,+1,0,1-). سطح منطقی صفر برای عدم گذر سیگنال و سطح منطقی یک برای گذر سیگنال می باشد.

به عنوان مثال داده '11111' را مطابق شکل ۴-۴۵ در نظر بگیرید. از آنجایی که مقدار یک معادل گذار سیگنال است، وجود یک های ثابت باعث ایجاد گذارهای متوالی سیگنال در هر بیت مطابق شکل می شود. با گذار سیگنال به نزدیکترین ولتاژ، تعداد گذارهای سیگنال کاهش پیدا می کند.

به دلیل این که کد MLT-3 نیاز به ۴ گذار دارد (1- به 0 و به 1+ و به 0 و به 1-)، برای اتمام یک

سیکل کامل، حداکثر فرکانس پایه ۴ برابر کاهش پیدا میکند، یعنی از 125MHz به 31.25MHz کاهش پیدا می کند.

شکل زیر جریان اطلاعاتی فریم های اترنت را در سرعت 100Mbit/s نشان می دهد.



شکل ۴-۴۵

٤-١٩. کد 4B/5B

علاوه بر استفاده از کد فیزیکی MLT3 در 100Base-TX ، از یک کد منطقی دیگر به نام 4B/5B یا به اصطلاح "Block Coding" نیز استفاده می شود.

کد های مورد استفاده در 100Base-TX باید دو ویژگی داشته باشند؛ اول قابلیت بازیابی کلاک در جریان طولانی داده های صفر. همان طور که اشاره شد صفر به معنی عدم وجود گذار سیگنال است. هنگام عدم وجود کلاک، گره فرستنده و گره گیرنده از حالت همزمان (synchronous) به دلیل ایجاد عواملی

مانند jitter خارج می شوند و در نتیجه داده ها تخریب می شوند.

دوم باید علاوه بر انتقال داده، قابلیت انتقال کد های انتقال مانند کد شروع فریم، پایان فریم، خطا و... را داشته باشد.

راه حل این مشکلات در سرعت 100 Mb/s مطابق جدول زیر، کد کردن هر چهار بیت با پنج بیت است. در نتیجه سرعت انتقال در رسانای فیزیکی از 100 Mb/s به 125 Mb/s افزایش پیدا می کند.

4B/5B ENCODING

Code	Value	Definition
0	11110	Data 0
1	01001	Data 1
2	10100	Data 2
3	10101	Data 3
4	01010	Data 4
5	01011	Data 5
6	01110	Data 6
7	01111	Data 7
8	10010	Data 8
9	10011	Data 9
A	10110	Data A
B	10111	Data B
C	11010	Data C
D	11011	Data D
E	11100	Data E
F	11101	Data F
I	11111	Idle
J	11000	SSD (Part 1)
K	10001	SSD (Part 2)
T	01101	ESD (Part 1)
R	00111	ESD (Part 2)
H	00100	Transmit Error

جدول ۴-۲۴

اگر به کدهای ارائه شده در جدول دقت کنیم، به غیر از کد H که کد خطا می باشد، در همه ی کد ها حداقل دو تا یک به معنای دو گذار در سیگنال کد شده به روش MLT3 وجود دارد و در نتیجه مسئله ی بازیابی کلاک به این شکل حل می شود.

با 2^5 کد برای ۱۶ مقدار، ۱۶ مقدار اضافی برای کد کردن سیگنال های انتقالی وجود دارد که شامل موارد زیر می شود :

شرایط آزاد (Idle) که جایگزین NPL ها (Normal Link Pulse یا پالس اتصال دهنده ی معمولی)

در 10Base-T می شوند.

SSD که جایگزین اولین بایت مربوط به Preamble در 10Base-T می شود.

ESD که جایگزین شکل موج های TP_IDL که در 10Base-T استفاده می شد می شود.

خطای ارسال که معادل مشابهی در 10Base-T ندارد.

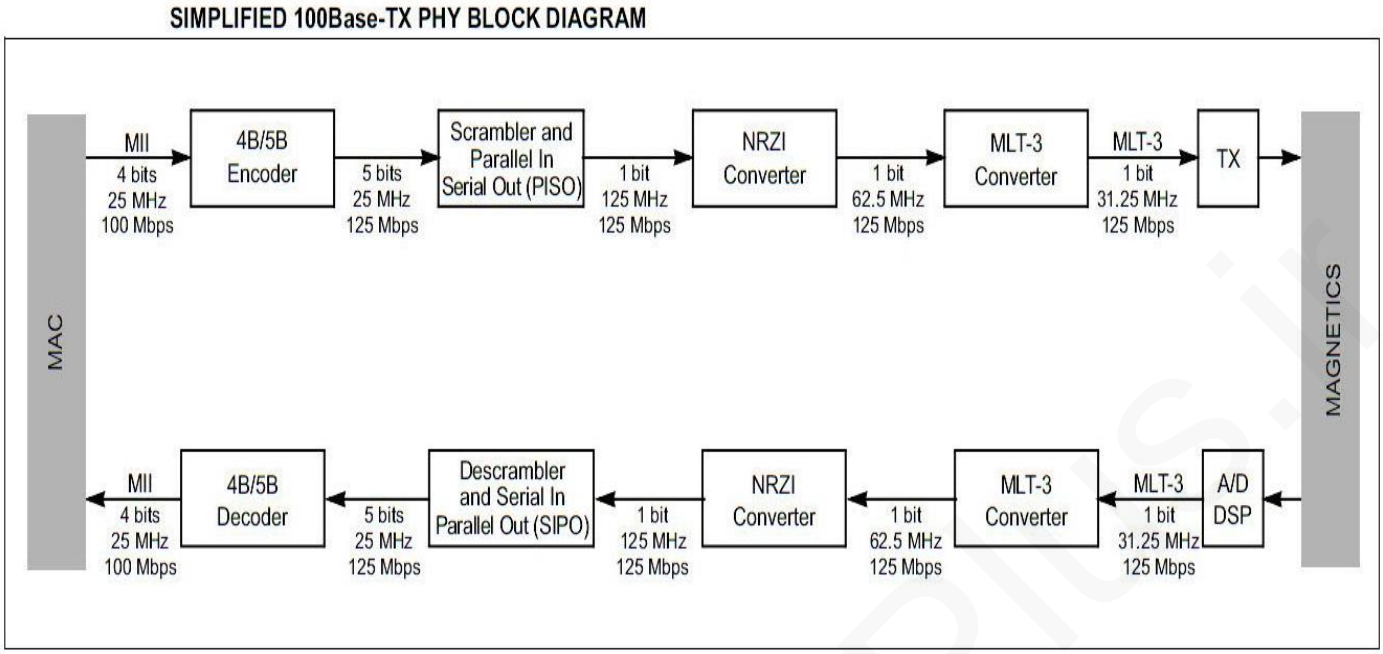
٤-٢٠. بلوک دیاگرام کلی کدینگ و دیکدینگ

تا اینجا در مورد کد کردن و دیکد کردن در 10Base-T که از کد منچستر استفاده می کرد و 100Base-TX که از کدهای NRZI ، MLT3 و 4B/5B استفاده می کرد توضیح دادیم.

در کد منچستر که هر گذار سیگنال تبدیل به یک بیت می شود، در سرعت 10 Mb/s نیاز به پهنای باند 10MHz در رسانای فیزیکی وجود دارد. همچنین از طریق استفاده از سیگنال های تفاضلی، مصونیت در برابر نویز (Noise Immunity) نیز وجود دارد.

سوالی که وجود دارد این است که چگونه همه ی این روش های کدکردن در 100Base-TX با هم ترکیب می شوند تا سیگنال انتقالی را تشکیل بدهند؟

شکل ۴-۴۶ بلوک دیاگرام ساده ای از لایه ی فیزیکی 100Base-TX و پهنای باند مورد نیاز در هر مرحله را نشان می دهد. در این بلوک دیاگرام می توان مشاهده کرد که اگرچه به واسطه ی استفاده از کد 4B/5B نرخ داده به 125 Mb/s افزایش پیدا کرده است، اما پهنای باند مورد نیاز رسانه ی فیزیکی از این مقدار بسیار کمتر است.



شکل ۴-۴۶

٤-۲۱. مذاکره خودکار (Auto-Negotiation)

مذاکره خودکار یا Auto-Negotiation ، فرایندی است که طی آن دو گره برای ارتباط برقرار کردن با یکدیگر قابلیت ها و توانایی های یکدیگر را (مانند سرعت، Full-Duplex یا Half-Duplex بودن ارتباط، پشتیبانی از فریم های pause و...) را برای ایجاد تطابق کامل با یکریگر مورد ارزیابی و بررسی قرار می دهند.

وجود فرایند مذاکره خودکار برای استانداردهای سرعت پایین مانند 10Base-T و 100Base-T دلخواه است اما برای gigabit Ethernet لازم و ضروری است.

فرایند مذاکره خودکار طبق شکل صفحه بعد، از طریق FLP ها (Fast Link Pulses یا پالس های اتصال دهنده ی سریع) صورت می پذیرد. FLP مشابه NLP (Normal Link Pulses) می باشد با این تفاوت که تعداد ۱۷ تا ۳۳ پالس متوالی (به نام link code word یا کلمات کد اتصال دهنده) را در بین NLP ها می فرستند.

تعداد بیت های کلمات کد اتصال دهنده که توسط هر گره فرستاده می شود توسط استاندارد IEEE 802.3 مشخص نشده است اما هر گره که از مذاکره خودکار پشتیبانی می کند، باید حداقل قادر به ارسال

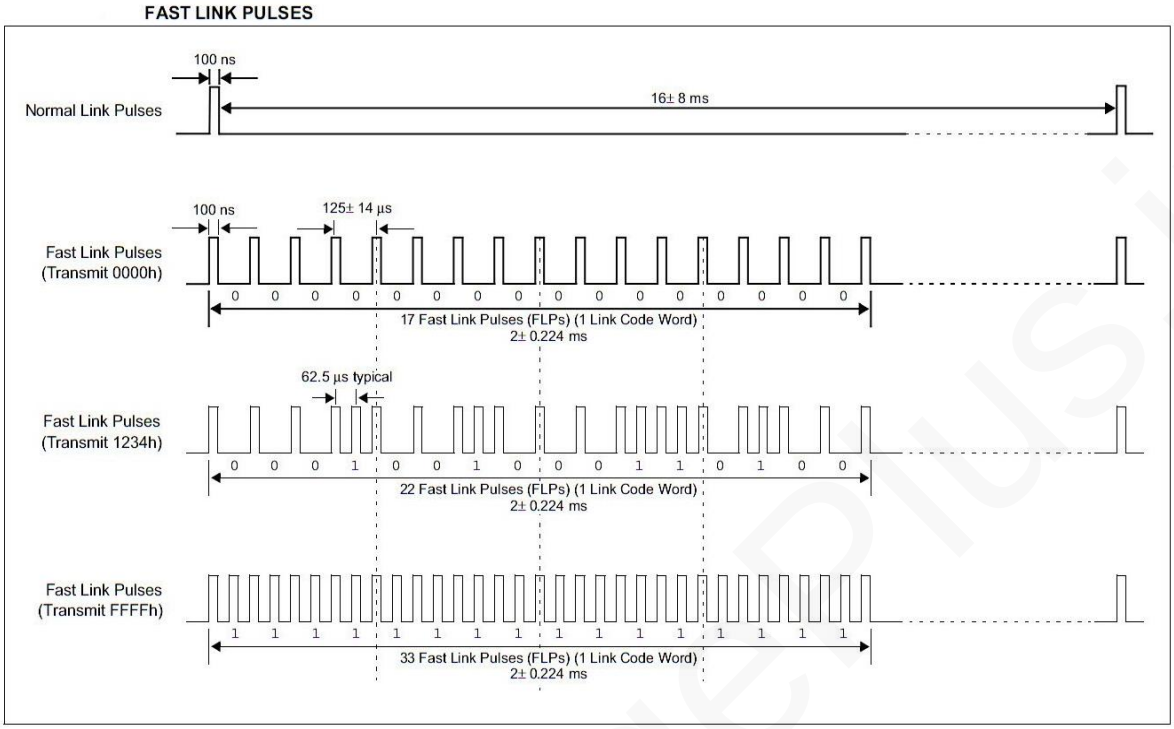
صفحه پایه (base page) مذاکره خودکار باشد.

صفحه پایه یک کد ۱۶ بیتی است که مشخصات و قابلیت های گره را مطابق جدول زیر مشخص می کند.

بیت های 0-4	به نام بخش انتخاب کننده (selector) شناخته می شود (S0-S4) و مشخص کننده نوع تکنولوژی LAN مورد استفاده است. برای اترنت طبق استاندارد IEEE 802.3 برابر عدد "10000" می باشد.
بیت های 5-12	به نام بخش توانایی تکنولوژی (technology ability) شناخته می شود (A5-A12) و توانایی های گره را مشخص می کند. بیت های 5-9 : مشخص کننده ی نوع اتصال و به ترتیب اولویت برابر است با : 100Base-TX full-duplex (bit 3 set) 100Base-T4 (bit 4 set) 100Base-TX (bit 2 set) 10Base-T full-duplex (bit 1 set) 10Base-T (bit 0 set) بیت 10 : یک بودن به معنای وجود قابلیت pause است بیت 11 : یک بودن به معنای پشتیبانی از pause های متقارن در اتصال دهنده های Full-Duplex است بیت 12 : پشتیبانی از صفحات توسعه یافته بعدی (Extended next pages)، فقط در gigabit Ethernet کاربرد دارد
بیت 13	بیت نشان دهنده ی RF (Remote Fault یا خطای از راه دور)
بیت 14	بیت ACK (Acknowledge یا تصدیق)، بیت ضروری که برای گیرنده ارسال می شود و نشان دهنده ی پیام FLP است. برای ارسال بیت تصدیق برای فرستنده باید سه پیام FLP یکسان را گیرنده دریافت کند تا بیت تصدیق را برای فرستنده بفرستد
بیت 15	بیت NP (Next Page یا صفحه بعدی) که نشان دهنده ی ارسال صفحه بعدی بعد از صفحه پایه است. صفحه بعدی در مذاکره خوکار استفاده می شود و برای ارسال اطلاعات بیشتر بین دو گره استفاده می شود

جدول ۴-۲۵

شکل زیر فرایند مذاکره خودکار را از طریق پالس های FLP نشان می دهد.



شکل ۴-۴۷

٤-۲۲. اترنت تعبیه شده (Embeded Ethernet)

مفهوم اترنت تعبیه شده یا embedded ethernet که در قالب سیستم های نهفته یا embedded systems تعریف می شود، به معنای استفاده از اترنت در سیستم های embedded که غالباً با استفاده از میکروکنترلر های مختلف طراحی می شوند می باشد. هدف پروژه انجام شده در این مقاله نیز مستند سازی مراحل و تکنیک های پیاده سازی اترنت تعبیه شده در پروژه های مختلف که نیاز به اتصال به شبکه های مبتنی بر تکنولوژی اترنت دارند می باشد.

شکل ۴-۴۸

معمولا سرعت های مورد استفاده در اترنت تعبیه شده 10Mbit/s یا 100Mbit/s می باشد.

برخی از کاربردهای پیاده سازی اترنت تعبیه شده عبارت اند از :

- وب سرور
- سرور های GSM
- اتوماسیون صنعتی
- گلخانه هوشمند
- اتوماسیون منازل
- VOIP
- Set-Top Box
- PVR
- DVR
- POS
- مودم های DSL
- IP and Video Phones
- اکسس پوینت ها
- تلویزیون های دیجیتال
- کنسول های بازی
- و ...

وب سرور یکی از پیاده سازی های جذاب و پر کاربردی است که توسط پیاده سازی اترنت می توان به مرحله اجرا گذاشت.

شکل ۴-۴۹

در این پروژه حافظه flash میکروکنترلر همانند یک FTP عمل کرده و پس از اتصال برد اترنت به شبکه، هر یک از کامپیوترهای موجود در شبکه با نوشتن آدرس IP دستگاه در مرورگر اینترنتی خود، به صفحه وب طراحی شده توسط برد هدایت می شوند.

در این مرحله می توان پایه های خروجی دستگاه را کنترل و یا اینکه اطلاعاتی را توسط دستگاه دریافت و در این صفحه به نمایش گذاشت. در این حالت نیازی به استفاده از یک دستگاه سرور واسط نبوده و شما با اجرای یک IP می توانید با دستگاه ارتباط برقرار کرده و بر اساس اطلاعات دریافتی دستورات لازم را ارسال نمایید.

به علت اهمیت و نیاز این پروژه در بسیاری از موارد، سورس این پروژه به همراه مقاله برای کامپایلرهای CodevisionAVR و WinAVR ارائه شده است.

سرور های GSM امروزه یکی از پرکاربردترین و مهم ترین دستگاه های ارتباطی عضر حاضر می باشند. شما توسط سرور های GSM می توانید از طریق رایانه به شبکه موبایل متصل شده و از این طریق امکانات این شبکه از قبیل GPRS ، SMS و انتقال صدا استفاده نمایید.

بیشترین کاربرد در این زمینه مربوط به سرور های SMS بوده که هم اکنون توسط شرکت های بزرگ به صورت شبکه داخلی یا سرویس هایی به صورت اینترنتی ارائه می گردند.

در این روش شما با اتصال یک دستگاه GSM و از طریق شبکه داخلی یا اترنت، تعداد زیادی کاربرد می توانند به صورت همزمان SMS ارسال نمایند.

در شرکت های تبلیغاتی که دارای چندین کارمند می باشند، با این روش سرعت ارسال SMS به مشتریان افزایش یافته و همچنین هزینه های راه اندازی و نگه داری به شدت کاهش می یابد.

پیشرفت تکنولوژی و افزایش جمعیت باعث شده تا هیچ یک از شرایط قدیمی برای تولید قابل اجرا نبوده و روش های جدید جایگزین روش های سنتی و قدیمی شوند.

یکی از مهم ترین تولیدات، کشاورزی و زراعت است. در روش های نوین، گلخانه ها جایگزین مزارع بزرگ شده و در مساحت بسیار کوچک، در تمامی فصول با کنترل بهینه و پیشرفته، بازدهی چندین برابر مزارع بزرگ به دست می آید.

گلخانه ها نیز در این چند سال اخیر دستخوش تغییرات بسیار زیادی شده و تکنولوژی های پیشرفته نیز به این صنعت راه یافته و نام گلخانه به گلخانه های هوشمند تغییر یافته است.

یکی از مهم ترین نکات گلخانه های هوشمند، نظارت ۲۴ ساعته از دورترین نقاط ممکن می باشد. یکی از روش های مناسب برای این منظور استفاده از شبکه اترنت می باشد.

شما می توانید با طراحی یک گره اترنت و قرار دادن آن در یک گلخانه و با استفاده از انواع سنسورهای دما و رطوبت و همچنین کنترل دستگاه های برودتی و گرمایشی و کنترل آنها از طریق شبکه اینترنت و کیلومترها دورتر، یک گلخانه هوشمند با کارایی بسیار بالا طراحی و اجرا نمایید.

شکل ۴-۵۰

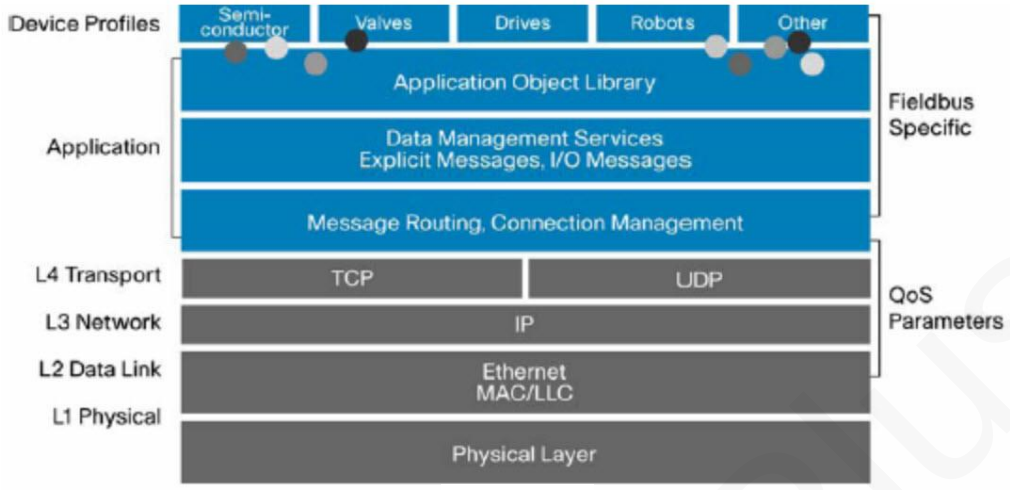
٤-۲۳. اترنت صنعتی (Industrial Ethernet)

اگرچه همان طور که در بخش قبل اشاره شد هدف از این مقاله پیاده سازی اترنت نهفته در کاربردهای مختلف می باشد، بد نیست اشاره ای هم به یکی از کاربردهای بسیار مهم و رو به گسترش اترنت تحت عنوان اترنت صنعتی یا Industrial Ethernet بیندازیم.

امروزه با توجه به ویژگی های اترنت، سازمان ها در حال انتقال از پروتکل صنعتی Fieldbus به سمت Industrial Ethernet هستند.

اترنت صنعتی به مجموعه استانداردهای اترنت گفته می شود که برای انتقال داده در شبکه های کنترل صنعتی مورد استفاده قرار می گیرند.

مطابق شکل زیر، مشخصات پروتکل Fieldbus در لایه های ۵ ، ۶ و ۷ مدل OSI قرار می گیرد.



شکل ۴-۵۱

اترنت صنعتی از همان استانداردهای اترنت استاندارد استفاده می کند که در فصول قبلی توضیح داده شد اما باید دارای ویژگی های دیگری باشد تا برای استفاده در محیط های صنعتی مناسب باشد، مثلا تحمل شرایط محیطی سخت، تعداد گره های شبکه بیشتر، استفاده از کابل های متنوع تر، عملکرد real-time بهتر، قابلیت پیشبینی و...

یکی از مهم ترین تفاوت های اترنت صنعتی و اترنت معمولی نوع سخت افزار به کار رفته می باشد.

شبکه اترنت صنعتی برای کار در شرایط سخت، درجه حرارت وسیع، لرزش و شوک زیاد، استفاده از قطعات با کیفیت بالا، سیستم های تهویه مناسب، تغذیه متفاوت با اترنت معمولی (معمولا ۲۴ ولت)، قابلیت تحمل خطا (fault-tolerant) طراحی شده است.

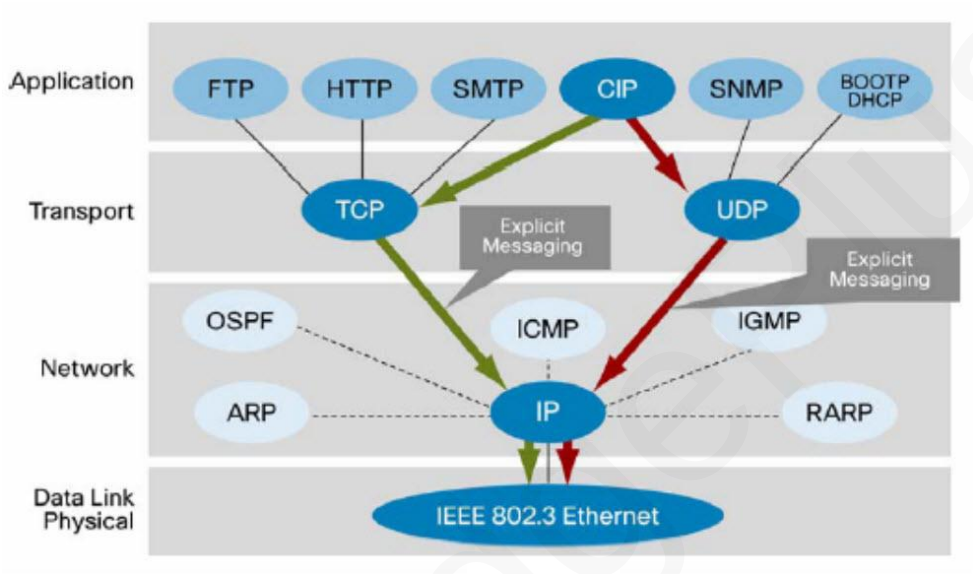
یکی دیگر از تفاوت های شبکه های اترنت صنعتی با اترنت استاندارد در این است که در شبکه اترنت استاندارد ۸۰ درصد ترافیک داده در شبکه محلی (local) است و به صورت multicast (یک فرستنده، چندین گیرنده) می باشد در حالی که در شبکه های صنعتی برعکس این قضیه است و ۸۰ درصد ترافیک شبکه به صورت unicast (یک فرستنده، یک گیرنده) به تجهیزات خارجی ارسال می شود. (مانند دیتاسنترها و یا شبکه اینترنت)

همچنین حفاظت ها و پیشبینی خطاهای احتمالی در شبکه های اترنت صنعتی با توجه به نوع کاربرد مورد استفاده اهمیت زیادی دارد.

اترنت صنعتی ضمن دارا بودن ویژگی های امنیتی و حفاظتی در Fieldbus ، از پهنای باند بیشتری بهره می برد.

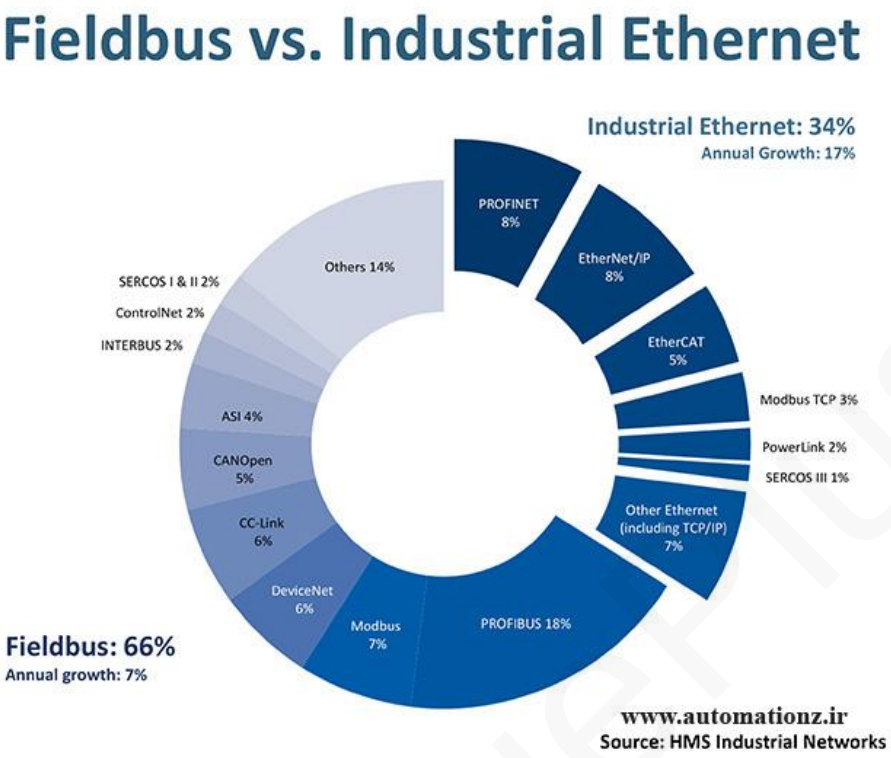
یکی از مسائل مهم در اترنت صنعتی، QOS (Quality Of Service) می باشد. در QOS یکی از مسائل مطرح اولیت بندی ترافیک شبکه به منظور عملکرد real-time می باشد. در شبکه ترافیک داده های مختلفی وجود دارد مانند داده هایی که نسبت به زمانبندی اهمیت زیادی دارند (time-critical)، داده های صوتی یا تصویری و ...

اولویت بندی این داده ها به منظور ایجاد قابلیت اطمینان در شبکه و بهبود QOS اهمیت زیادی دارد.



شکل ۴-۵۲

شرکت سوئدیHMS که در حوزه شبکه های صنعتی فعالیت می کند، چند سالی است که به واسطه تولید محصولات حرفه ای و کارآمد به یکی از شرکت های خوشنام و پیشرو در جهان تبدیل شده است. محصولات این شرکت تحت سه برند Anybus ، IXXAT و Netbiter تولید و ارائه می شوند که برخی از محصولات برند Anybus این شرکت در پروژه های مختلفی در ایران نیز استفاده شده است. این شرکت سوئدی به تازگی یک بررسی آماری جالب توجه در مورد میزان استفاده از شبکه های صنعتی مختلف در جهان انجام داده است و نتایج آن را مطابق شکل ۴-۵۳ بر روی وبسایت خود منتشر کرده است.



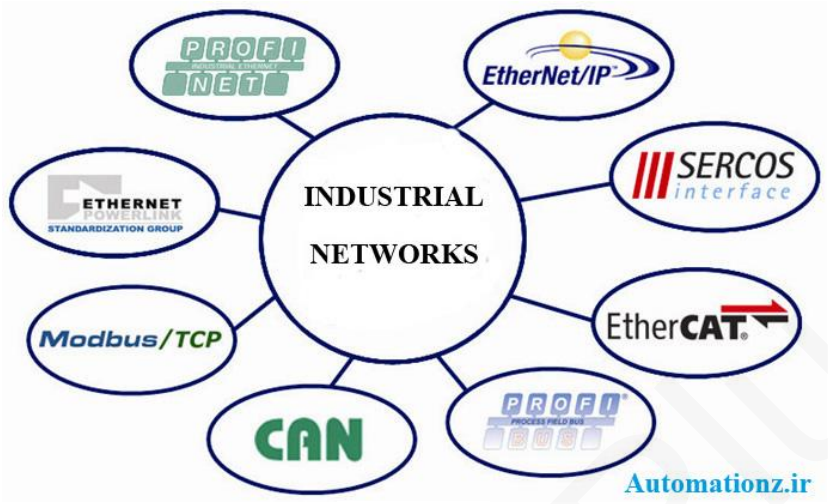
شکل ۴-۵۳

طبق اطلاعات منتشر شده، شرکت HMS برای بازار های اتوماسیون صنعتی در سال 2015 پیش بینی می کند که PROFIBUS همچنان رایج ترین شبکه صنعتی در جهان باقی بماند اما شبکه های دیگر در حال کم کردن فاصله خود با این شبکه هستند. شبکه های PROFINET و EtherNet/IP هر کدامشان با داشتن ۸ درصد از سهم بازار در حال رقابت با یکدیگر برای کسب جایگاه نخست در شبکه های صنعتی مبتنی بر اترنت صنعتی هستند.

در حال حاضر میزان استفاده از هر دو نوع شبکه های اترنت صنعتی و فیلد باس رو به افزایش است اما سرعت رشد شبکه های صنعتی مبتنی بر اترنت بالاتر می باشد. با توجه به گره هایی که اخیرا در کارخانجات در سراسر دنیا نصب شده است، شبکه های فیلد باس (روش انتقال سریال) با دارا بودن ۶۶ درصد از سهم بازار هنوز دارای اکثریت هستند و شرکت HMS پیش بینی کرده است که شبکه های فیلد باس می توانند برای سال های آینده نیز شاهد رشد هفت درصدی باشند.

فاکتور های اصلی برای رشد فیلد باس ها عبارت اند از سادگی، قابلیت اطمینان و اینکه هنوز استفاده از آنها به صورت سنتی در بسیاری از موارد پیشنهاد می شود. شبکه فیلد باس غالب در بازار شبکه پروفیباس است که دارای سهم ۱۸ درصدی از کل بازار شبکه های صنعتی است و به دنبال آن شبکه های modbus

۷ درصد، DeviceNet ۶ درصد و CC-Link ۶ درصد در جایگاه های بعدی قرار دارند. شبکه های CAN Open، ASI و Interbus نیز به ترتیب دارای سهم های پنج، چهار و دو درصدی هستند.



شکل ۴-۵۴

شبکه های اترنت صنعتی در حال حاضر صاحب ۳۴٪ از کل بازار هستند ولی با سرعت بالاتری نسبت به شبکه های فیلد باس در حال رشد هستند. با توجه به نرخ هفده درصدی سالیانه، پیش بینی میشود که به زودی فاصله خود را با شبکه های فیلد باس کاهش دهد ولی گرفتن اکثریت سهم بازار هنوز کمی زمان بر خواهد بود. فاکتور های اصلی برای رشد شبکه های اترنت صنعتی عبارت است از عملکرد بالاتر و قابلیت یکپارچه سازی آنها با شبکه های اداری. شبکه های PROFINET و EtherNet/IP با دارا بودن ۸ درصد از کل سهم بازار به طور مشترک شبکه غالب در این گروه محسوب می شوند و به دنبال آن ها EtherCAT ۵ درصد و Modbus-TCP با ۳ درصد قرار دارند.

در آمریکا شبکه EtherNet/IP با فاصله کمی نسبت به DeviceNet بیشترین سهم بازار را در اختیار دارد و در ادامه شبکه های PROFIBUS و EtherCAT در جایگاه های بعدی قرار دارند. علاوه بر این شبکه Profinet در آمریکا نیز در حال افزایش سهم خود از بازار است.

در آسیا طبق بررسی شرکت HMS نمی توان هیچ شبکه ای را به عنون شبکه برتر نام برد اما PROFIBUS ، DeviceNet و Modbus به شکل گسترده ای در قاره آسیا استفاده می شوند، ضمن اینکه CC-Link در کشور ژاپن شبکه غالب است و EtherCAT مثل سایر نقاط جهان در حال افزیش سهم خود است.

فصل پنجم

نرم افزار

۵-۱. مقدمه

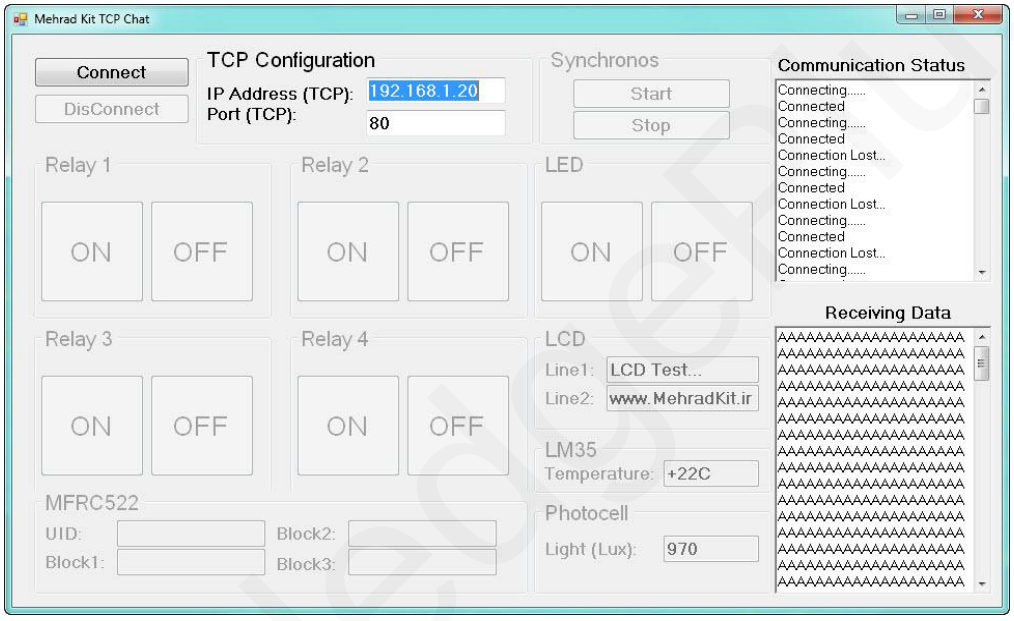
برخلاف سخت افزار لازم برای پیاده سازی اترنت که معمولا شکل ثابتی دارد، بخش نرم افزاری و کدنویسی اترنت با توجه کاربرد مورد نظر متفاوت می باشد. در هر کاربرد با توجه به نیاز ها و شرایط لازم، باید روش مناسب برای طراحی نرم افزار و رابط کاربری برای ارتباط با شبکه در نظر گرفته شود.

همان طور که در فصول قبلی اشاره شد، در پروتکل اترنت با استفاده از بخش FCS و روش های مختلفی مانند CRC یا محاسبات checksum می توان از صحت بسته های داده دریافت شده اطمینان حاصل نمود. محاسبات این بخش معمولا توسط تراشه (واحد) کنترل کننده اترنت به صورت خودکار صورت می پذیرد. علاوه بر این محاسبات، در پیاده سازی پروتکل های لایه های بالاتر، هر پروتکل معمولا شامل روش هایی برای اطمینان از دریافت صحیح داده می باشد. محاسبات این بخش معمولا باید توسط برنامه نویس توسط کدهای مناسب پیاده سازی شود. بدیهی است که این محاسبات باعث کاهش سرعت می شود و باید متناسب با کاربرد مورد نظر و حساسیت لازم برای دریافت بسته های صحیح، در مورد پیاده سازی این بخش ها تصمیم گرفت.

از دیگر مسائلی که باید در پیاده سازی پروتکل های ارتباطی مانند اترنت به آن توجه شود، بحث زمان بندی یا timing می باشد. در این بخش باید با توجه به سرعت شبکه، بخش نرم افزار شبکه به طور مناسبی بتواند بسته های شبکه را دریافت و درخواست های ارسال شده را پاسخ بدهد، به عنوان مثال در

صورتی که بسته هایی نیاز به پاسخگویی سریع تر دارند، در وقفه های نرم افزاری باید پاسخگویی به این بسته ها در اولویت قرار گیرد. در صورت عدم در نظر گرفتن مسائل زمان بندی، ممکن است سرعت عملکرد شبکه افت کند و یا حتی کارکرد شبکه با اختلال مواجه شود.

در بخش پیاده سازی رابط کاربری، با توجه به کاربرد مورد نظر می توان از روش های متفاوتی استفاده نمود. به عنوان مثال ممکن است لازم باشد با استفاده از زبان هایی مانند Java ، C# یا زبان های سطح بالای مشابه یک رابط کاربری برای ارتباط با شبکه طراحی شود. شکل زیر یک نمونه از این رابط های گرافیکی را نشان می دهد.



شکل ۵-۱

در پروژه ای که در این مقاله قصد پیاده سازی آن را داریم، از پروتکل های مختلف در لایه های مدل TCP/IP برای پیاده سازی بستر ارتباط و از پروتکل HTTP و زبان برنامه نویسی HTML برای طراحی یک صفحه وب استفاده می شود.

در فصل چهارم در مورد پروتکل ارتباطی اترنت و ساختار بسته های اترنت و جزئیات دیگر در مورد این پروتکل توضیح دادیم.

با توجه به این که در عمل در کاربردهای مختلف علاوه بر پیاده سازی پروتکل ایترنت، نیاز به پیاده سازی پروتکل های دیگر نیز که در فصل قبلی نیز اشاره ای به آنها شد وجود دارد و در این مقاله نیز در پیاده سازی عملی پروژه از این پروتکل ها استفاده شده است، در این فصل توضیحاتی در مورد این پروتکل ها از جنبه عملی و پیاده سازی آنها ارائه خواهد شد.

به طور کلی هر پروتکل شامل یک بخش سرآیند (Header) شامل اطلاعات و تنظیمات مخصوص آن

پروتکل و بخش داده (Payload) که توسط لایه های بالاتر تولید می شود می باشد. در واقع همان طور که در فصول قبلی در بخش مدل های شبکه توضیح داده شد، هر لایه (که توسط یک پروتکل پیاده سازی می شود)، بسته داده را از لایه بالاتر دریافت کرده و به آن سرآیند خود را اضافه می کند. به عنوان مثال پروتکل TCP بسته تشکیل شده توسط پروتکل HTTP را (در لایه بالاتر) دریافت کرده و به آن سرآیند خود را اضافه کرده و به پروتکل IP در لایه پایین تر تحویل می دهد، پروتکل IP سرآیند خود را به بسته اضافه کرده و به پروتکل Ethernet در لایه پایین تر تحویل می دهد و پروتکل Ethernet سرآیند خود را به بسته اضافه کرده و آن را در شبکه ارسال می کند.

همان طور که در فصل قبل توضیح داده شد، به فرایند اضافه کردن سرآیند به بسته ها، encapsulation یا بسته بندی گفته می شود.

در بخش بعدی به توضیح مختصر عملکرد و کاربرد هر پروتکل و بخش های مختلف سرآیند هر پروتکل می پردازیم. در فصل انتهایی این مقاله با استفاده از نوشتن کدهای مناسب و مطابق توضیحات ارائه شده در این فصل، پروتکل های مختلف را برای ایجاد یک وب سرور پیاده سازی می کنیم. همچنین می توانید خودتان هنگام خواندن توضیحات ارائه شده برای هر پروتکل در این فصل، به کتابخانه های ارائه شده برای هر پروتکل مراجعه فرمایید و نحوه ی پیاده سازی آن را در هنگام مطالعه این بخش دنبال کنید.

۵-۲. پروتکل HTTP

HTTP (مخفف Hypertext Transfer Protocol یا پروتکل انتقال ابرمتن)، یکی از پروتکل های لایه ی کاربرد می باشد و یکی از اصلی ترین پروتکل های مورد استفاده در شبکه اینترنت می باشد. در این بخش به برخی از نکات مربوط به پیاده سازی این پروتکل اشاره می شود.

۵-۲-۱. جلسه HTTP (HTTP Session)

به سلسله ای از درخواست ها و پاسخ ها در شبکه بر مبنای پروتکل HTTP ، HTTP session گفته می شود.

client (سرویس گیرنده) یک درخواست توسط پروتکل TCP و با شماره پورت مشخص (معمولا ۸۰) برای طرف server (سرویس دهنده) ارسال می کند.

سمت سرور که دائما در حال بررسی پورت ها برای پاسخ دادن به درخواست های ارسال شده است، پس از دریافت هر درخواست، به آن پاسخ می دهد. این پاسخ شامل خط وضعیت (status line) در ابتدای پاسخ مانند "HTTP/1.1 200 OK" و در ادامه در بخش بدنه (body) اطلاعات و داده های متناسب با آن درخواست را ارسال می کند.

۵-۲-۲. روش های درخواست (Request methods)

برای ارسال درخواست از سمت سرور به سمت کلاینت، روش های مختلفی توسط نسخه های مختلف HTTP ارائه شده است. به عنوان مثال در نسخه اول HTTP (HTTP/1.0) سه روش GET ، POST و HEAD ارائه شده است.

نکته : در روش GET درخواست فاقد بدنه (body) است ولی پاسخ شامل بدنه می باشد.

۵-۲-۳. ساختار پیام ها (Message Format)

ساختار پیام درخواست :

- یک خط درخواست مثلا "GET /images/logo.png HTTP/1.1" که درخواست فایل /images/logo.png را از سرور می کند)
- خط عنوان درخواست (Request header fields) مثلا "Accept-Language: en" برای تنظیم زبان انگلیسی
- یک خط خالی
- اطلاعات و داده ها

ساختار پیام پاسخ :

- خط وضعیت که شامل کد وضعیت (status code) می شود.(مثلا "HTTP/1.1 200 OK" که به معنای دریافت موفق آمیز درخواست کلاینت است)
- خط عنوان پاسخ (Response header fields) مثلا Content-Type: text/html که برای تعیین نوع داده متنی با فرمت HTML استفاده می شود.
- یک خط خالی
- اطلاعات و داده ها

مثال :

Client request (درخواست سرویس گیرنده) :

```
GET /index.html HTTP/1.1
Host: www.example.com
```

Server response(پاسخ سرویس دهنده) :

```
HTTP/1.1 200 OK
Date: Mon, 23 May 2005 22:38:34 GMT
Content-Type: text/html; charset=UTF-8
Content-Encoding: UTF-8
Content-Length: 138
Last-Modified: Wed, 08 Jan 2003 23:11:55 GMT
Server: Apache/1.3.3.7 (Unix) (Red-Hat/Linux)
ETag: "3f80f-1b6-3e1cb03b"
Accept-Ranges: bytes
Connection: close
<html>
<head>
  <title>An Example Page</title>
</head>
<body>
  Hello World, this is a very simple HTML document.
</body>
</html>
```

ETag برای بررسی وجود یک نسخه cache از اطلاعات درخواست شده در سرور به کار می رود،

Content-Type مشخص کننده نوع پیام (متن، تصویر و...)، Content-Length مشخص کننده اندازه پیام بر حسب بایت، عبارت Connection: close به این معنا است که پس از ارسال جواب برای سرویس گیرنده، سرویس دهنده ارتباط TCP خود را خاتمه می دهد.

۵-۳. پروتکل TCP

پروتکل TCP (مخفف Transmission Control Protocol یا قرارداد کنترل انتقال)، پروتکلی اتصال گرا (connection-oriented) می باشد. علت این امر ایجاد یک ارتباط مجازی بین گره فرستنده و گیرنده بعد از ارسال اطلاعات است. پروتکل هایی از این نوع، امکانات بیشتری را به منظور کنترل خطاهای احتمالی در ارسال اطلاعات فراهم نموده ولی به دلیل افزایش بار عملیاتی سیستم، کارایی آنان کاهش خواهد یافت. از پروتکل TCP به عنوان یک پروتکل قابل اطمینان نیز یاد می‌شود، به این علت که برای آگاهی از صحت اطلاعات ارسال شده، اطلاعات دیگری نیز به گیرنده فرستاده می‌شود. در صورتی که بسته‌های داده به درستی به گیرنده نرسد، فرستنده مجدداً اقدام به ارسال بسته ها می‌نماید.

در مقابل، پروتکل UDP بر خلاف پروتکل TCP به صورت بدون اتصال (connection-less) می باشد. بدیهی است که سرعت پروتکل فوق نسبت به TCP سریع تر بوده ولی از بعد کنترل خطا، تنظیمات لازم را ارائه نخواهد داد. بهترین جایگاه استفاده از پروتکل UDP در مواردی است که برای ارسال و دریافت اطلاعات به یک سطح بالا از اطمینان نیاز نداشته باشیم.

TCP در کاربردهایی مانند اتصال به شبکه جهانی وب یا همان اینترنت، ارسال و دریافت ایمیل، پروتکل انتقال فایل (FTP)، ارسال فایل های چند رسانه ای (streaming media) و ... استفاده می شود.

در کاربردهای real-time مانند انتقال صدا در بستر شبکه (Voice over IP یا VOIP)، استفاده از پروتکل هایی مانند پروتکل انتقال همزمان (Real-time Transport Protocol یا RTP)، استفاده از UDP پیشنهاد می شود.

در پروتکل TCP برای این که از انتقال صحیح داده اطمینان حاصل شود، از تکنیکی به نام انتقال مجدد با تصدیق مثبت (Positive acknowledgement with re-transmission) استفاده می شود.

در این تکنیک پس از ارسال یک بسته توسط فرستنده، یک تایمر در فرستنده فعال می شود. گیرنده وظیفه دارد بلافاصله پس از دریافت بسته، یک پیام تصدیق به معنی دریافت موفقیت آمیز پیام برای

فرستنده ارسال کند. در صورتی که در مدت زمان مشخص شده توسط تایمر، فرستنده پیام تصدیق گیرنده را دریافت نکند، متوجه می شود که گیرنده پیام را به شکل صحیحی دریافت نکرده است و مجددا پیام را برای گیرنده ارسال می کند.

TCP وظیفه ایجاد سگمنت ها (بسته های کوچک داده که از تقسیم فریم ها تشکیل می شود) برای انتقال در بستر شبکه را دارد.

به عنوان مثال اگر وب سرور قصد ارسال یک فایل HTML را داشته باشد، پروتکل TCP این فایل را تبدیل به بسته های اطلاعاتی کوچک تر می کند و آنها را به پروتکل IP تحویل می دهد. پروتکل IP فرایند بسته بندی را برای تشکیل بسته های داده IP (از طریق اضافه کردن سرآیند پروتکل IP که شامل مواردی مانند IP مقصد می شود) روی بسته های دریافت شده از پروتکل TCP انجام می دهد. در گیرنده پس از دریافت بسته های داده، پروتکل TCP بسته ها را برای نداشتن خطا و همچنین مرتب بودن آنها (ordered) بررسی می کند. سپس سگمنت های مختلف به یکدیگر متصل می شوند تا داده اصلی (فایل HTML) تشکیل شود.

بسته های TCP مانند دیگر پروتکل ها شامل یک بخش سرآیند و یک بخش داده می باشد.

در بخش سرآیند ۱۰ بخش ضروری وجود دارد که حتما باید تنظیم شوند و بخش انتهایی سرآیند (که در جدول زیر به رنگ صورتی نشان داده شده است) اختیاری است.

بخش داده (Payload) که توسط لایه بالایی یعنی کاربرد تنظیم می شود، بعد از سرآیند قرار می گیرد.

نکته : همان طور که در جدول زیر مشخص است، در سرآیند پروتکل TCP ، اندازه بخش داده مشخص نمی شود. برای محاسبه آن می توان اندازه سرآیند IP و سرآیند TCP را از اندازه کل بسته های IP (که در بخش سرآیند پروتکل IP آورده می شود) کم کرد.

TCP Header

Offsets	Octet	0								1								2								3							
Octet	Bit	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22	23	24	25	26	27	28	29	30	31
0	0	Source port																Destination port															
4	32	Sequence number																															
8	64	Acknowledgment number (if ACK set)																															
12	96	Data offset				Reserved 0 0 0				NS	CWR	ECE	URG	ACK	PSH	RST	SYN	FIN	Window Size														
16	128	Checksum																Urgent pointer (if URG set)															
20 ...	160 ...	Options (if *data offset* > 5. Padded at the end with "0" bytes if necessary.) ...																															

جدول ۵-۱

Source port (16 bits) :

مشخص کننده پورت مورد استفاده برای ارسال داده ها

Destination port (16 bits) :

مشخص کننده پورت مورد استفاده برای دریافت داده ها

Sequence number (32 bits) :

به مقدار پرچم SYN بستگی دارد، اگر این پرچم یک باشد، این بخش نشان دهنده ی اولین شماره ترتیب (Sequence number) است. اگر پرچم SYN برابر صفر باشد، مقدار این بخش برابر شماره ترتیب اولین بایت داده segment است.

در پرتکل TCP شماره ترتیب، بر حسب شماره آخرین بایتی است که در بسته جاری قرار دارد. به عنوان مثال اگر در این بخش عدد ۱۹۳۴۱ قرار بگیرد به این معناست که داده‌ها تا بایت ۱۹۳۴۱ درون این بخش قرار دارند.

Acknowledgment number (32 bits) :

اگر پرچم ACK یک باشد، این بخش نشان دهنده ی شماره ترتیب بعدی برای ارسال می باشد. به عنوان مثال اگر در این بخش عدد ۱۲۳۶۵ قرار گرفته شود به این معنی می باشد که دادها تا بایت ۱۲۳۶۵ صحیح و کامل دریافت شده‌است و در انتظار بایت های ۱۲۳۶۷ به بعد می‌باشد.

Data offset (4 bits) :

تعیین کننده اندازه بخش سرآیند TCP برحسب کلمات ۳۲ بیتی (حداقل ۵ و حداکثر ۱۵ یا بر حسب بایت حداقل ۲۰ بایت و حداکثر ۶۰ بایت). به عنوان مثال اگر در این بخش عدد ۷ قرار بگیرد، طول سرآیند بسته برابر است با ۲۸=۷*۴ بایت خواهد شد.

Reserved (3 bits) :

غیرقابل استفاده و باید برابر صفر تنظیم شود.

Flags (9 bits) (aka Control bits) :

شامل ۹ پرچم برای انجام تنظیمات مختلف پروتکل TCP

Window size (16 bits) :

مقدار قرار گرفته در این بخش مشخص می‌کند که مقدار بافر گیرنده چند بایت دیگر فضای خالی دارد.

Checksum (16 bits) :

به منظور بررسی خطای اطلاعات دریافت شده در بخش سرآیند و داده

Urgent pointer (16 bits) :

اگر پرچم URG یک باشد، نشان دهنده ی offset بخش شماره ترتیب می باشد.

در این بخش عدد به عنوان اشاره گر قرار می‌گیرد که موقعیت داده‌های اضطراری را درون بسته مشخص می‌کند. این داده‌ها زمانی اتفاق می‌افتد و ارسال می‌شود که عملی شبیه وقوع وقفه در هنگام اجرای یک برنامه کاربردی رخ دهد. بدون آن که ارتباط قطع شود داده‌ها درون همین بسته جاری قرار گرفته و ارسال می‌شوند. لازم است ذکر شود که از این بخش لایه‌های بالاتر استفاده می‌کنند.

Options (Variable 0–320 bits, divisible by 32) :

تنظیمات اختیاری

Padding :

شامل بیت های صفر برای تکمیل کردن سرآیند TCP (در صورت نیاز)

نکته : با توجه به اتصال گرا بودن پروتکل TCP، باید ابتدا پروتکل یا فرایندی به نام Handshaking (تکان دادن دست) صورت پذیرد تا یک اتصال ایجاد شود.

۵-٤. پروتکل UDP

در پروتکل UDP (مخفف User Datagram Protocol یا قرارداد بسته داده کاربر) مشابه پروتکل TCP، گره های شبکه با قادر به ارسال پیام (که در این پروتکل آن را بسته داده یا Datagram می نامیم) در شبکه می باشند.

همان طور که در بخش پروتکل TCP اشاره شد، پروتکل UDP برخلاف پروتکل TCP، پروتکلی بدون اتصال (connectionless) می باشد که از مدل انتقال ساده بدون استفاده از تکنیک دست تکانی (Handshaking) که برای ایجاد قابلیت اطمینان (Reliability) ، مرتب‌سازی و یکپارچه سازی داده‌ها بکار می‌رود، بهره می‌جوید؛ بنابراین UDP سرویس غیرمطمئنی را ارائه می‌دهد و ممکن است بسته های داده نامرتب یا تکراری بوده و یا بدون اطلاع قبلی از دست بروند.

از UDP در کاربردهایی استفاده می شود که نیازی به انجام عملیات خطایابی و تصحیح خطا وجود ندارد یا این عملیات در سطح لایه های نرم افزاری صورت می پذیرد.

برنامه‌های که نسبت به زمان حساس هستند (time-critical) از UDP استفاده می‌کنند، زیرا از دست دادن بسته‌ها بهتر از منتظر ماندن برای بسته ها می باشد، بنابراین پروتکل UDP بهترین گزینه برای سیستم‌های بلادرنگ به حساب می‌آید. اگر برنامه‌ای نیاز به امکانات تصحیح خطا در سطح واسط شبکه داشته باشد، می‌تواند از پروتکل TCP یا SCTP (Stream Control Transmission Protocol) که به طور خاص برای این منظور طراحی شده است استفاده کند.

ساختار سرآیند پروتکل UDP مطابق شکل زیر می باشد.

UDP Header

Offsets	Octet	0								1								2								3							
Octet	Bit	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22	23	24	25	26	27	28	29	30	31
0	0	Source port																Destination port															
4	32	Length																Checksum															

جدول ۵-۲

سرآیند UDP شامل ۴ بخش است که هر بخش ۲ بایت ظرفیت دارد.

نکته : بخش های Checksum و Source port (به رنگ صورتی در جدول) در IPv4 دلخواه هستند و در IPv6 بخش source port دلخواه است.

: Source port number

این بخش شماره پورت فرستنده یا مبدا را مشخص می‌کند و زمانی معنا پیدا می‌کند که برای پاسخ دادن احتیاج به شماره پورت فرستنده داشته باشیم. اگر از آن استفاده نشود، عدد صفر در آن قرار می‌گیرد. اگر میزبان مبدأ یک کلاینت یا سرویس گیرنده باشد، شماره پورت به احتمال زیاد یک شماره پورت موقتی خواهد بود. اگر میزبان مبدأ یک سرور یا سرویس دهنده باشد، احتمالاً شماره پورت جزو پورت‌های عمومی شناخته شده خواهد بود.

: Destination port number

این بخش شماره پورت مقصد را تعیین می کند. همانند شماره پورت مبدأ، اگر میزبان مبدأ یک کلاینت باشد، شماره پورت به احتمال زیاد جزو پورت‌های موقتی خواهد بود و اگر میزبان مقصد یک سرور باشد، شماره پورت جزو پورت‌های عمومی شناخته شده خواهد بود. (در فصل چهارم در مورد شماره پورت های متداول توضیح داده شده است)

: Length

این بخش طول کل بسته داده را شامل سرآیند و بخش داده بر حسب بایت مشخص می کند. حداقل مقدار قابل تنظیم ۸ بایت است که برابر طول سرآیند می‌باشد. حداکثر مقدار این بخش از لحاظ تئوری برابر ۶۵۵۳۵ بایت (۸ بایت برای سرآیند + ۶۵۵۲۷ بایت برای داده) می باشد اما حداکثر اندازه عملی برای IPv4 برابر ۶۵۵۰۷ بایت و برای IPv6 امکان استفاده از اندازه بیش از ۶۵۵۳۵ بایت وجود دارد. طبق استاندارد RFC 2675 ، در صورتی که اندازه سرآیند و بخش داده بسته های UDP بیش از ۶۵۵۳۵ بایت می باشد، باید مقدار این بخش برابر صفر تنظیم شود.

: Checksum

این بخش برای عملیات Checksum برای بررسی خطای سرآیند و داده استفاده می‌شود. اگر از عملیات checksum استفاده نمی شود، باید این بخش برابر صفر تنظیم شود. این بخش در IPv4 اختیاری و در IPv6 الزامی است.

۵-۵. پروتکل ICMP

پروتکل ICMP (مخفف Internet Control Message Protocol یا پروتکل کنترل پیام‌های اینترنتی)، مسئول ارائه توابع عیب یابی و گزارش خطاء در صورت عدم توزیع صحیح اطلاعات است.

ساختار سرآیند این پروتکل مطابق شکل زیر می باشد.

Offsets	Octet	0								1								2								3							
Octet	Bit	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22	23	24	25	26	27	28	29	30	31
0	0	Type								Code								Checksum															
4	32	Rest of Header																															

جدول ۵-۳

همه ی بسته های ICMP دارای بسته های داده با سرآیند ۸ بایت و بخش داده متغیر می باشند. ۴ بایت اول سرآیند دارای فرمتی ثابت است اما ۴ بایت بعدی با توجه به نوع بسته ICMP تغییر می کند.

Type :

تعیین کننده ی نوع پیام کنترلی می باشد.

Code :

به همراه مشخصه type تعیین کننده ی نوع پیام کنترلی می باشد.

Checksum :

بیت های مربوط به کنترل خطا

Rest of Header :

۴ بایت دیگر که بستگی به نوع type و code استفاده شده دارد.

نکته : در بخش type و code با توجه به نوع پیام، مقادیر متفاوتی باید قرار گیرد. دو نوع متداول پیام که در این پروژه نیز از آنها استفاده شده است عبارت است از پیام درخواست (Echo Request) و پیام

پاسخ (Echo Reply) که برای پیاده سازی عملیات ping در شبکه از آنها استفاده می شود. برای پیام درخواست مقدار type باید برابر عدد ۸ و مقدار code باید برابر عدد صفر تنظیم شود و برای پیام پاسخ هر دو بخش type و code باید برابر عدد صفر تنظیم شوند.

جدول زیر ساختار پیام درخواست را نشان می دهد.

00 01 02 03 04 05 06 07	08 09 10 11 12 13 14 15	16 17 18 19 20 21 22 23 24 25 26 27 28 29 30 31
Type = 8	Code = 0	Header Checksum
Identifier		Sequence Number
Payload		

جدول ۵-۴

مقدار identifier یا شناسه و sequence number یا شماره ترتیب به شکل اختیاری انتخاب می شوند. البته گاهی اوقات لازم است برای یکسان بودن با مقادیر در پیام ها، در مقدار مشخصی مطابق با پیام درخواست تنظیم شوند.

جدول زیر ساختار پیام پاسخ را نشان می دهد.

00 01 02 03 04 05 06 07	08 09 10 11 12 13 14 15	16 17 18 19 20 21 22 23 24 25 26 27 28 29 30 31
Type = 0	Code = 0	Header Checksum
Identifier		Sequence Number
Payload		

جدول ۵-۵

نکته : بخش داده پیام پاسخ باید دقیقا برابر بخش داده پیام درخواست باشد.

نکته : ping عملیاتی است رایج در شبکه های کامپیوتری برای بررسی امکان دسترسی به شبکه و همچنین محاسبه round-trip time (مدت زمان لازم برای رفت و برگشت سیگنال). عملیات ping توسط پیام های Echo Request و Echo Reply در پروتکل ICMP پیاده سازی می شود.

علت نام گذاری نیز از نظر تاریخی مربوط به تکنیک سونار می باشد که در آن یک سیگنال پالس به عمق دریا فرستاده می شود و منتظر دریافت انعکاس (echo) سیگنال پس از برخورد با اجسام داخل آب می ماند.

بخش داده شامل سرآیند پروتکل IPv4 به همراه ۸ بایت اول بسته های داده پروتکل IPv4 که باعث ایجاد خطا شده است می باشد.

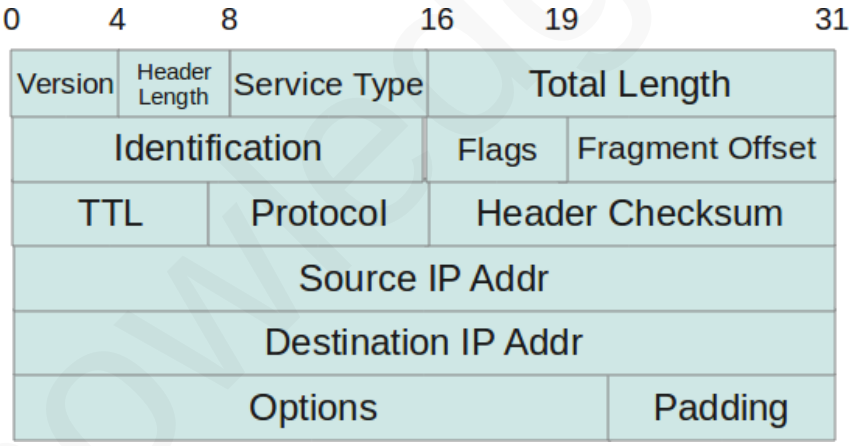
۵-۶. پروتکل IP

پروتکل IP (مخفف Internet Protocol)، یکی از پروتکل های اصلی Internet protocol suite یا مدل TCP/IP می باشد که وظیفه هدایت بسته ها را در شبکه را به عهده دارد. IP برای هدایت بسته ها در شبکه از آدرس IP استفاده می کند. IP یک پروتکل بدون اتصال است.

پروتکل IP پس از دریافت اطلاعات از TCP ، شروع به قطعه قطعه کردن آن به قطعه های کوچک به اسم fragment می نماید، پس از این مرحله برای هر فرگمنت یک بسته IP می سازد که حاوی اطلاعات مورد نیاز بسته برای حرکت در طول شبکه می باشد و بسته IP را به بسته TCP اضافه می کند و شروع به ارسال این فرگمنت ها می نماید.

نسخه غالب IP که بیشتر مورد استفاده قرار می گیرد IPv4 می باشد. در حال حاضر IPv6 نیز ارائه شده است.

ساختار سرآیند پروتکل IPv4 مطابق شکل زیر می باشد.



جدول ۵-۶

سرآیند پروتکل IPv4 شامل ۲۰ بایت می باشد که قابلیت افزایش آن نیز وجود دارد که البته معمولا از این قابلیت استفاده نمی شود.

Version :

برای IPv4 باید مقدار ۴ در این بخش تنظیم شود.

: Header Length

اندازه بخش سرآیند بر حسب کلمات ۳۲ بیتی (۴ بایتی)، معمولا برابر عدد ۵ (۲۰ بایت)

: Service Type

یکی از مشخصه های تنظیم کننده پارامتر QOS ، معمولا برابر عدد صفر تنظیم میشود.

: Total Lenght

اندازه مجموعه بخش سرآیند و داده برحسب بایت

: Identification

عددی دو بایتی که به همراه آدرس IP فرستنده، مشخص کننده بسته در شبکه است و هنگام مونتاژ بسته های دریافتی در گیرنده مورد استفاده قرار می گیرد.

: Flags

شامل ۴ بیت که ۳ بیت آن مورد استفاده قرار می گیرد، به عنوان مثال از پرچم DF (Don't Fragment) برای عدم اجازه به روتر در شبکه برای تقسیم بسته های داده به بسته های کوچکتر استفاده می شود.

: Fragment Offset

شمارنده بسته های ارسالی که توسط روتر تنظیم می شود.

: (Time To Live) TTL

TTL مکانیزمی است که با استفاده از آن مدت زمان حضور یک بسته داده در شبکه محدود می شود تا از گردش بی جهت یک داده در شبکه و اشغال پهنای باند جلوگیری شود. این عدد نشان دهنده ی حداکثر تعداد تجهیزات شبکه است (مانند hup ، switch و...) که مجاز به ارسال و دریافت این بسته داده هستند.

: Protocol

نشان دهنده ی نوع پروتکل مورد استفاده در لایه انتقال می باشد. عدد ۱ برای پروتکل ICMP ، عدد ۲ برای پروتکل IGMP ، عدد ۶ برای پروتکل TCP و عدد ۱۷ برای پروتکل UDP

: Header Checksum

محاسبات checksum برای بخش سرآیند IP

نکته : در صورتی که نتایج محاسبات checksum در گیرنده غلط باشد، گیرنده آن بسته را رها می کند.

: Source IP Addr

آدرس IP فرستنده

: Destination IP Addr

آدرس IP گیرنده

: Options

برای ایجاد تنظیمات بیشتر، معمولا استفاده نمی شود.

: Padding

برای اضافه کردن بیت های صفر به انتهای سرآیند در صورت ناکافی بودن اندازه سرآیند

۵-۷. پروتکل ARP

ARP (Address Resolution Protocol یا پروتکل تفکیک آدرس)، وظیفه بدست آوردن آدرس

MAC با داشتن آدرس IP یک گره در شبکه را دارد. این پروتکل مشابه برخی پروتکل های دیگر مانند HTTP بر مبنای درخواست (request) و پاسخ (reply) عمل می نماید.

همان طور که در فصل دوم نیز توضیح داده شده، آدرس IP یک شناسه منطقی است و در لایه شبکه کار می‌کند. بسته های IP در فریم های لایه فیزیکی قرار می‌گیرند و سپس درون شبکه ارسال می‌شوند، اما باید به این نکته توجه کرد که دستگاه‌های درون شبکه برای ارتباط با یکدیگر نیازمند دانستن آدرس سخت‌افزاری یا همان آدرس MAC یکدیگر هستند. این در حالیست که ما از کامپیوترها و دستگاه‌هایی مانند سوییچ، مودم، روتر، پرینتر و ... که در شبکه با آن‌ها کار می‌کنیم فقط IP آنها را می‌دانیم در حالی که در عمل به شناسه سخت‌افزاری یا آدرس MAC آنها نیاز است. در نتیجه باید به روشی آدرس MAC آن سیستم را به دست آوریم. برای این منظور باید یک درخواست با آدرس MAC مقصد برابر FF:FF:FF:FF:FF:FF (که از نوع broadcast است و توسط همه ی گره ها دریافت و مورد پردازش قرار می گیرد) و آدرس IP مقصد ارسال کنیم. گیرنده ای که آدرس MAC آن را میخواهیم پس از دریافت این پیام، آدرس MAC به همراه آدرس IP خود را برای ما ارسال می کند.

به عنوان مثال برای فهمیدن نحوه گرفتن به دست آوردن آدرس MAC یک سیستم در شبکه با در دست داشتن آدرس IP آن، عملیات ping را می توان در این مورد مثال زد.

فرض کنیم IP فرستنده ۱۹۲,۱۶۸,۳,۳ و IP گیرنده ۱۹۲,۱۶۸,۳,۴ باشد. حال اگر ما بخواهیم دستور ping 192.168.3.4 را از سیستم ۱۹۲,۱۶۸,۳,۳ اجرا کنیم، نیاز به دانستن آدرس MAC سیستم ۱۹۲,۱۶۸,۳,۴ داریم.

سیستم ۱۹۲,۱۶۸,۳,۳ در صورت نداشتن آدرس MAC سیستم 192.168.3.4، یک پیام به صورت broadcast در کل شبکه ارسال می‌کند مبنی بر این که سیستم دارای آدرس IP برابر 192.168.3.4 چه کسی است، به من بگوید من ۱۹۲,۱۶۸,۳,۳ هستم.

```
who-has 192.168.3.4 tell 192.168.3.3
```

سپس این سیستم طی پیامی به عنوان جواب، آدرس MAC خود را به ما اعلام می کند.

```
reply 192.168.3.4 is-at 88:ae:1d:34:1d:7c
```

و سیستم ما این آدرس MAC را در جدولی با نام ARP Table (یا translation table) ذخیره

می‌کند.

به این فرایند که طی آن با در دست داشتن آدرس IP یک گره، آدرس MAC آن را به دست می آوریم، اصطلاحا Who-is گفته می شود. (پاراگراف زیر در مورد ادامه این فرایند توضیح می دهد اما دانستن آن برای این مقاله لازم نیست و به عنوان توضیح بیشتر آورده شده است)

حال در صورتی که سیستم ما تشخیص دهد که سیستم مورد نظر با سیستم ما در یک شبکه قرار ندارند (یعنی subnetmask متفاوت دارند)، ابتدا Route Table خود را می‌بیند و در صورتی که در این جدول موردی برای رسیدن به این IP موجود باشد، بررسی می‌کند که برای ارتباط با این IP باید به سمت کدام روتر اطلاعات را بفرستد، سپس با استفاده از همان مراحل گفته شده آدرس MAC روتر مورد نظر را با توجه به داشتن IP آن پیدا می‌کند و در Destination MAC address آدرس مربوط به روتر و در بخش destination IP address مقصد سیستمی که می‌خواهد با آن ارتباط داشته باشد را قرار می‌دهد.

شکل زیر ساختار سرآیند پروتکل ARP را در پروتکل IPv4 نشان می دهد.

Internet Protocol (IPv4) over Ethernet ARP packet		
octet offset	0	1
0	Hardware type (HTYPE)	
2	Protocol type (PTYPE)	
4	Hardware address length (HLEN)	Protocol address length (PLEN)
6	Operation (OPER)	
8	Sender hardware address (SHA) (first 2 bytes)	
10	(next 2 bytes)	
12	(last 2 bytes)	
14	Sender protocol address (SPA) (first 2 bytes)	
16	(last 2 bytes)	
18	Target hardware address (THA) (first 2 bytes)	
20	(next 2 bytes)	
22	(last 2 bytes)	
24	Target protocol address (TPA) (first 2 bytes)	
26	(last 2 bytes)	

جدول ۵-۷

Hardware type (HTYPE) :

نوع پروتکل شبکه مورد استفاده، برای پروتکل Ethernet برابر ۱ تنظیم می شود.

Protocol type (PTYPE) :

نوع پروتکلی که ARP از آن استفاده می کند. برای IPv4 برابر 0x0800 تنظیم می شود.

Hardware length (HLEN) :

طول آدرس سخت افزاری، برای آدرس MAC برابر ۶ بایت

Protocol length (PLEN) :

طول آدرس پروتکل، در IPv4 طول آدرس IP برابر ۴ بایت می باشد.

Operation :

نوع عملیات یا پیام ارسالی : ۱ برای پیام درخواست و ۲ برای پاسخ پاسخ

Sender hardware address (SHA) :

آدرس MAC فرستنده

Sender protocol address (SPA) :

آدرس IP فرستنده

Target hardware address (THA) :

آدرس MAC گیرنده

Target protocol address (TPA) :

آدرس IP گیرنده

نکته : نوع اترنت (EtherType) بسته های پروتکل ARP برابر عدد 0x0806 می باشد.

نکته : طول بسته های ARP بستگی به آدرس های IP و MAC دارد. معمولا نسخه پروتکل مورد استفاده به عنوان IP نسخه چهارم IP (IPv4) می باشد، در اینصورت ۴۸ بیت برای آدرس MAC فرستنده و گیرنده و ۳۲ بیت برای آدرس IP فرستنده و گیرنده استفاده می شود و طول بسته ها برابر ۲۸ بایت می باشد.

فصل ششم

سخت افزار

٦-١. مقدمه

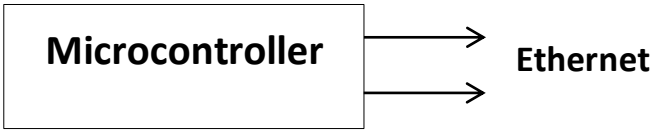
در این فصل به مرور روش های مختلف پیاده سازی سخت افزار لازم برای اتصال به شبکه اترنت می پردازیم.

در ابتدای روش های طراحی سخت افزار را با توجه به تراشه های مورد استفاده در طراحی تقسیم بندی کرده و در ادامه فصل برای هر روش، تراشه هایی به منظور پیاده سازی سخت افزاری معرفی می شود.

٦-٢. بلوک دیاگرام اتصال به شبکه اترنت

به طور کلی با توجه به تراشه های مورد استفاده در طراحی، چهار حالت برای اتصال به شبکه اترنت در کاربردهای embedded وجود دارد.

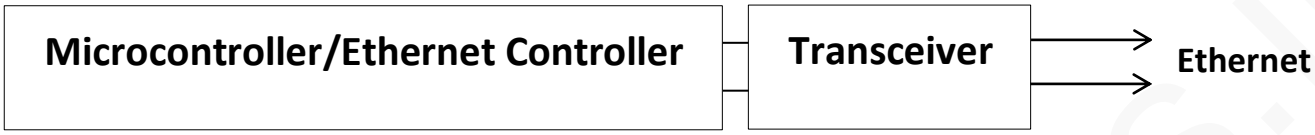
حالت اول :



شکل ۶-۱

در این حالت لایه های فیزیکی و پیوند داده (شامل دو زیر لایه MAC و LLC) در میکروکنترلر وجود دارد و نیازی به استفاده از تراشه های جانبی وجود ندارد.

حالت دوم :



شکل ۶-۲

در این حالت لایه پیوند داده در میکروکنترلر و لایه فیزیکی توسط تراشه Transceiver پیاده سازی می شود، البته معمولا حالت اول متداول تر است و میکروکنترلرهایی که دارای قابلیت اترنت هستند، لایه فیزیکی را نیز پیاده سازی می کنند.

حالت سوم :

شکل ۶-۳

در این حالت لایه پیوند داده و لایه فیزیکی توسط تراشه Ethernet Controller پیاده سازی می شود.

حالت چهارم :

شکل ۶-۴

در این حالت لایه پیوند داده در تراشه Ethernet Controller و لایه فیزیکی در تراشه Transceiver پیاده سازی می شود.

٦-٣. چند نکته برای اتصال به شبکه اترنت

پیش از معرفی تراشه های مختلف برای پیاده سازی اترنت، چند نکته در مورد اتصال به شبکه ایترنت را بررسی می کنیم.

به طور معمول برای جلوگیری از آسیب دیدن تراشه های مورد استفاده برای اتصال به شبکه در اثر عواملی مانند نویزها، القای الکتریسیته ساکن و... از سمت شبکه، باید در مسیر تراشه و کانکتور RJ-45 فیلترها و مدارات مناسب قرار گیرند.

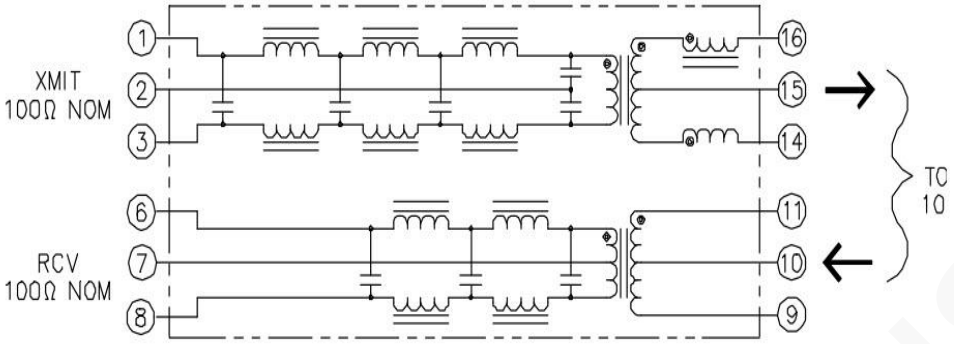
فیلتری که برای این منظور معمولا استفاده می شود را چوک یا ترانس شبکه می نامند و حداقل دو فیلتر پایین گذر (Low Pass Filter) ایزوله از هم، یکی در مسیر ارسال و دیگری در مسیر دریافت تشکیل می گردد. دلایل عمده ضروری بودن فیلتر عبارت اند از :

- رفتار پایین گذر فیلتر در مسیر ارسال، تشعشعات رادیویی (RFI) حاصل از پالس های مربعی را که توسط تراشه شبکه تولید می شود را کاهش می دهد. (پالس های مربعی علاوه بر فرکانس پایه، طبق قضیه فوریه، دارای فرکانس های بالاتری نیز می باشند که امواج الکترومغناطیسی بیشتری ایجاد می کنند)
- نویزهای فرکانس های بالای ورودی نیز به دلیل خاصیت پایین گذر مسیر دریافت به سیستم کاهش می یابند و از تفسیر غلط آنها در تراشه جلوگیری می شود.
- در صورت بروز شوک الکتریکی مانند ساعقه یا خرابی دستگاه های دیگر روی شبکه یا اتصال آنها با برق، این تغییرات به دلیل ایزوله بودن به مدار وارد نمی شوند. به این ترتیب از بروز آسیب دائمی مدار جلوگیری می گردد.
- ایزولاسیون الکتریکی مدار از قسمت های دیگر شبکه باعث حذف شدن حلقه زمین (Ground Loop) می شود و اثر نویزهای موجود در محیط را کاهش می دهد.

نکته : ممکن است مدار در صورت نصب نکردن فیلتر نیز در فواصل کوتاه کار کند، البته این کار به هیچ وجه توصیه نمی شود.

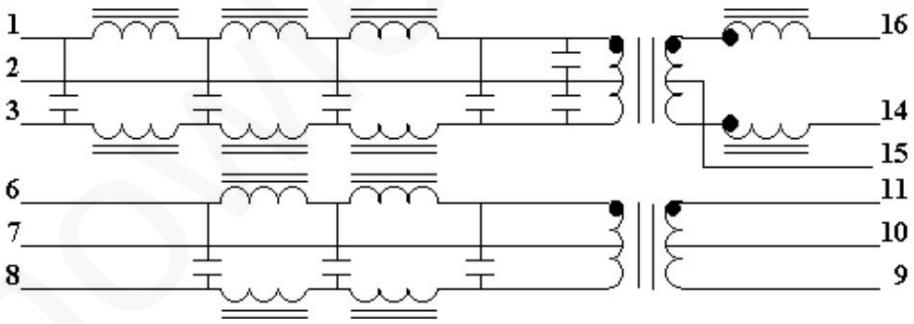
برای سرعت های پایین مانند 10Mbit/s معمولا از فیلتر FL1012 یا 20F001N استفاده می شود که علاوه بر تشابه ترتیب پایه ها، عملکرد مشابهی دارند و می توانند به جای هم نیز استفاده شوند.

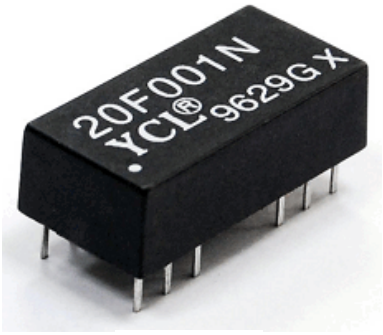
شکل زیر ساختار داخلی فیلتر FL1012 و یک نمونه از آن را نشان می دهد.



شکل ۶-۵

شکل زیر ساختار داخلی فیلتر 20F001N و یک نمونه از آن را نشان می دهد.

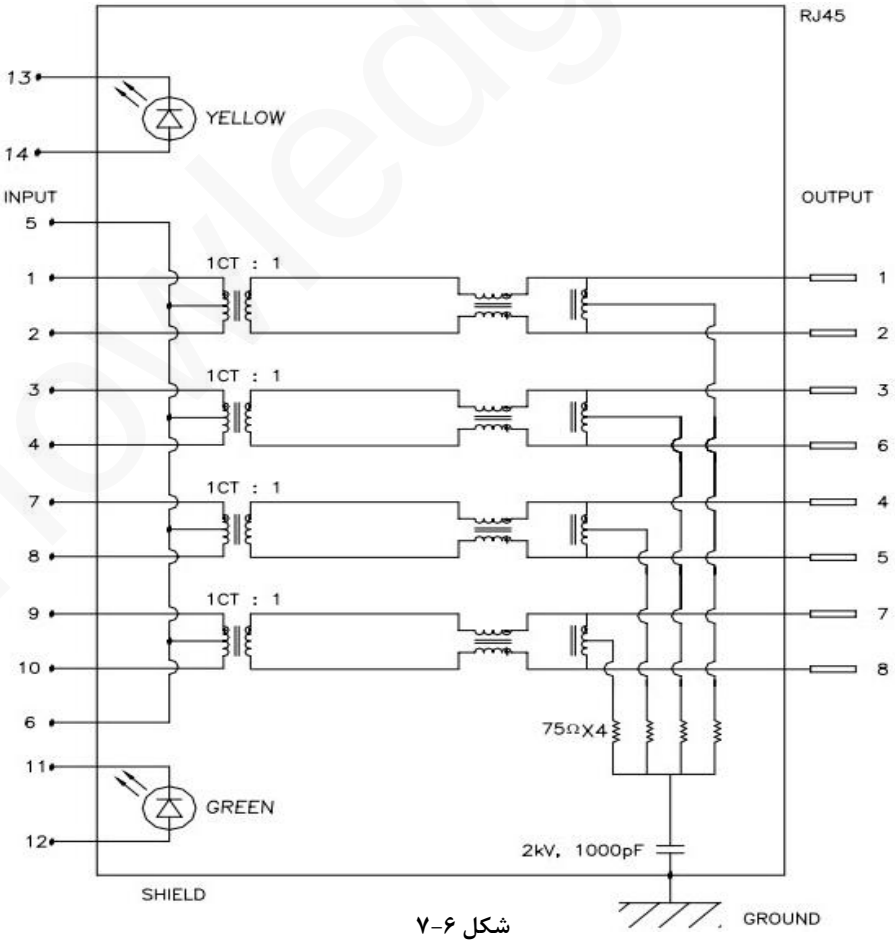




شکل ۶-۶

سمت چپ مربوط به مدار داخلی دو فیلتر که با شماره های 1,2,3,6,7,8 مشخص شده است باید به کانکتور RJ-45 متصل شود و بخش سمت راست باید به تراشه اترنت متصل گردد. پایه های شماره ۱ و ۳ به همراه پایه مشترک ۲ برای سیگنال فرستنده (TX) و پایه های شماره ۶ و ۸ به همراه پایه مشترک ۷ برای سیگنال گیرنده (RX) می باشند که باید به شکل صحیح و توسط مدارات مناسب، به کانکتور RJ-45 متصل گردند. به همین ترتیب پایه های شماره ۱۶ و ۱۴ به همراه پایه مشترک ۱۵ برای سیگنال فرستنده (TX) و پایه های شماره ۱۱ و ۹ به همراه پایه مشترک ۱۰ برای سیگنال فرستنده (RX) می باشند که باید به شکل صحیح و توسط مدارات مناسب به پایه های تراشه اترنت متصل گردند. در مورد مدارات جانبی مورد استفاده در بین تراشه اترنت و کانکتور RJ-45 در بخش های بعدی به همراه معرفی تراشه ها آشنا خواهیم شد.

نکته : برخی از کانکتورهای RJ-45 دارای فیلتر داخلی نیز می باشند که نیاز به فیلتر مجزا را رفع می سازند. این نوع از کانکتورها معمولا MAG-JACK نامیده می شوند و از برتری هایی نظیر کوچکتر شدن مدار، کمتر شدن تعداد المان ها و زیبایی برخوردارند. P65-101-AK9 از جمله این کانکتورها می باشد که ساختار داخلی آن در شکل زیر نشان داده شده است.



شکل ۶-۷

این کانکتور علاوه بر داشتن فیلتر داخلی، دارای دو عدد LED به رنگ های سبز و زرد نشان دهنده ی وضعیت اتصال نیز می باشد.

شکل زیر یک نمونه از کانکتور RJ-45 که دارای LED های وضعیت شبکه است را نشان می دهد.

شکل ۶-۸

مطابق شکل ساختار داخلی کانکتور در صفحه قبل، پایه های سمت چپ که شامل LED های وضعیت شبکه نیز می شود باید به تراشه اترنت متصل گردند و پایه های سمت راست باید به کابل شبکه متصل گردد.

نکته : مطابق شکل بالا، کانکتور RJ-45 دارای ۸ پایه برای اتصال به شبکه اترنت می باشد که همان طور که در فصل چهارم توضیح داده شد، در استانداردهای سرعت های پایین مانند 10Base-T از ۲ زوج سیم یا ۴ سیم (۴ پایه از این کانکتور) و در استاندارد سرعت های بالاتر از هر ۴ زوج سیم یا هر ۸ سیم (۸ پایه کانکتور) باید استفاده شود.

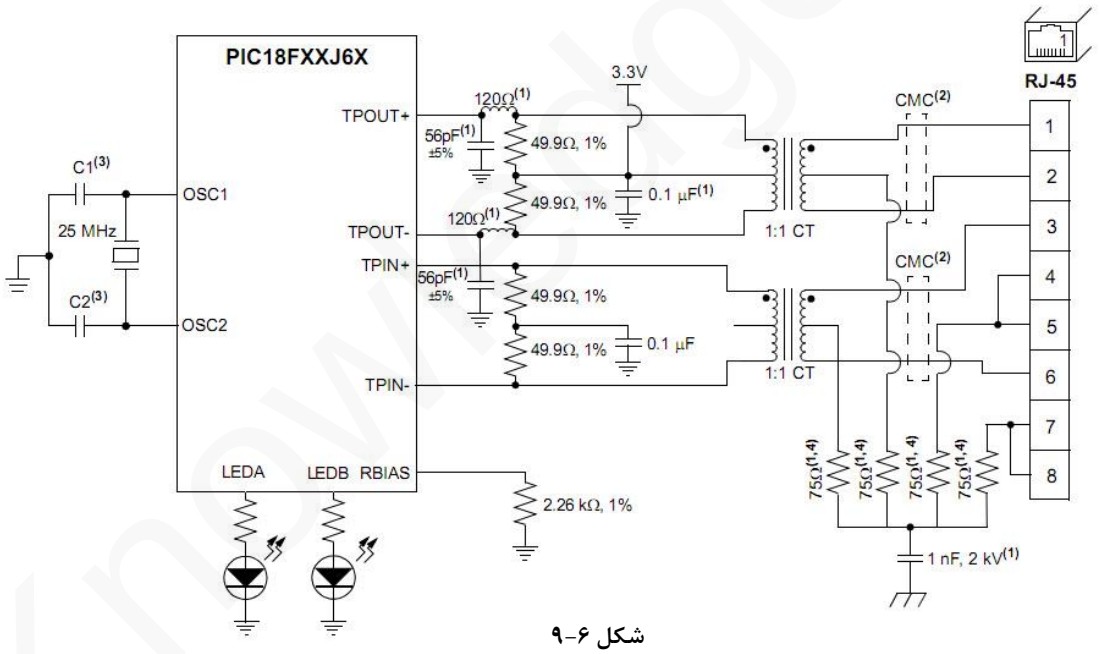
٦-٤. معرفی تراشه های مورد نیاز برای حالت پیاده سازی اول

٦-٤-١. میکروکنترلر PIC18F87J60

- کنترل کننده اترنت به همراه لایه فیزیکی
- پشتیبانی از سرعت های 10Mbit/s (10Base-T)
- سازگاری با استاندارد IEEE 802.3

پایه های TPIN+ و TPIN- به همراه پایه های TPOUT+ و TPOUT- برای اتصال به شبه اترنت به کار می روند.

شکل زیر اتصال میکروکنترلر را به شبکه نشان می دهد.



شکل ۶-۹

٦-٤-٢. میکروکنترلر PIC18F97J60

مشابه میکروکنترلر PIC18F87J60

٦-٥. معرفی تراشه های مورد نیاز برای حالت پیاده سازی دوم

در این میکروکنترلرها فقط لایه پیوند داده اترنت پیاده سازی شده است و برای استفاده از این تراشه برای اتصال به شبکه اترنت، باید از تراشه های transceiver (که در بخش های بعدی نمونه های مختلفی معرفی می شود) در کنار این میکروکنترلرها استفاده شود.

٦-٥-١. میکروکنترلرهای LPC1769/68/67/66/65/64/63

- پشتیبانی از سرعت های 10Mbit/s (استاندارد 10Base-T) و 100Mbit/s (استانداردهای 100Base-TX ، 100Base-FX و 100Base-T4)
- تطابق با استاندارد IEEE 802.3

پایه هایی که با ENET شروع می شوند مربوط به ارتباط اترنت می شود. پایه های ENET_TXD0 و ENET_TXD1 برای ارسال داده ها به تراشه transceiver و پایه های ENET_RXD0 و ENET_RXD1 برای دریافت داده های شبکه از تراشه transceiver به کار می رود.

٦-٥-٢. میکروکنترلرهای LPC2364/66/68

مشابه سری LPC1769/68/67/66/65/64/63

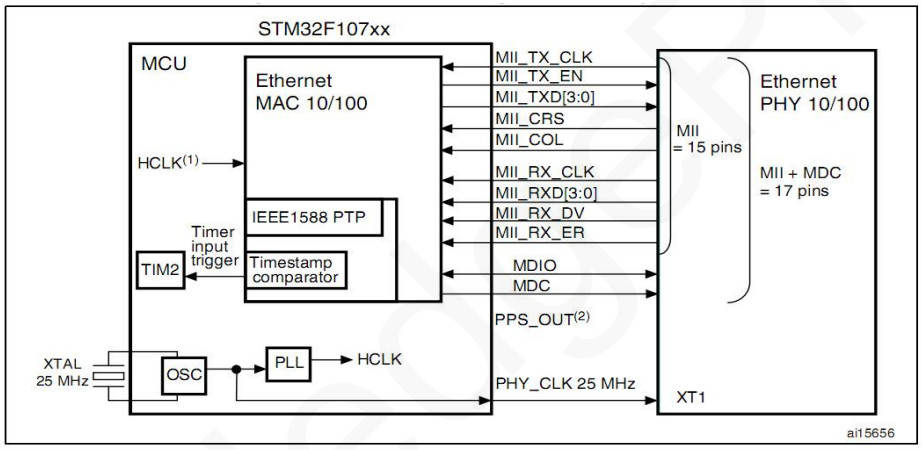
٦-٥-٣. میکروکنترلرهای سری STM32F105xx و STM32F107xx

- پشتیبانی از سرعت های 10Mbit/s و 100Bmit/s

اتصال بین میکروکنترلر و تراشه transceiver از طریق حداکثر ۱۷ پایه (بخش MII) یا حداقل ۹ پایه (RMII) صورت می پذیرد.

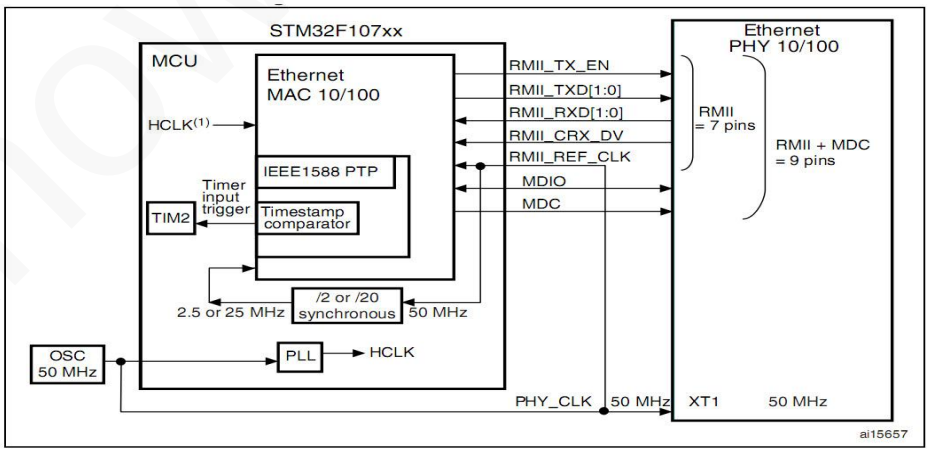
شکل های زیر انواع روش های اتصال این میکروکنترلر را به تراشه transceiver نشان می دهد.

شکل اول استفاده از پایه های MII و کریستال 25MHZ را نشان می دهد.



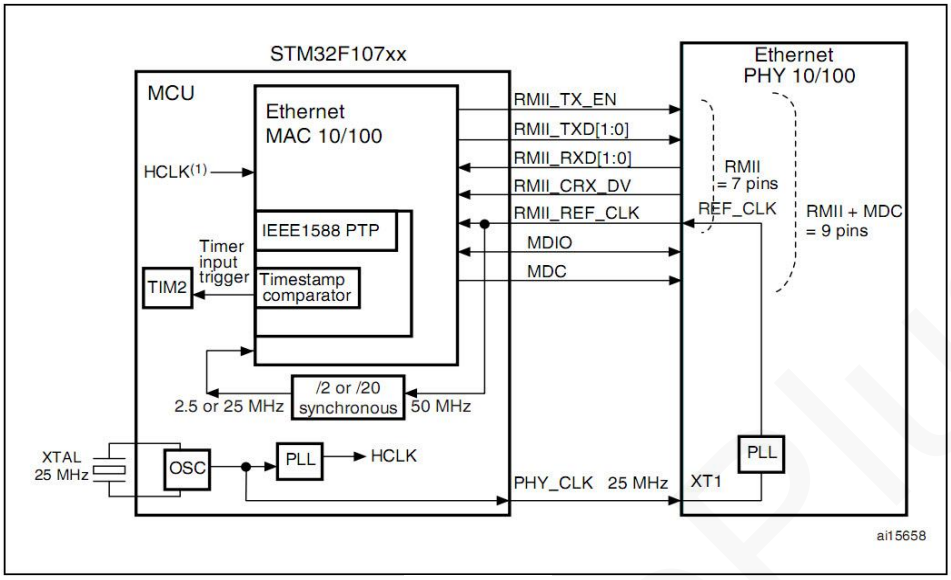
شکل ۶-۱۰

این شکل استفاده از پایه های RMII و کریستال 50MHZ را نشان می دهد.



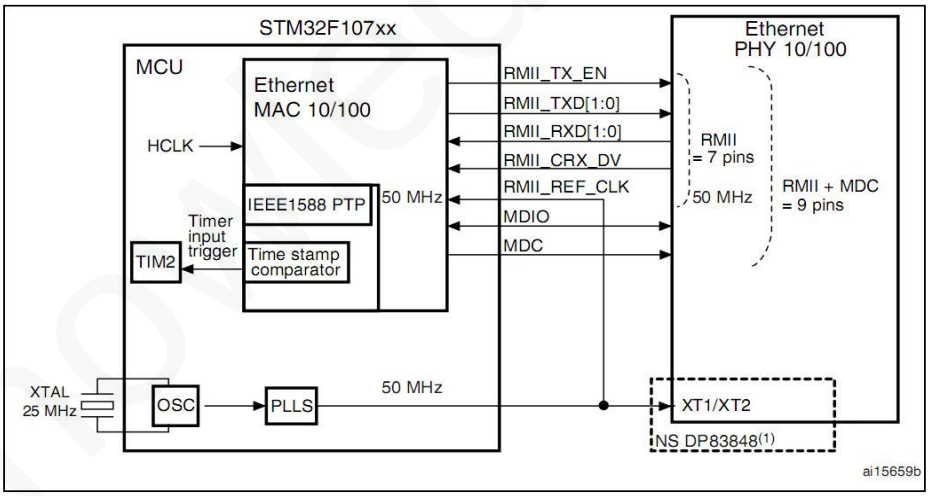
شکل ۶-۱۱

این شکل استفاده از پایه های RMII ،کریستال 25MHZ و استفاده از PLL در تراشه transceiver را نشان می دهد.



شکل ۶-۱۲

این شکل استفاده از پایه های RMII و کریستال 25MHZ را نشان می دهد. تراشه transceiver پیشنهاد داده شده در این شکل NS DP83848 می باشد که از نظر jitter با خروجی تراشه تطابق دارد.



شکل ۶-۱۳

٦-٦. معرفی تراشه های مورد نیاز برای حالت پیاده سازی سوم

همان طور که می دانیم، به تراشه هایی که پردازش های لازم به منظور ارتباط با شبکه اترنت را انجام می دهند، کنترل کننده اترنت (Ethernet Controller) می گویند.

در این قسمت تراشه های Ethernet Controller که دارای بخش transceiver نیز هستند معرفی می شوند.

با توجه به سرعت مورد نیاز، پایه های آزاد میکروکنترلر، نحوه ی ارتباط با میکروکنترلر و حتی توابع کتابخانه ای موجود در زبان برنامه نویسی، می توان یکی از آنها را انتخاب کرد و به کار گرفت.

٦-٦-١. تراشه ENC28J60

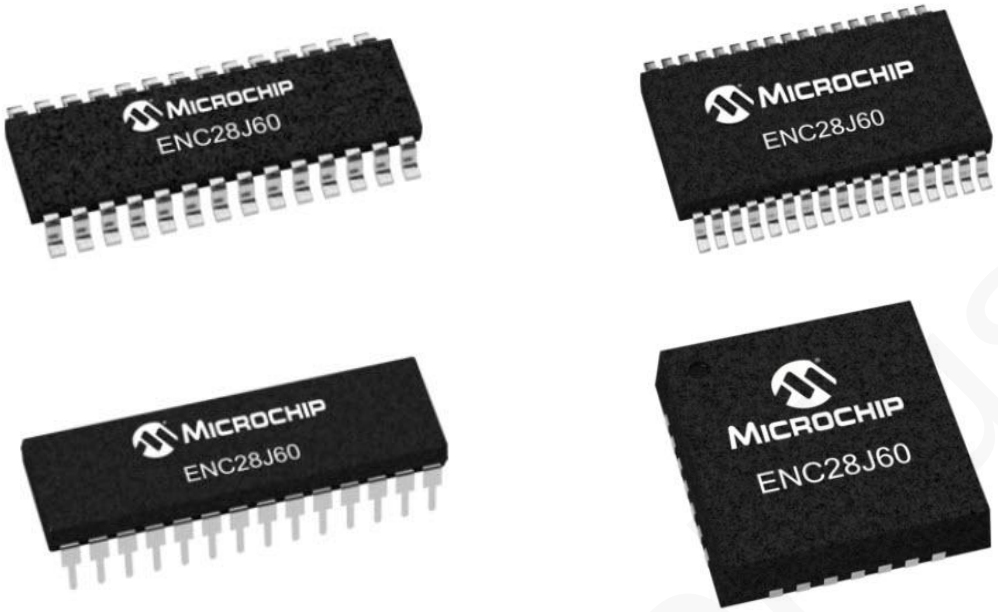
این تراشه توسط شرکت Microchip (که اخیرا شرکت Atmel را نیز خریداری کرده است) برای استفاده در مدارات سرعت پایین که به ارتباط با شبکه اترنت نیاز دارند طراحی و ساخته شده است. این تراشه که به عنوان یک کنترل کننده ی کامل اترنت به شمار می رود، به راحتی راه اندازی می شود و می تواند با داشتن حافظه داخلی کافی برای سیستم هایی که به سرعت و حافظه بالای پردازش اطلاعات نیاز ندارند به کار رود.

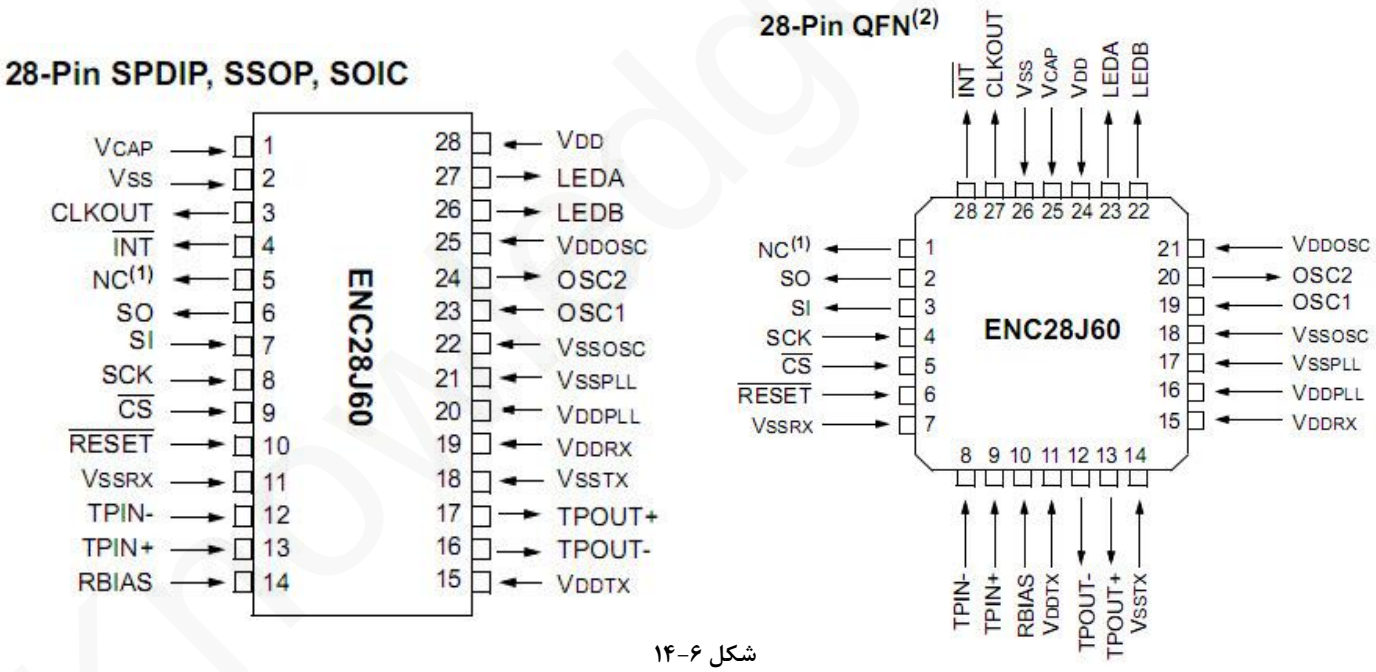
لازم به ذکر است همان طور که اشاره شد، علاوه بر لایه پیوند داده، لایه ی فیزیکی نیز در این تراشه پیاده سازی شده است و لایه های بالاتر از لایه ی شبکه در میکروکنترلر به صورت نرم افزاری پیاده سازی می شود.

از جمله مشخصات بارز این کنترل کننده می توان به موارد زیر اشاره کرد :

- سازگار با استاندارد IEEE 802.3 (سرعت انتقال داده 10Mbit/s)
- پیاده سازی هر دو لایه ی MAC و لایه ی فیزیکی 10BASE-T
- پشتیبانی از حالت های Half-Duplex و Full-Duplex
- ارتباط SPI با سرعت 20MHz
- ٨ کیلوبایت حافظه داخلی
- بسته بندی های ٢٨ پایه SPDIP/SOIC/SSOP/QFN

انواع بسته بندی ها و ترتیب پایه های این تراشه در شکل های زیر نشان داده شده است.





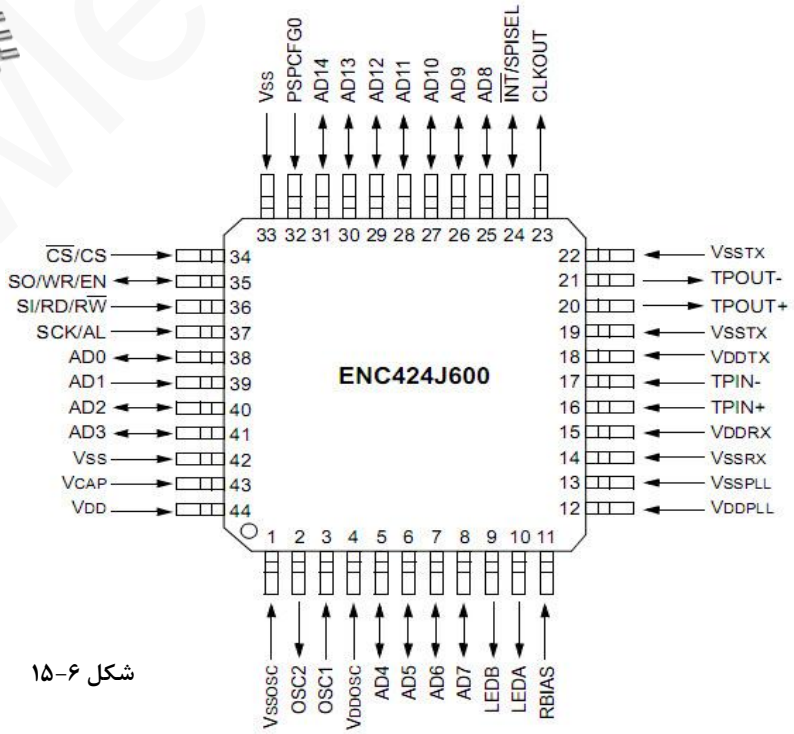
شکل ۶-۱۴

نکته : با توجه به پارامترهایی چون هزینه، سهولت ساخت نمونه، و تعداد پایه های ارتباطی، از این تراشه برای انجام پروژه های عملی فصل آخر استفاده می شود. به همین منظور درفصل هفتم به طور کامل در مورد مشخصات و نحوه ی کار با این تراشه توضیحاتی ارائه خواهد شد.

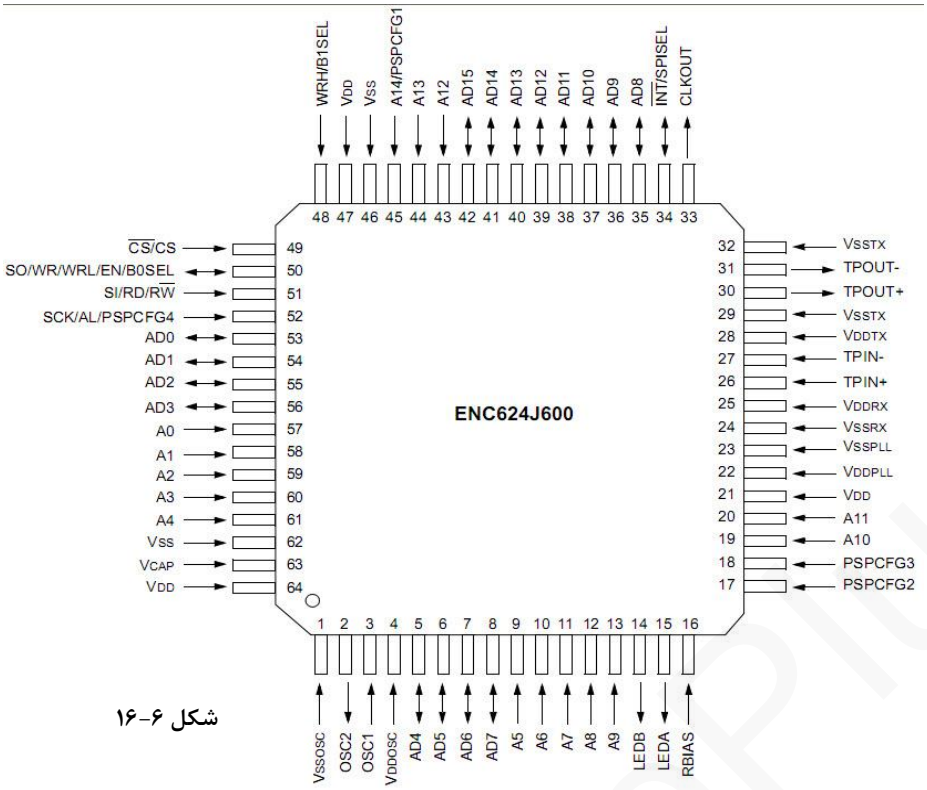
٦-٦-٢. تراشه ENC424J600/626J600

- پشتیبانی از سرعت های 10Mbit/s و 100Mbit/s (10Base-T ، 100Base-TX)
- تطابق با استاندارد IEEE 802.3
- پشتیبانی از AUTO-MDIX
- پشتیبانی از ارتباط SPI
- محدوده تغذیه 3v تا 3.6v
- قابلیت تحمل ولتاژ 5v در پایه های I/O
- بسته بندی QFP و QFN ۴۴ پایه و بسته بندی TQFP ۶۴ پایه

ترتیب پایه ها و بسته بندی این تراشه های ۶-۱۵ و ۶-۱۶ نشان داده شده است.



شکل ۶-۱۵



شکل ۶-۱۶

تنها تفاوت دو تراشه ENC424J600 و ENC624J600 در تعداد پایه های واسط موازی است.

جدول زیر مقایسه ای بین این دو تراشه را نشان می دهد.

Feature	ENC424J600	ENC624J600
Pin Count	44	64
Ethernet Operating Speed	10/100 Mbps (auto-negotiate, auto-sense or manual)	
Ethernet Duplex Modes	Half and Full (auto-negotiate and manual)	
Ethernet Flow Control	Pause and Backpressure (auto and manual)	
Buffer Memory (bytes)	24K (organized as 12K word x 16)	
Internal Interrupt Sources	11 (mappable to a single external interrupt flag)	
Serial Host Interface (SPI)	Yes	Yes
Parallel Host Interface:		
Operating modes	2	8
Muliplexed, 8-bit	Yes	Yes
16-bit	No	Yes
Demultiplexed, 8-bit	No	Yes
16-bit	No	Yes
Cryptographic Security Options:		
AES, 128/192/256-bit	Yes	Yes
MD5/SHA-1	Yes	Yes
Modular Exponentiation, 1024-bit	Yes	Yes
Receive Filter Options	Accept or reject packets with CRC match/mismatch, runt error collect or reject, Unicast, Not-Me Unicast, Multicast, Broadcast, Magic Packet™, Pattern Table and Hash Table	
Packages	44-Pin TQFP, QFN	64-Pin TQFP

جدول ۶-۱

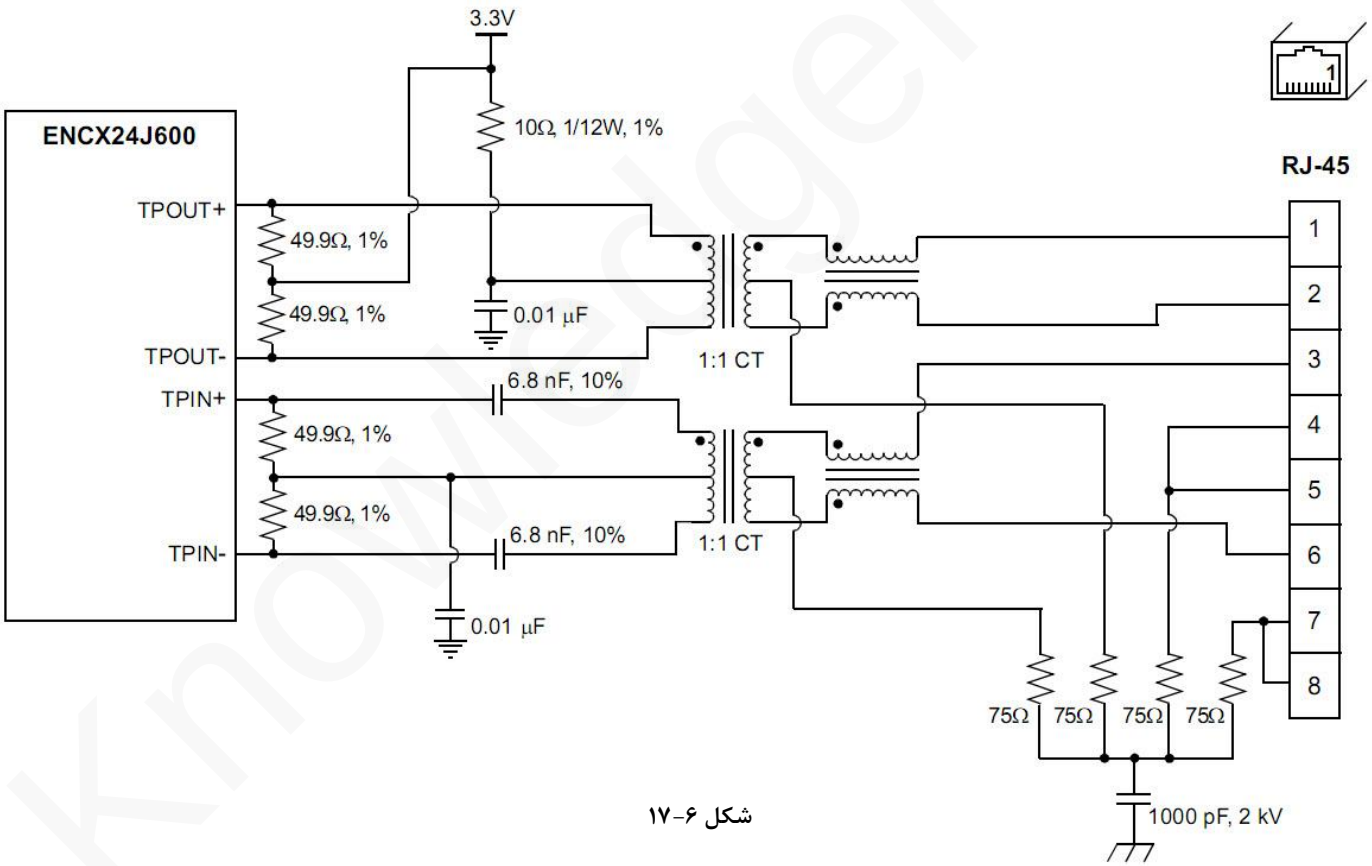
پایه های OSC1 و OSC2 برای اتصال کریستال 25MHZ استفاده می شوند.

پایه های TPIN+ و TPIN- به همراه پایه های TPOUT+ و TPOUT- برای اتصال به شبکه به کار می روند.

این تراشه دارای ۹ پایه برای تغذیه است که شامل تغذیه 3.3v و زمین مرجع برای قسمت های مختلف تراشه مانند بخش فرستنده و گیرنده لایه فیزیکی و واحد PLL می شود.

پایه های CS (پایه شماره ۳۴ در بسته بندی ۴۴ پایه و پایه شماره ۴۹ در بسته بندی ۶۴ پایه)، SCK ، SI و SO برای اتصال تراشه توسط رابط SPI به میکروکنترلر به کار می رود. همچنین توسط پایه SPISEL می توان رابط مورد استفاده را (SPI یا PSP) انتخاب کرد.

شکل زیر اتصال تراشه را به شبکه از طریق مدارات جانبی نشان می دهد.



شکل ۶-۱۷

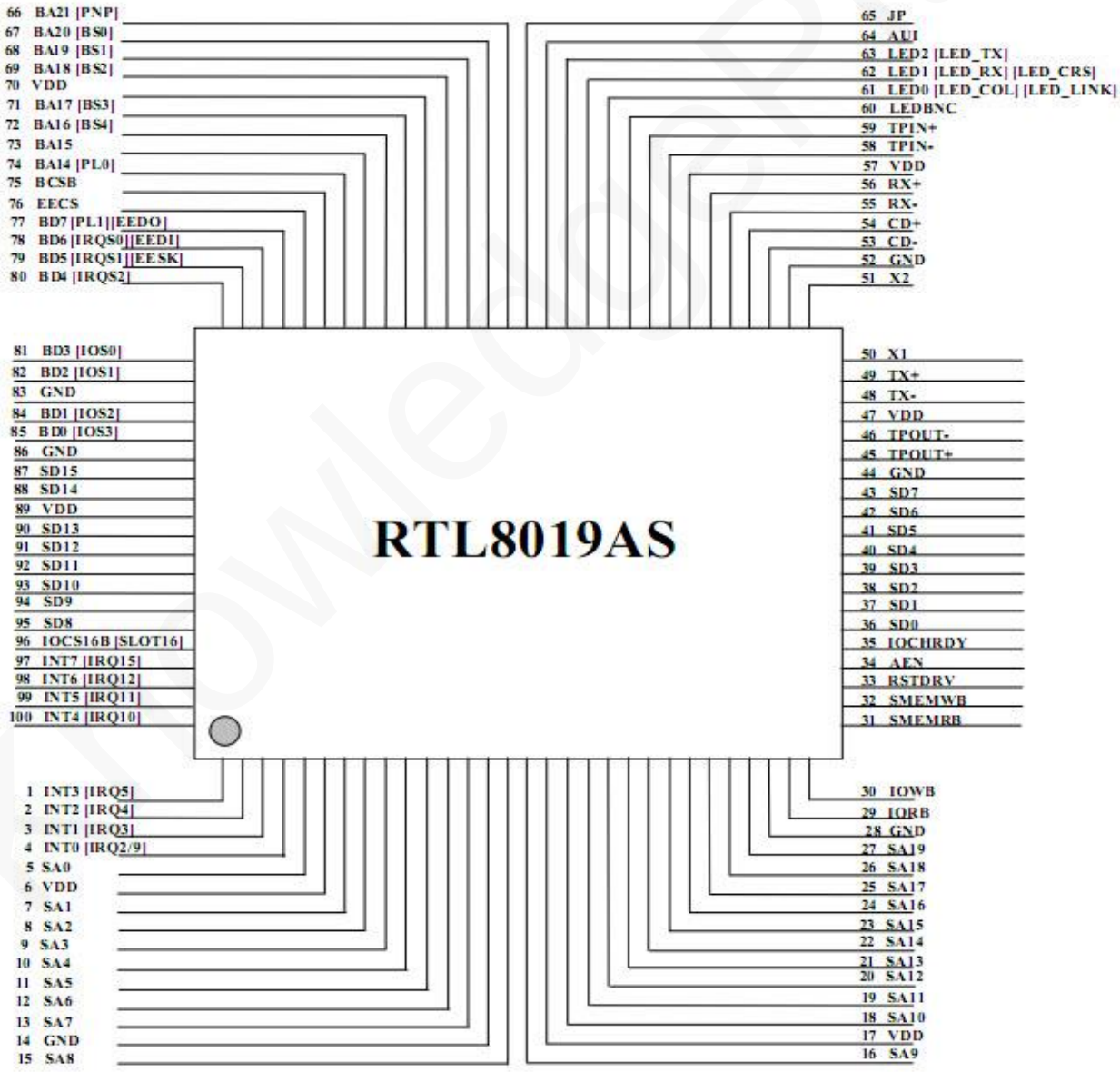
٦-٦-٣. تراشه RTL8019AS

اصلی ترین کاربرد این تراشه که از محصولات قدیکی شرکت RealTek می باشد، در طراحی و ساخت کارت شبکه می باشد. این تراشه قابلیت های فراوانی دارد و از جمله مشخصات آن می توان به موارد زیر اشاره کرد :

- سازگار با استاندارد IEEE 802.3 (سرعت انتقال داده 10Mbit/s)، 10Base5,10Base2,10BaseT
- دارا بودن ۸ وقفه
- ۱۶ کیلوبایت حافظه داخلی
- اتصال استاندارد به میکروپروسسور (قرارگیری مستقیم روی باس داده و آدرس مانند RAM)
- تشخیص خودکار نوع بسته داده (Data Pocket)
- پشتیبانی مستقل از حافظه دائم
- استفاده از EEPROM برای ذخیره سازی تنظیمات
- بسته بندی ۱۰۰ پایه PQFP

به طور کلی می توان گفت به علت اینکه شیوه دسترسی به حافظه تراشه همانند RAM می باشد، ارتباط مدار اصلی با این تراشه می تواند بسیار سریع باشد اما به دلیل داشتن تعداد زیاد پایه و مشکلات طراحی و مونتاژ، معمولا برای دستگاه های کوچک استفاده نمی شود و جز کارت شبکه، در دستگاه هایی با میکروپروسسور یا میکروکنترلرهایی با پایه های متعدد به کار می رود. کنترل این تراشه در مواردی که با میکروکنترلرهای دارای پایه کم همراه می شود، به صورت آدرس دهی غیرمستقیم است که باعث کاهش سرعت می شود.

ترتیب پایه ها و بسته بندی این تراشه در شکل های زیر نشان داده شده است.

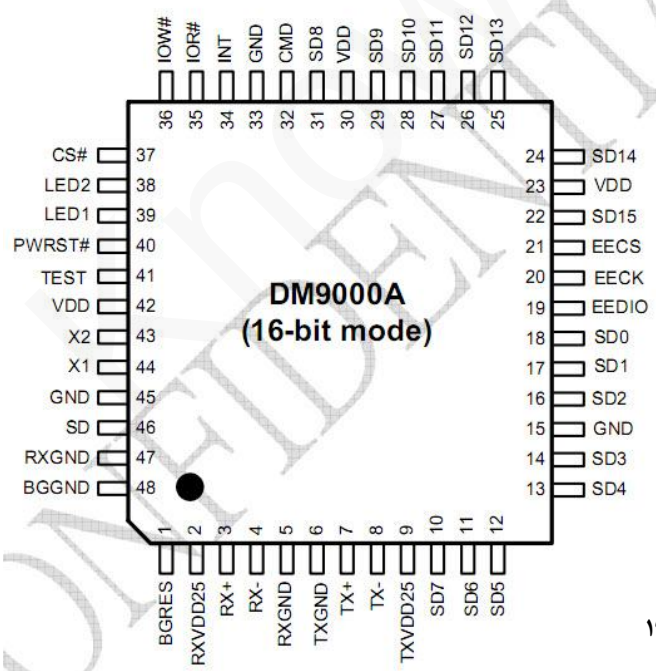


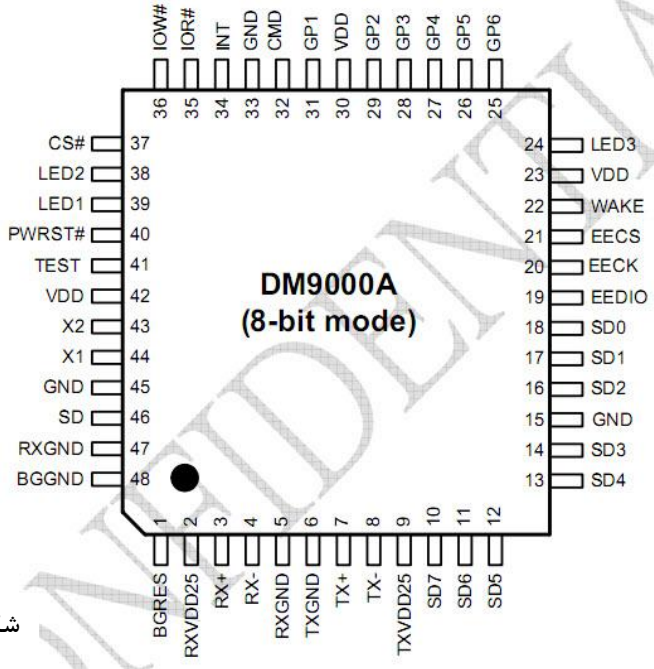
شکل ۶-۱۸

٦-٦-٤. تراشه DM9000AEP

- پشتیبانی از سرعت های 10Mbit/s و 100Mbit/s (10Base-T و 100Base-TX)
- تغذیه 3.3v
- قابلیت تحمل ولتاژ 5v بر روی پایه های I/O
- قابلیت دسترسی ۸ بیتی و ۱۶ بیتی به حافظه داخلی برای میکروکنترلر
- پشتیبانی از AUTO-MDIX
- تطبیق با استاندارد IEEE 802.3u
- پشتیبانی از عملیات checksum

ترتیب پایه ها و بسته بندی این تراشه در شکل های زیر نشان داده شده است.





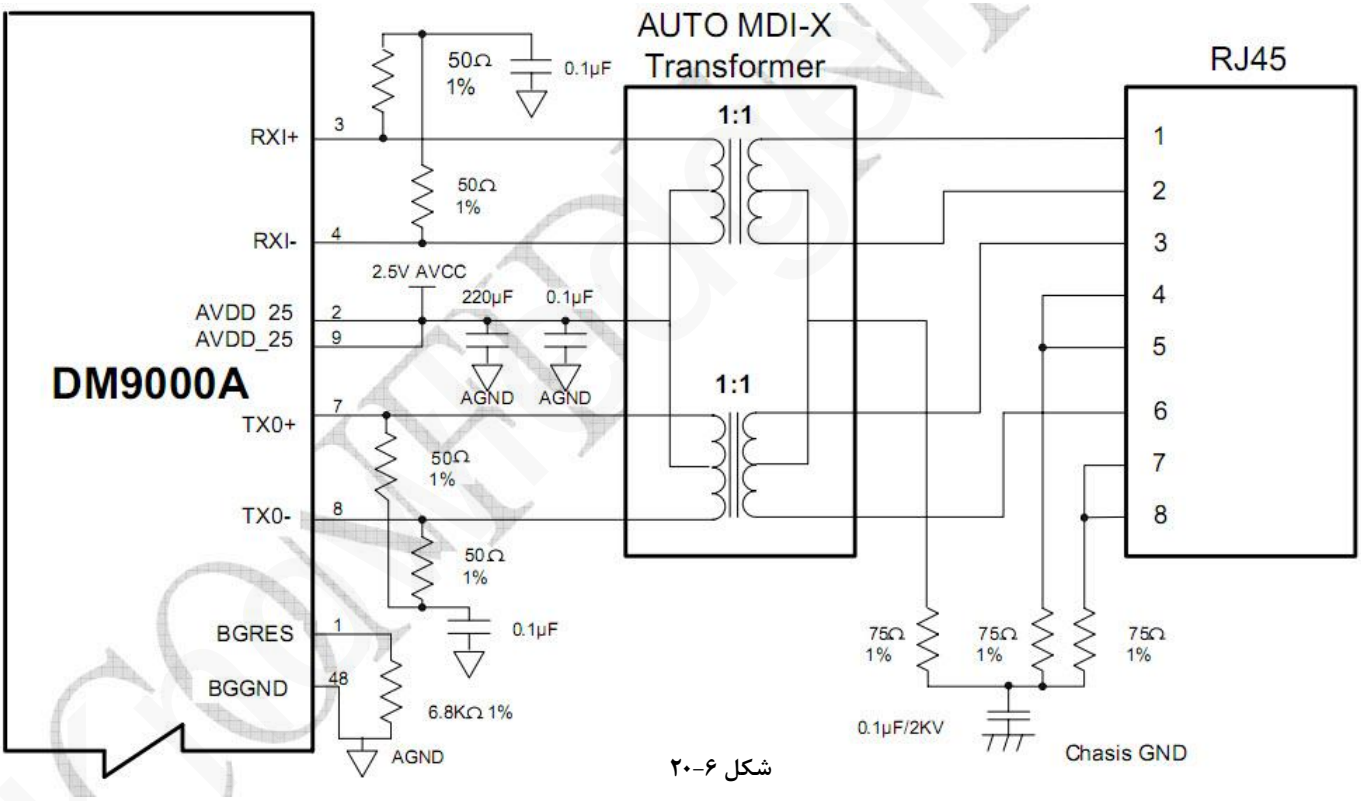
شکل ۶-۱۹

پایه های SD0 تا SD15 یک باس داده ۱۶ بیتی یا دو بایتی را تشکیل می دهند که برای ارسال و دریافت داده به سمت میکروپروسسور یا میکروکنترلر مورد استفاده قرار می گیرند. پایه های دیگر نیز مانند IOR# (دستور خواندن) یا IOW# (دستور نوشتن) در کنار این ۱۶ پایه برای ارتباط با میکروکنترلر به کار می روند.

پایه های X1 و X2 برای اتصال کریستال 25MHZ استفاده می شوند.

پایه های RX+ و RX- نیز به همراه پایه های TX+ و TX- برای اتصال به کابل شبکه به کار می روند.

شکل زیر استفاده از این تراشه را به همراه مدارات جانبی برای اتصال به کانکتور RJ-45 را برای هر دو سرعت 10Mbit/s و 100Mbit/s نشان می دهد.

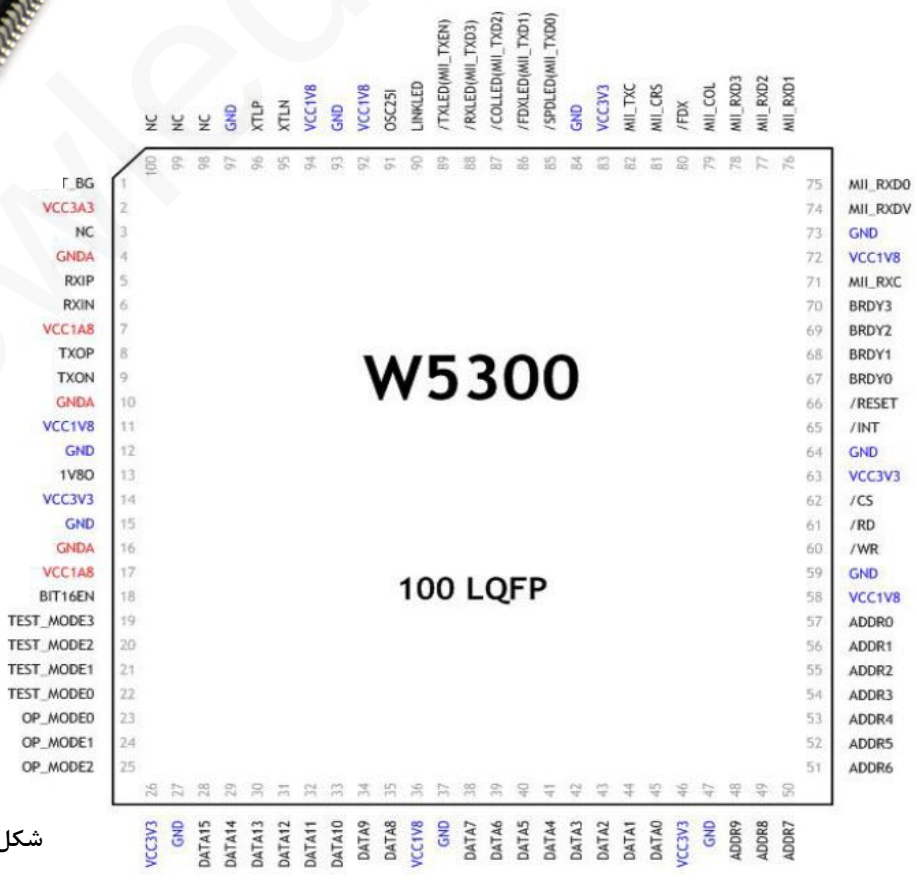


شکل ۶-۲۰

٦-٦-٥. تراشه W5300

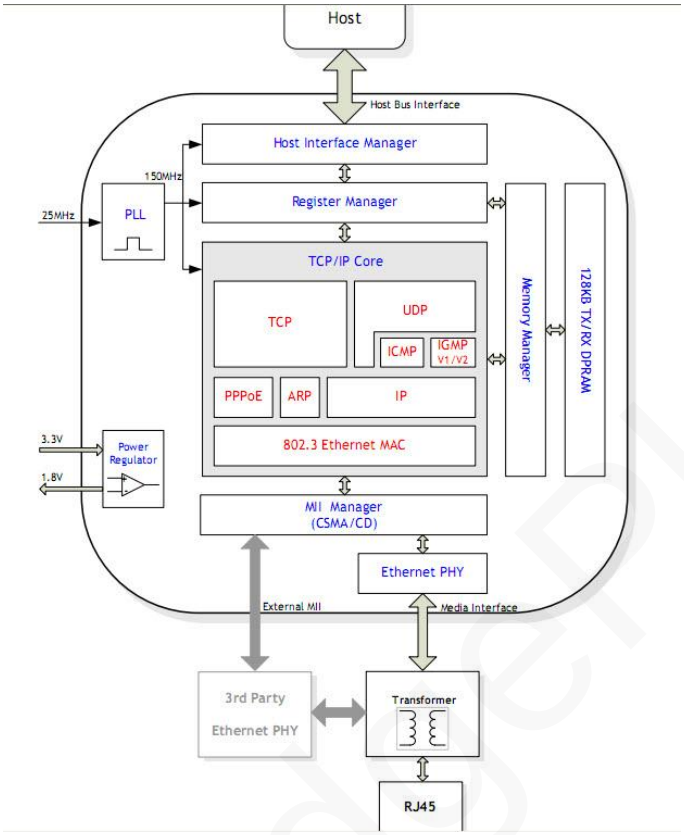
- پشتیبانی از سرعت های 10Mbit/s و 100Mbit/s (10Base-T ، 100Base-TX)
- پشتیبانی از AUTO-MDIX
- تغذیه 3.3v
- قابلیت تحمل ولتاژ 5v در پایه های I/O
- مناسب برای کاربردهای embedded internet مانند IP-TV ، IP-STB و DTV
- استفاده از تکنولوژی CMOS و توان مصرفی بسیار پایین
- محافظت در برابر ESD
- بسته بندی QFP ۱۰۰ پایه

ترتیب پایه ها و بسته بندی این تراشه در شکل های زیر نشان داده شده است.



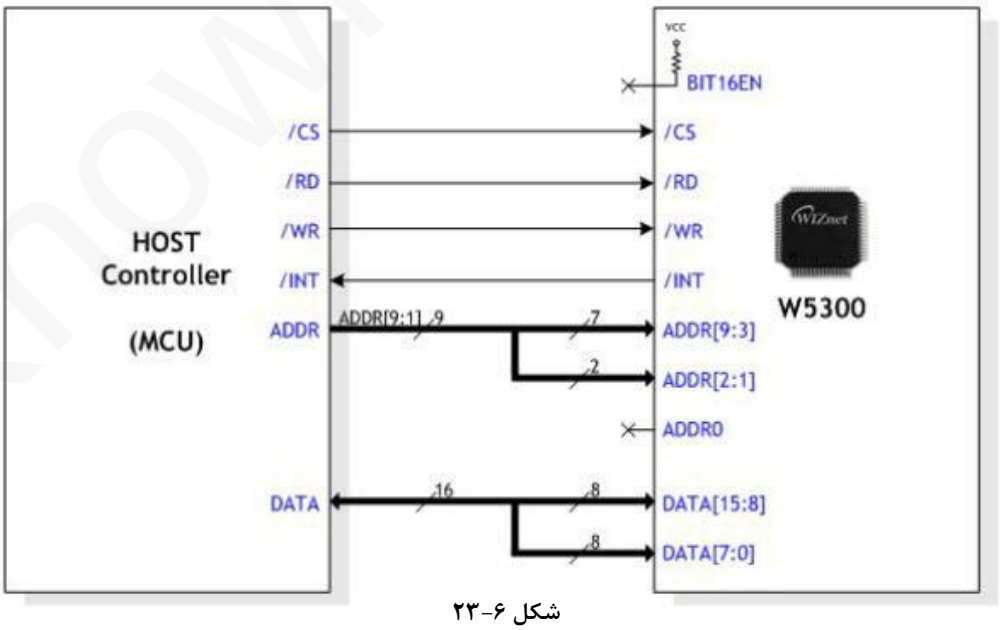
شکل ۶-۲۱

شکل زیر بلوک دیاگرام داخلی این تراشه را نشان می دهد. مطابق شکل، یکی از ویژگی های این تراشه پیاده سازی پروتکل هایی مانند TCP ، UDP، PPOE ، ARP و IP به صورت سخت افزاری می باشد.



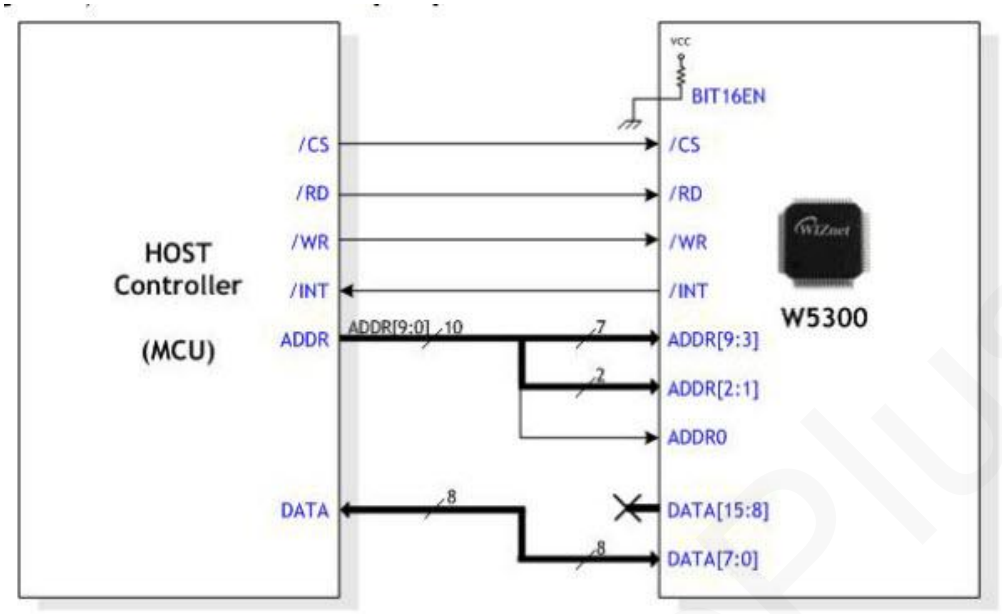
شکل ۶-۲۲

شکل زیر اتصال این تراشه را توسط باس ۱۶ بیتی به میکروکنترلر را نشان می دهد.



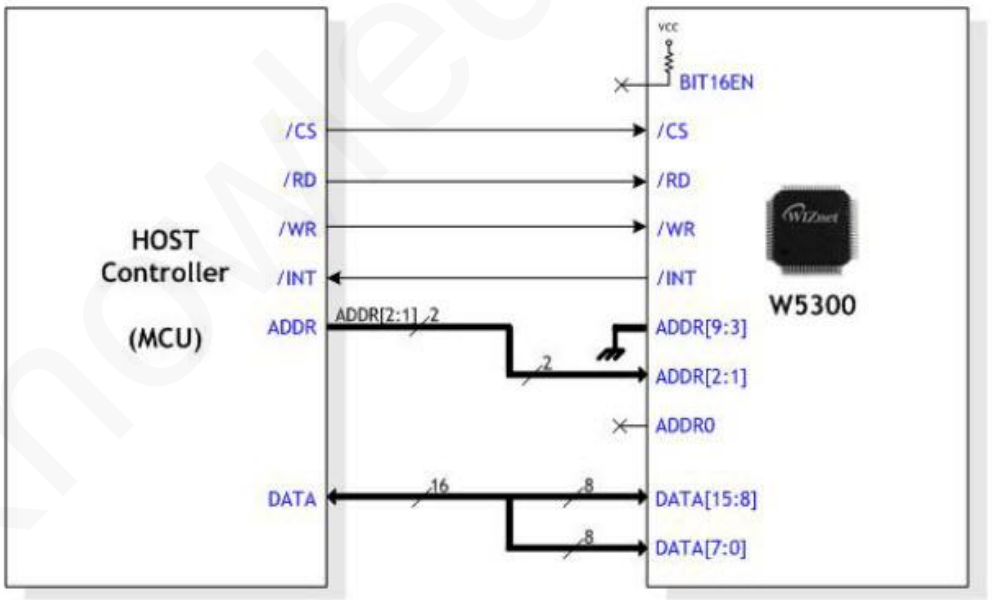
شکل ۶-۲۳

شکل زیر اتصال این تراشه را توسط باس ۸ بیتی به میکروکنترلر را نشان می دهد.



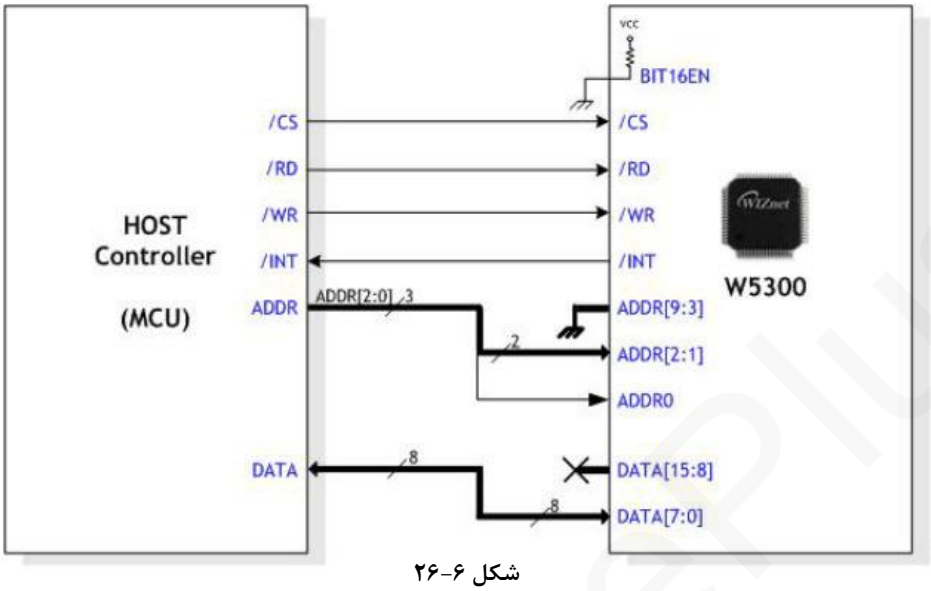
شکل ۶-۲۴

شکل زیر اتصال این تراشه را توسط باس ۱۶ بیتی برای آدرس دهی غیرمستقیم حافظه به میکروکنترلر را نشان می دهد.



شکل ۶-۲۵

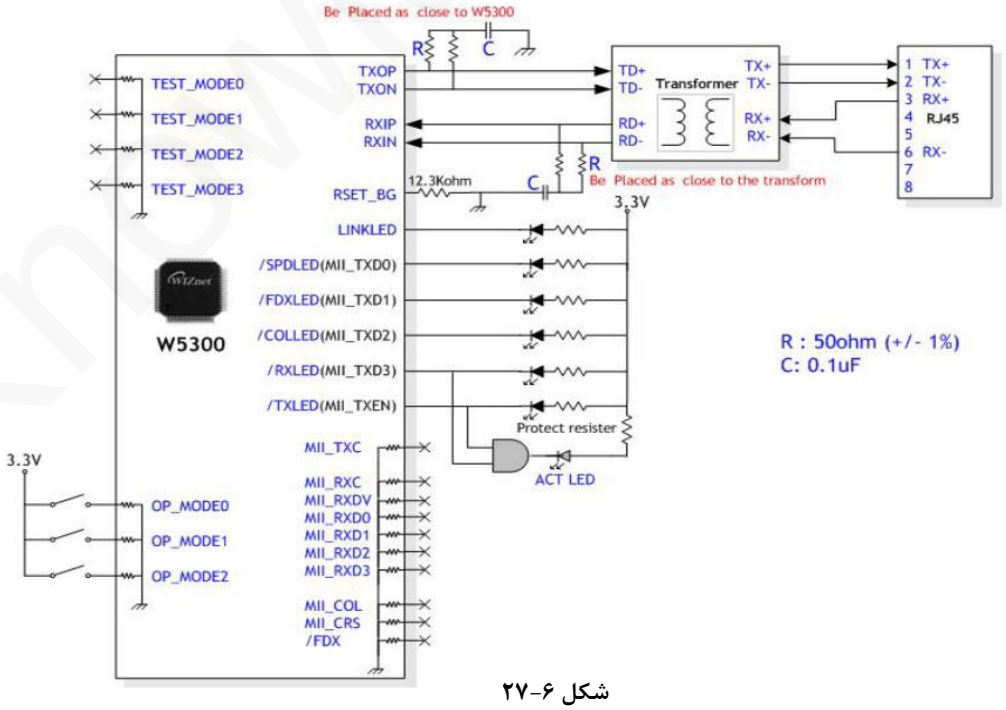
شکل زیر اتصال این تراشه را توسط باس ۸ بیتی برای آدرس دهی غیرمستقیم حافظه به میکروکنترلر را نشان می دهد.



شکل ۶-۲۶

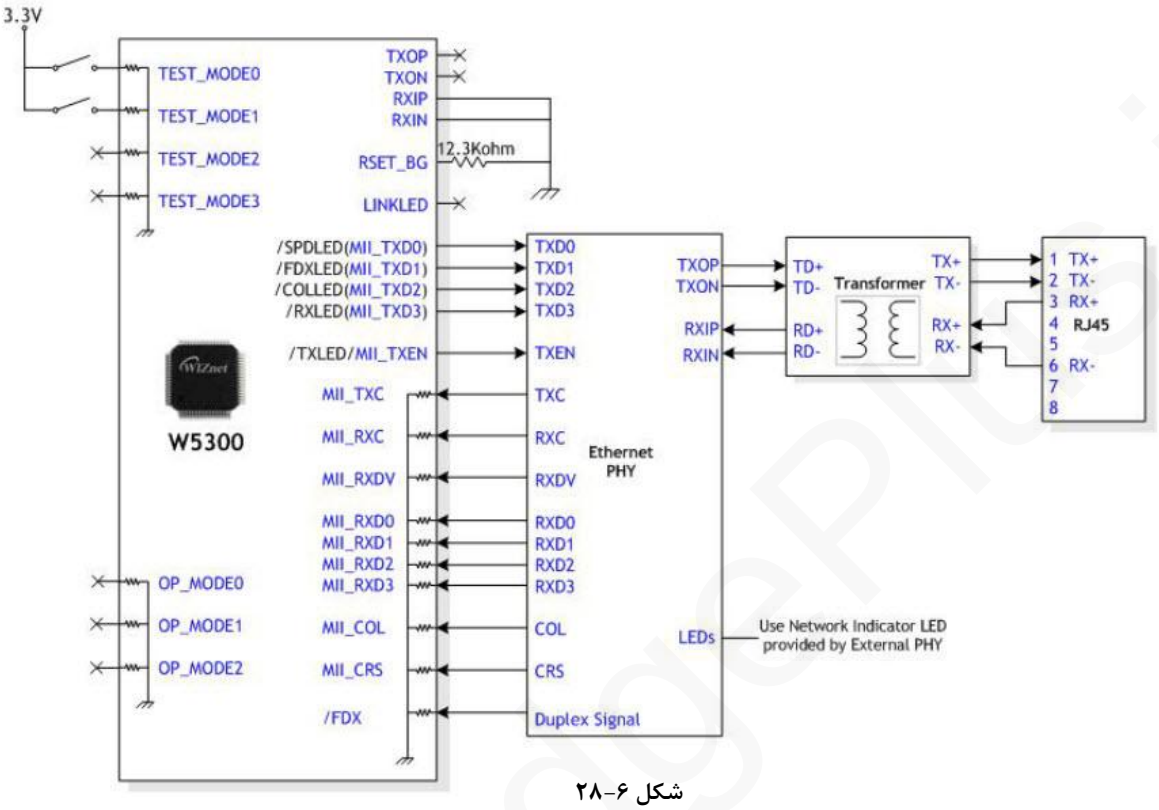
یکی از ویژگی های این تراشه این است که علاوه بر این که در خود تراشه لایه ی فیزیکی پیاده سازی شده است، این امکان وجود دارد که برای پیاده سازی لایه ی فیزیکی از تراشه های transceiver جداگانه نیز استفاده نمود.

شکل زیر اتصال تراشه را به شبکه توسط لایه ی فیزیکی داخلی نشان می دهد.



شکل ۶-۲۷

شکل زیر اتصال تراشه را به شبکه توسط تراشه transceiver برای پیاده سازی لایه ی فیزیکی نشان می دهد.



شکل ۶-۲۸

در انتهای این بخش، با توجه به استفاده از تراشه ENC28J60 شرکت Mircochip در این مقاله و با توجه به مشابهت کار تراشه های مختلف این شرکت، لیست زیر دیگر تراشه های این شرکت را برای پیاده سازی لایه پیوند داده و لایه فیزیکی نشان می دهد که می توان از آنها به عنوان جایگزین تراشه ENC28J60 استفاده نمود.

Ethernet Controllers						
ENC28J60	10Base-T Ethernet Controller	SPI	–	–	✓	28-pin SPDIP, SSOP, SOIC, QFN
ENC624J600	10Base-T/100Base-TX Ethernet Controller with Security	SPI/Parallel	–	–	✓	24-pin TQFN, QFN, 64-pin TQFN
LAN9217	10Base-T/100Base-TX Ethernet Controller with 16-bit/MII interface	16-bit Host Bus/MII	–	–	–	100-pin TQFP
LAN9218	10Base-T/100Base-TX Ethernet Controller with 32-bit interface	32-bit Host Bus	–	–	✓	100-pin TQFP
LAN9220	10Base-T/100Base-TX Ethernet Controller with 16-bit interface	16-bit Host Bus	–	–	–	56-pin QFN
LAN9221	10Base-T/100Base-TX Ethernet Controller with 16-bit interface	16-bit Host Bus	–	–	✓	56-pin QFN
LAN9420	10Base-T/100Base-TX Ethernet Controller with 32-bit PCI interface	32-bit PCI 3.0	–	–	✓	128-pin VTQFP
LAN89218	TrueAuto, 10Base-T/100Base-TX Ethernet Controller with 32-bit interface	32-bit Host Bus	–	–	Automotive	100-pin TQFP
KSZ8851	10/100Base-TX Ethernet Controller	8-/16-/32-bit or SPI	✓	–	Automotive	32-pin QFN, 48-pin LQFP, 128-pin PQFP
KSZ8852	2-Port 10/100Base-TX Ethernet Controller	8-/16-/32-bit	✓	✓	✓	64-pin LQFP
KSZ8441	10/100Base-TX/FX Ethernet Controller with 1588v2 PTP and Clock Synchronization	8-/16-/32-bit or PCI	✓	✓	✓	64-pin LQFP

*Note: All products above are supported with 3.3V operating voltage

جدول ۶-۲

٦-٧. معرفی تراشه های مورد نیاز برای حالت پیاده سازی چهارم

در این بخش یک تراشه Ethernet Controller که دارای بخش transceiver نیست برای حالت چهارم پیاده سازی معرفی می شود.

٦-٧-١. تراشه W3100A

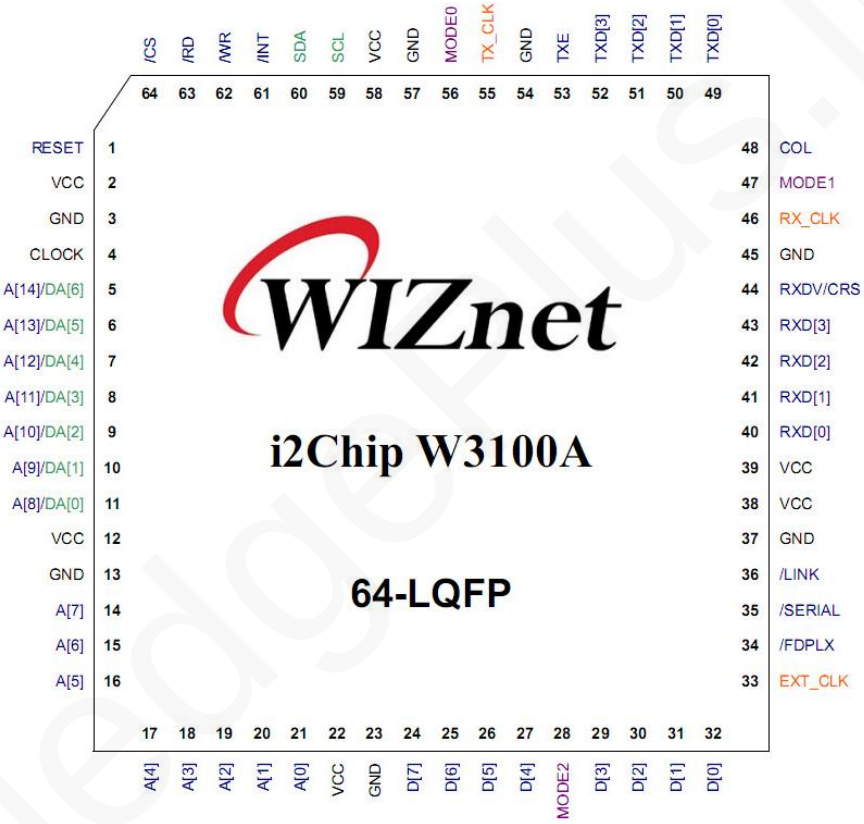
این تراشه محصول شرکت WIZnet است و به عنوان یک کنترل کننده ی اترنت قابلیت های فراوانی دارد و علاوه بر پیاده سازی لایه پیونده داده، لایه های TCP و UDP را نیز به صورت سخت افزاری پیاده سازی می کند و همین امر توان عملیاتی و سرعت خالص سیستم را افزایش می دهد.

مشخصات این تراشه عبارت اند از :

- سازگاری با استاندارد MII برای ارتباط با تراشه لایه ی فیزیکی
- پیاده سازی سخت افزاری پروتکل های TCP و UDP
- ۱۶ کیلوبایت حافظه داخلی
- اتصال استاندارد به میکروپروسسور (قرارگیری مستقیم روی باس داده و آدرس مانند RAM)
- ارتباط I2C
- بسته بندی ۶۴ پایه LQFP

این تراشه در صورت استفاده از شیوه اتصال استاندارد، می تواند با توجه به پیاده سازی داخلی لایه های حامل به سرعت واقعی 20Mbit/s و در صورت استفاده از تراشه لایه ی فیزیکی به سرعت 100Mbit/s برسد.

ترتیب پایه ها و بسته بندی این تراشه در شکل زیر نشان داده شده است.



شکل ۶-۲۹

٦-٨. تراشه های Transceiver اترنت

در این بخش تراشه های transceiver مورد نیاز برای پیاده سازی لایه فیزیکی (حالت های پیاده سازی دوم و چهارم) معرفی می شوند.

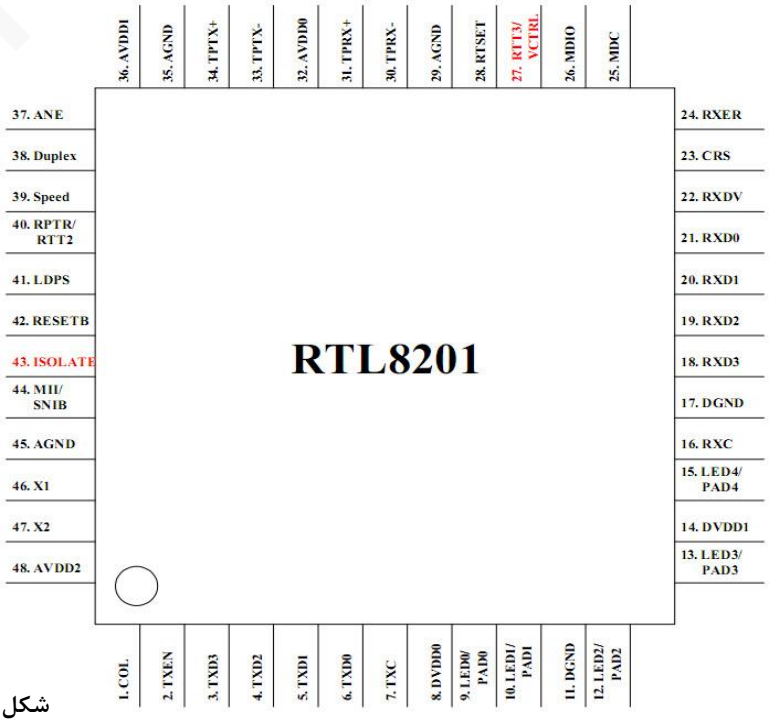
٦-٨-١. تراشه RTL8201

این تراشه transceiver محصول شرکت RealTek است و برای پیاده سازی لایه فیزیکی طراحی شده است.

مشخصات این تراشه عبارت اند از :

- سازگاری با استاندارد MII برای ارتباط با تراشه لایه پیوند داده
- پشتیبانی از سرعت های 10Mbit/s و 100Mbit/s
- بسته بندی ۴۸ پایه LQFP

ترتیب پایه ها و بسته بندی این تراشه در شکل زیر نشان داده شده است.

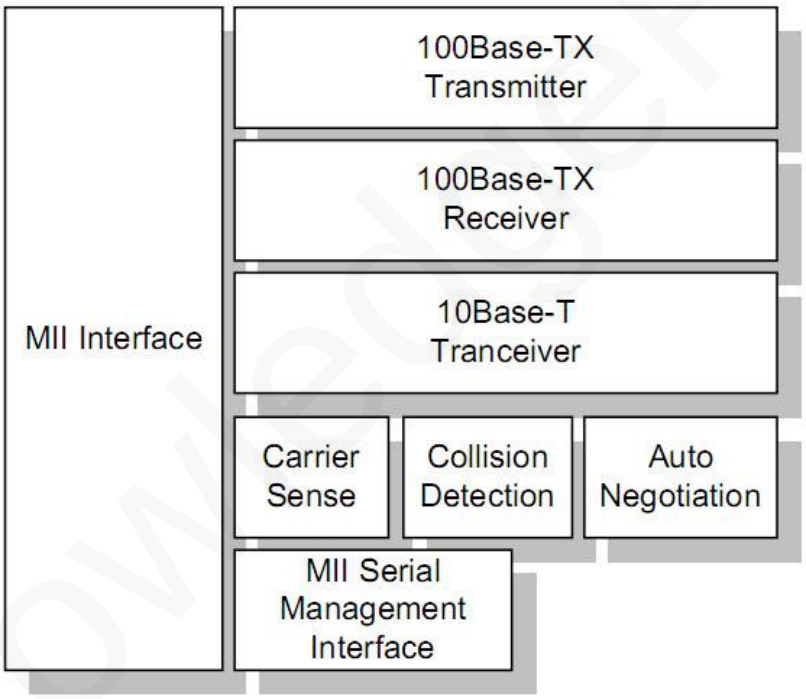


شکل ۶-۳۰

٦-٨-٢. تراشه DM916

- مطابق استانداردهای 100BASE-TX و 10Base-T
- طبق استاندارد IEEE802.3u برای 100BASE-TX شامل تمام استانداردهای لایه فیزیکی
- پشتیبانی از auto-negotiation برای تشخیص نوع سرعت و نوع ارتباط
- شامل فیلترهای جانبی، بدون نیاز به فیلترهای خارجی برای اتصال به شبکه

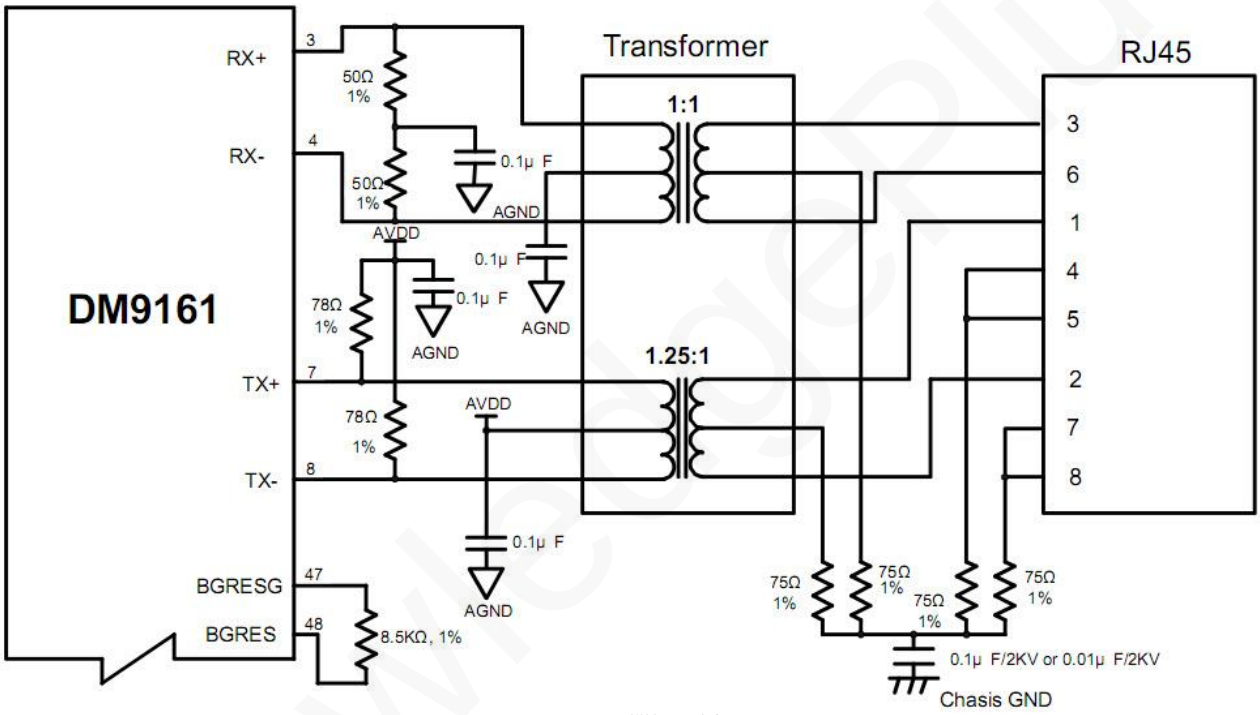
شکل زیر بلوک دیاگرام این تراشه را نشان می دهد.



شکل ۶-۳۱

پایه های ۳ و ۴ تراشه به نام های RX+ و RX- و پایه های ۷ و ۸ به نام های TX+ و TX- برای ارسال و دریافت سیگنال ها به شبکه به کار می رود.

شکل زیر اتصال این تراشه را از طریق ترانسفورماتور به کانکتور RJ-45 نشان می دهد. مطابق شکل، برای پایه های فرستنده از دو مقاومت ۷۸ اهم برای فراهم کردن بایاس DC واحد فرستنده تراشه استفاده شده است و سر وسط ترانسفورماتور این بخش مستقیما به تغذیه آنالوگ تراشه متصل شده و از طریق یک خازن با ظرفیت 0.1uF به زمین آنالوگ تغذیه متصل شده است. در بخش گیرنده به دلیل تفاوت مدارات داخلی از مدار دیگری استفاد شده است، به این ترتیب که با استفاده از دو مقاومت ۵۰ اهمی و خازن متصل شده بین آنها، دو پایه گیرنده به یکدیگر متصل شده اند و سر وسط ترانسفورماتور نیز توسط یک خازن 0.1uF به زمین متصل شده است.

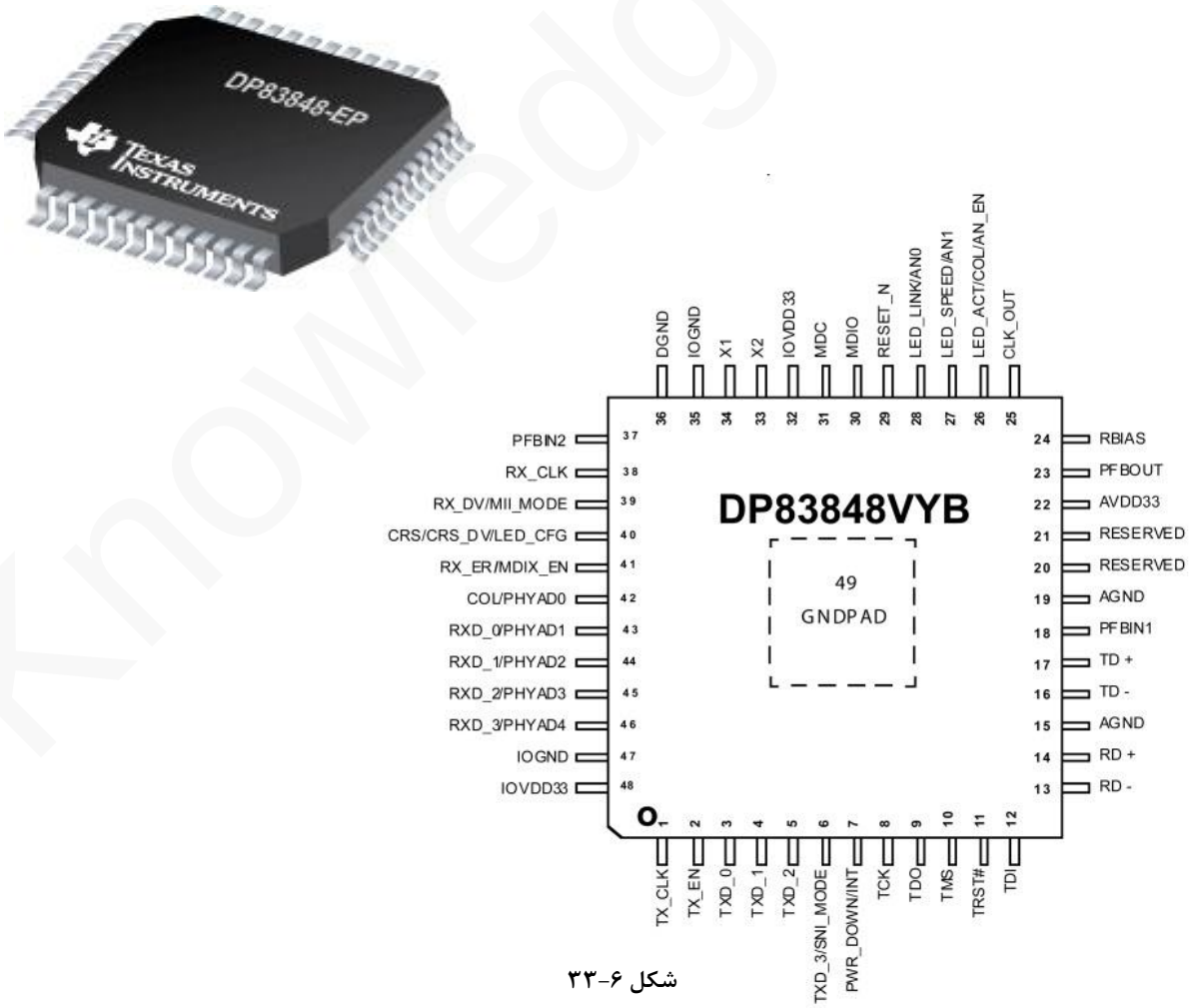


شکل ۶-۳۲

حدود ۲۱ پایه از این تراشه ۴۸ پایه، برای ارتباط با تراشه کنترل کننده اترنت مورد استفاده قرار می گیرند. پایه های دیگر مربوط به LED نشان گر وضعیت شبکه، اسیلاتور و تغذیه تراشه می باشند.

٦-٨-٣. تراشه DP83848

- پشتیبانی از سرعت های 10Mbit/s و 100Mbit/s (10Base-T ، 100Base-TX)
- دارای فیلترهای جانبی برای اتصال مستقیم به شبکه
- تطبیق با استانداردهای IEEE 802.3 ، IEEE 802.3u و IEEE 802.3ab
- پشتیبانی از AUTO-MDIX
- پشتیبانی از JTAG
- تغذیه 3.3v
- استفاده از تکنولوژی CMOS و توان مصرفی بسیار پایین
- محافظت در برابر ESD
- بسته بندی QFP



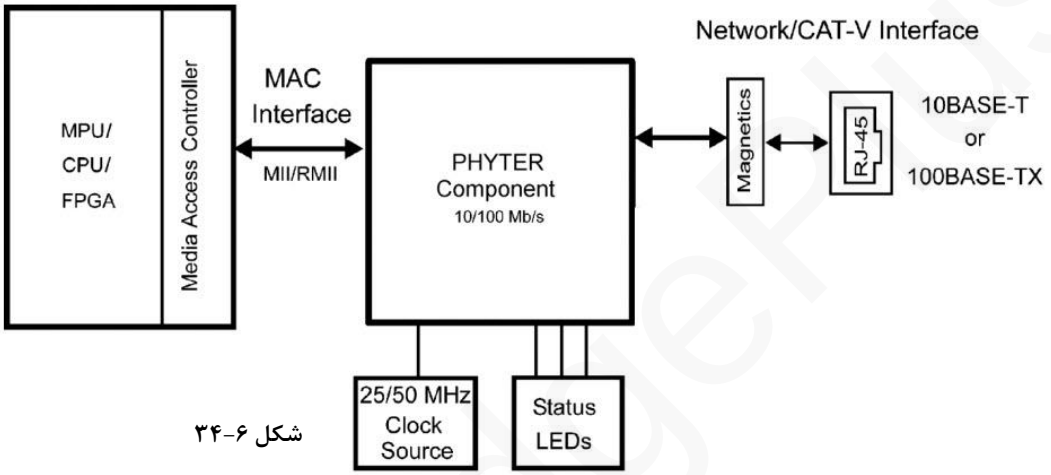
شکل ۶-۳۳

پایه های RD- و RD+ به همراه پایه های TD- و TD+ برای اتصال به شبکه به کار می روند.

پایه های TXD0 تا TXD4 و پایه های RXD0 تا RXD3 برای ارسال و دریافت داده به همراه پایه های جانبی دیگر برای اتصال به میکروپروسسور یا میکروکنترلر مورد استفاده قرار می گیرند.

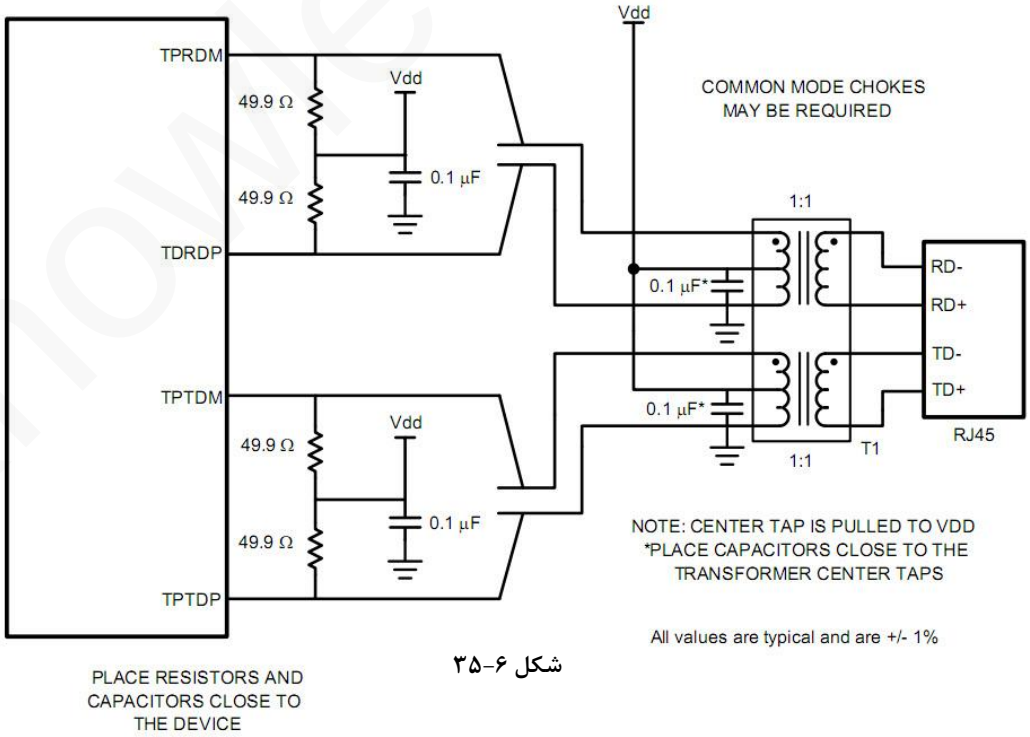
این تراشه دارای ۹ پایه برای تغذیه شامل تغذیه بخش آنالوگ، تغذیه بخش دیجیتال، تغذیه پایه های I/O و زمین آنالوگ و دیجیتال می باشد.

شکل زیر بلوک دیاگرام کلی برای ایجاد یک گره شبکه اترنت توسط این را نشان می دهد.



شکل ۶-۳۴

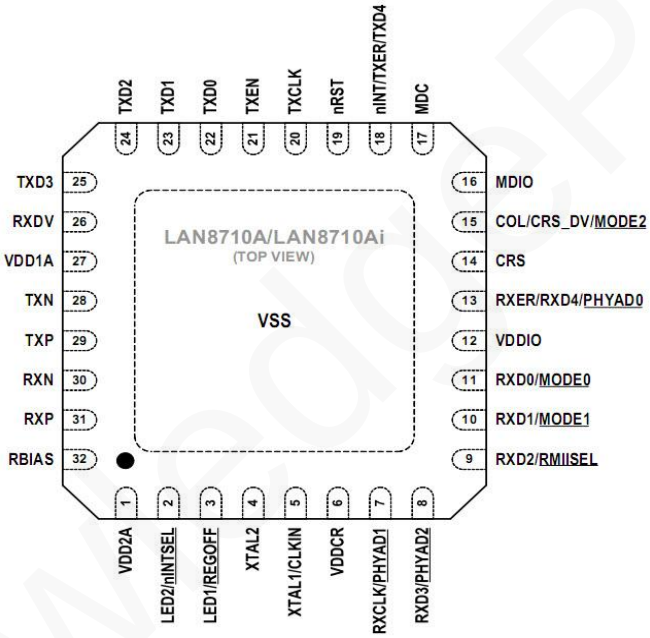
شکل زیر یک نمونه مدار کاربردی را برای این تراشه نشان می دهد.



شکل ۶-۳۵

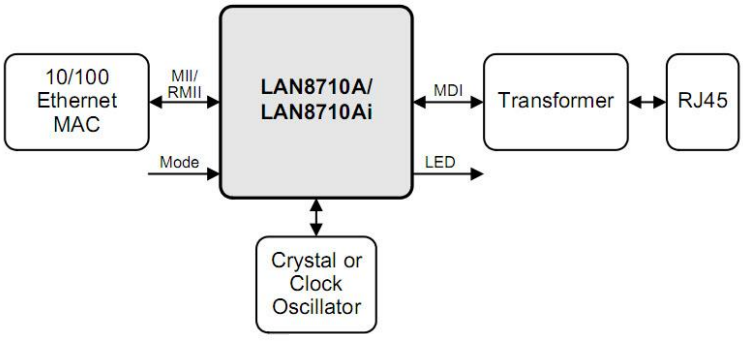
٦-٨-٤. تراشه LAN8710A

- پشتیبانی از سرعت های 10Mbit/s و 100Mbit/s (10Base-T ، 100Base-TX)
- تطبیق با استانداردهای IEEE 802.3 ، IEEE 802.3u ، ISO 802-3
- پشتیبانی از AUTO-MDIX ، Auto-negotiation
- ولتاژ تغذیه 3.3v ، رگولاتور 1.2v با قابلیت غیرفعال سازی
- قابلیت تحمل ولتاژ 3.6v بر روی پایه های I/O
- بسته بندی QFN ۳۲ پایه



شکل ۶-۳۶

شکل زیر استفاده از این تراشه را جهت طراحی گره اترنت نشان می دهد.



شکل ۶-۳۷

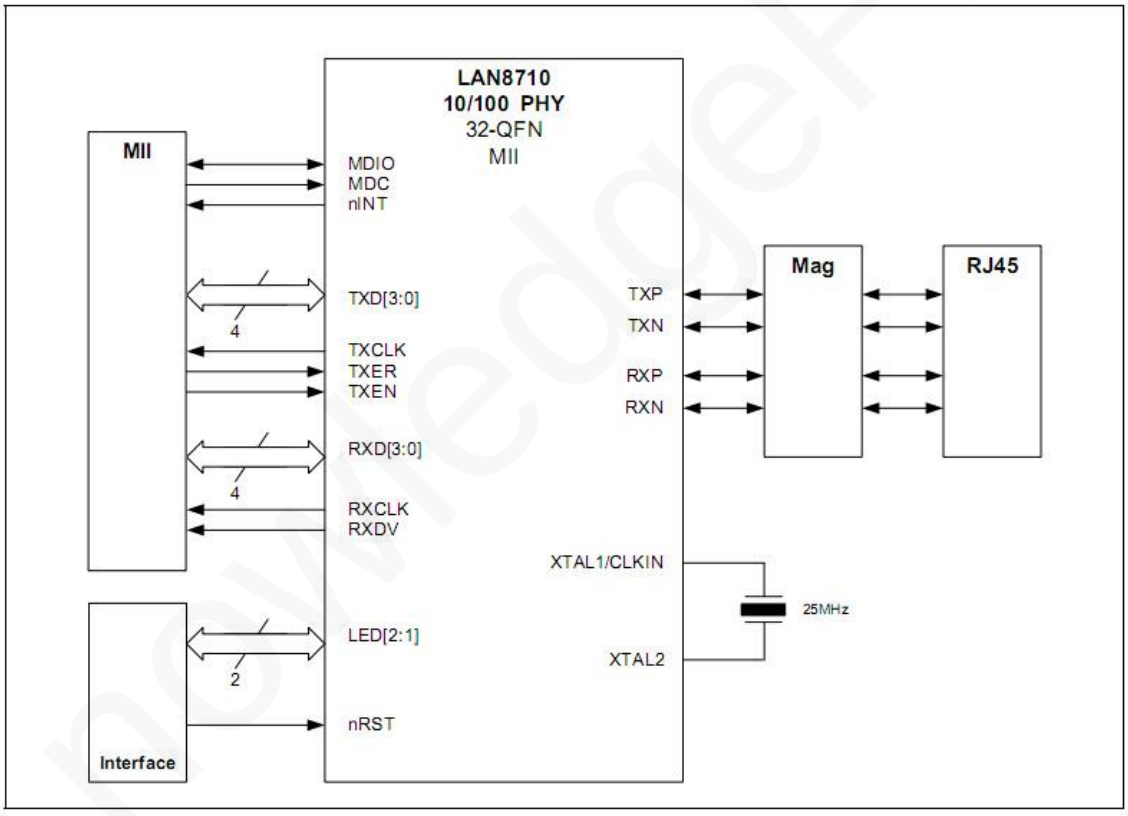
بلوک سمت چپ را که به منظور پیاده سازی لایه MAC استفاده می شود می توان در عمل توسط یک تراشه کنترل کننده اترنت و یا توسط یک میکروکنترلر پیاده سازی نمود.

پایه های RMII/MII شامل پایه هایی از تراشه است که با حروف TX و RX شروع می شوند (مانند TXD ، TXR و...) به همراه پایه های کنترلی دیگر مانند nINT برای وقفه

بلوک کریستال معمولا در اغلب تراشه ها از یک کریستال 25MHZ استفاده می شود.

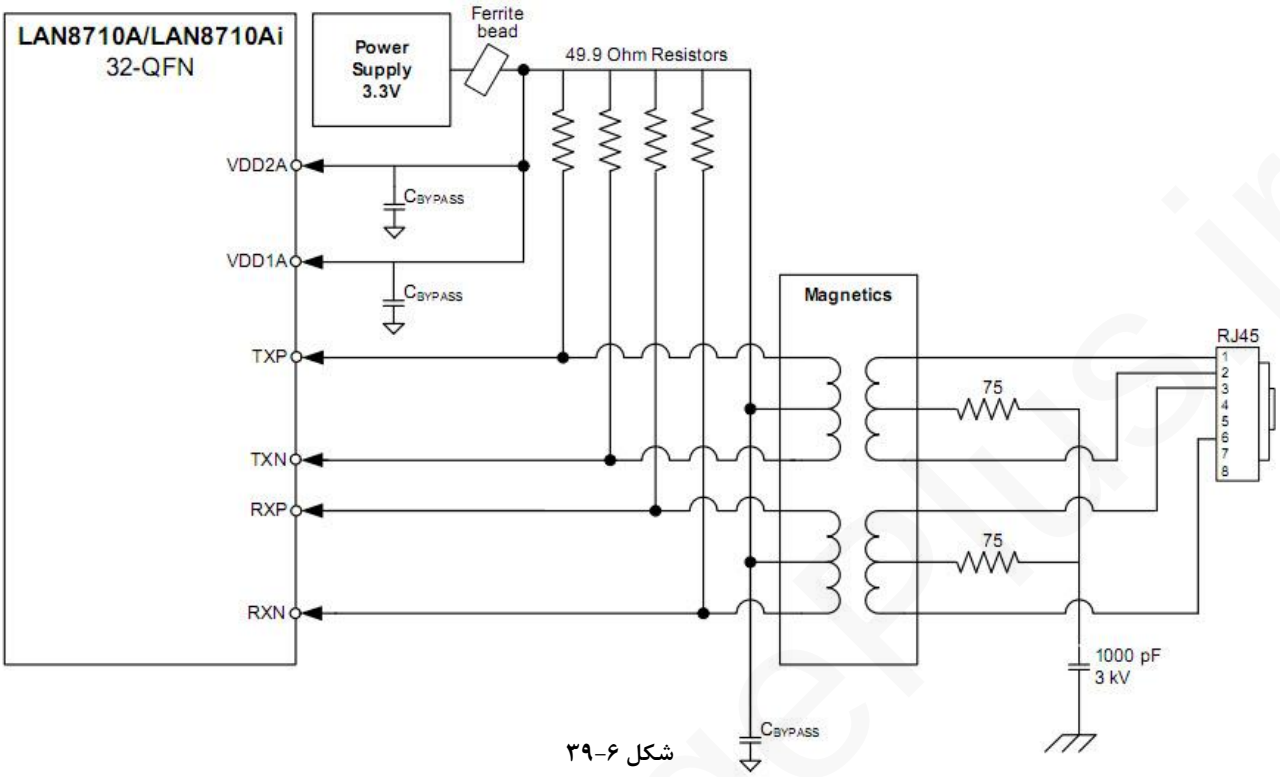
در سمت راست نیز در صورت عدم وجود مدارات ایزوله کننده و ترانسفورماتور در کانکتور RJ-45، باید به صورت خارجی این مدارات را بین تراشه transceiver و کانکتور RJ-45 استفاده نمود.

پایه های TXP و TXN به همراه پایه های RXP و RXN به عنوان پایه های MDI برای اتصال تراشه به کانکتور RJ-45 استفاده می شوند.

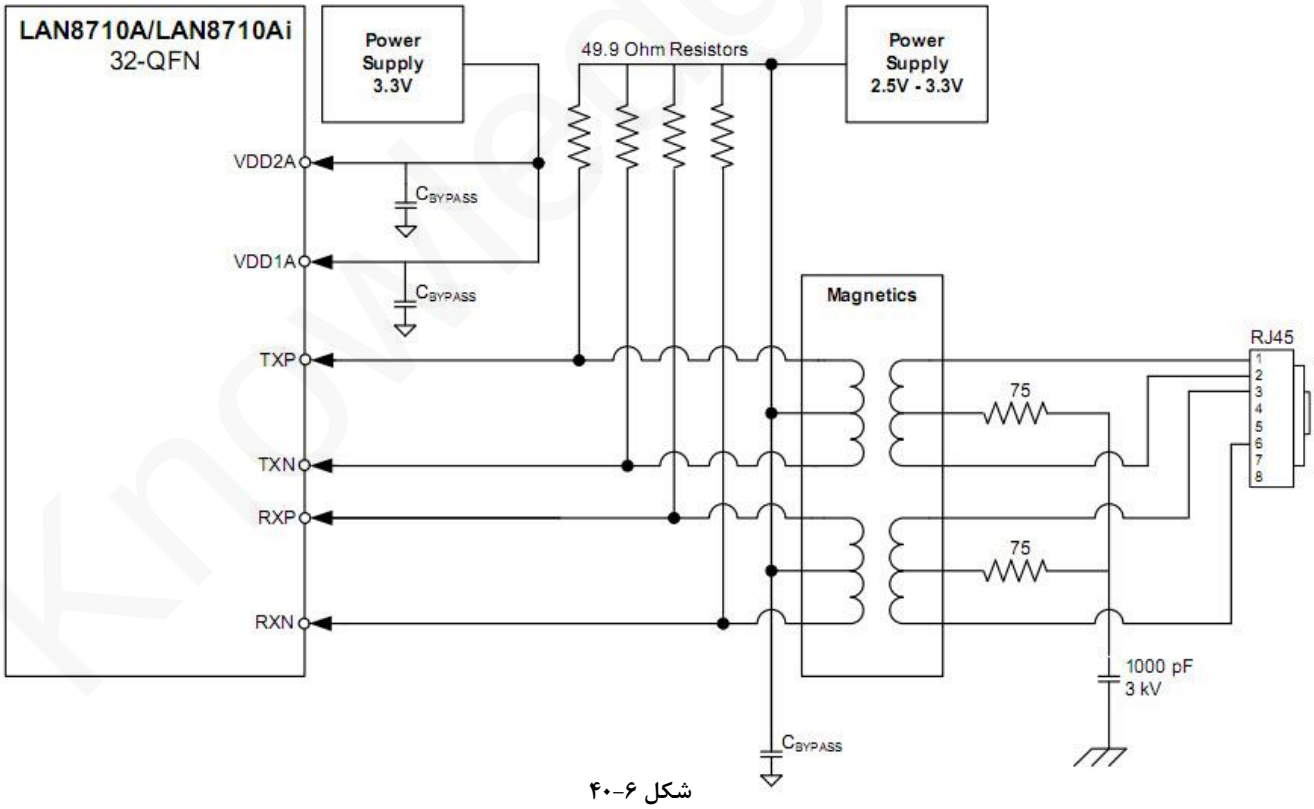


شکل ۶-۳۸

در شکل ۶-۳۹ مدار کاربردی این تراشه با استفاده از یک منبع تغذیه و شکل ۶-۴۰ مدار کاربردی این تراشه را به همراه استفاده از دو منبع تغذیه جداگانه نشان می دهد.



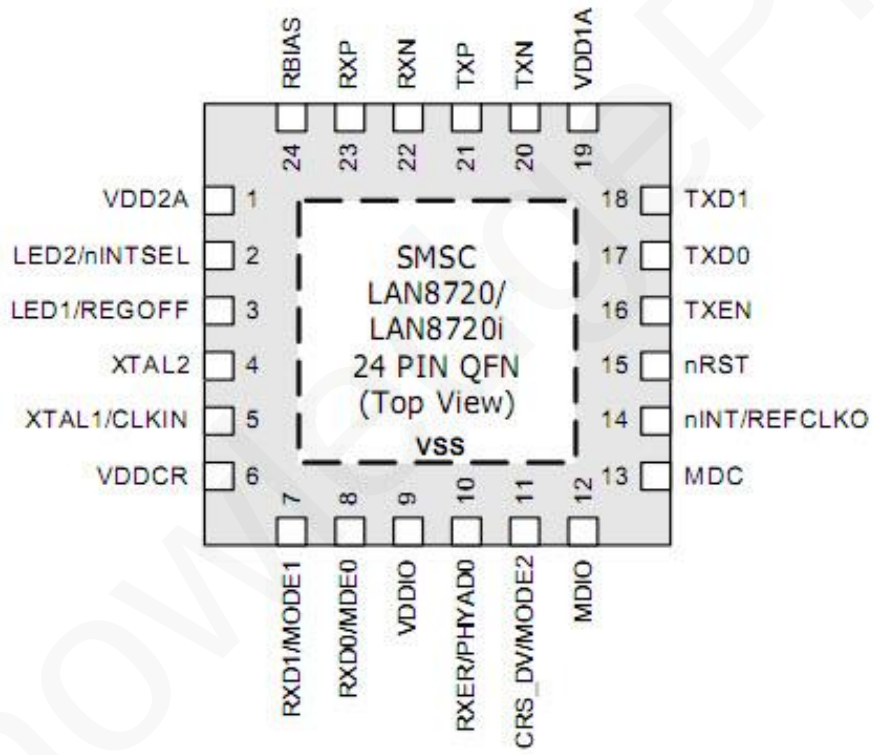
شکل ۶-۳۹



شکل ۶-۴۰

٦-٨-٥. تراشه LAN8720

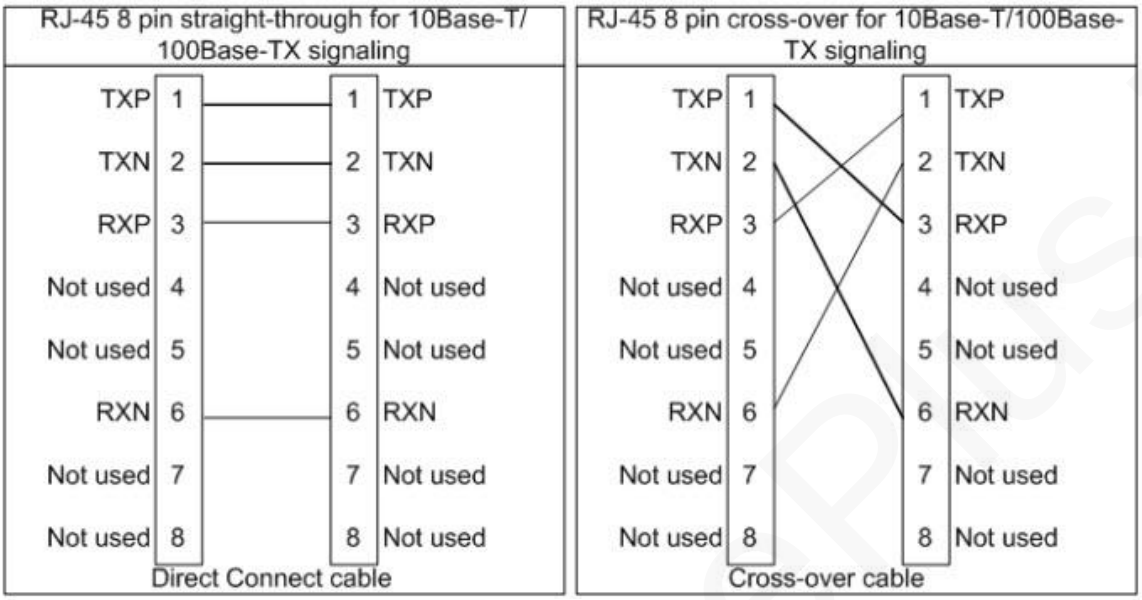
- پشتیبانی از سرعت های 10Mbit/s و 100Mbit/s (100Base-TX ، 10Base-T)
- تطبیق با استانداردهای IEEE 802.3 ، IEEE 802.3u
- پشتیبانی از AUTO-MDIX
- استفاده از تغذیه 3.3v با رگولاتور داخلی 1.2v یا ولتاژ تغذیه 1.2v بدون رگولاتور داخلی
- قابلیت تحمل ولتاژ 3.6v بر روی پایه های I/O
- بسته بندی QFP ۲۴ پایه



شکل ۶-۴۱

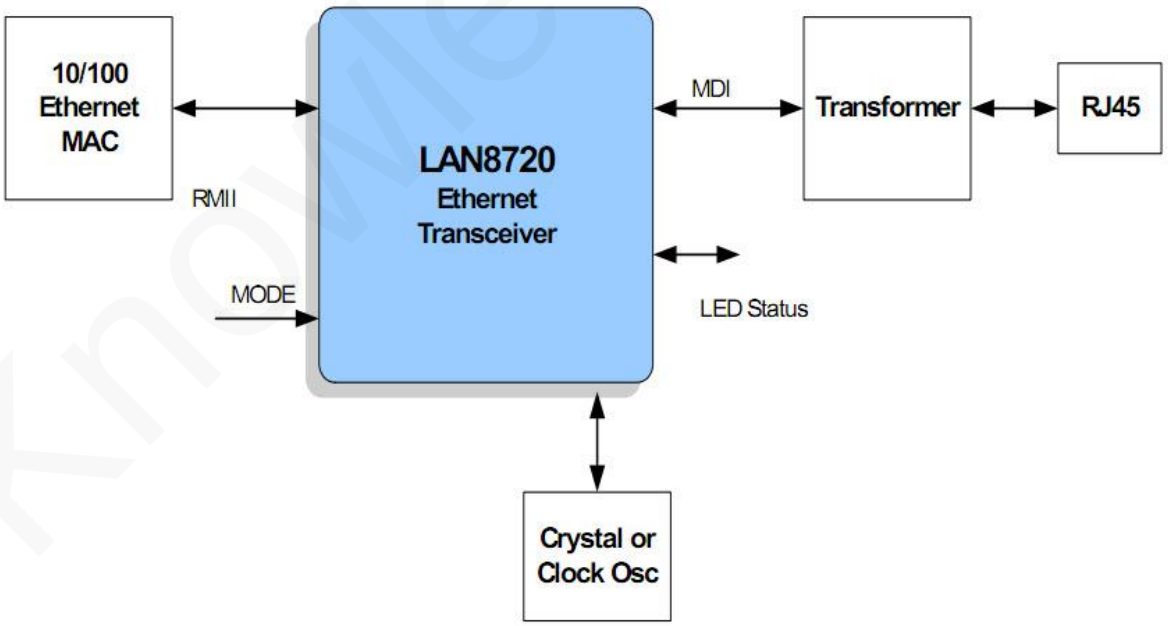
پایه های RXP و RXN به همراه پایه های TXP و TXN برای اتصال به کانکتور RJ-45 به کار می روند.

شکل زیر استفاده از دو کابل straight-through و cross-over را برای اتصال به تراشه نشان می دهد.



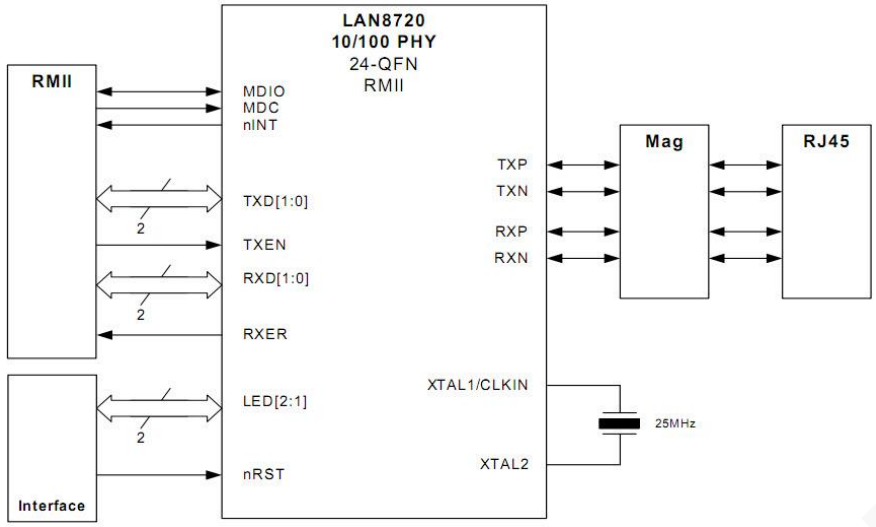
شکل ۶-۴۲

شکل های ۶-۴۳، ۶-۴۴ و ۶-۴۵ اتصال تراشه را به میکروکنترلر و کانکتور RJ-45 نشان می دهند.

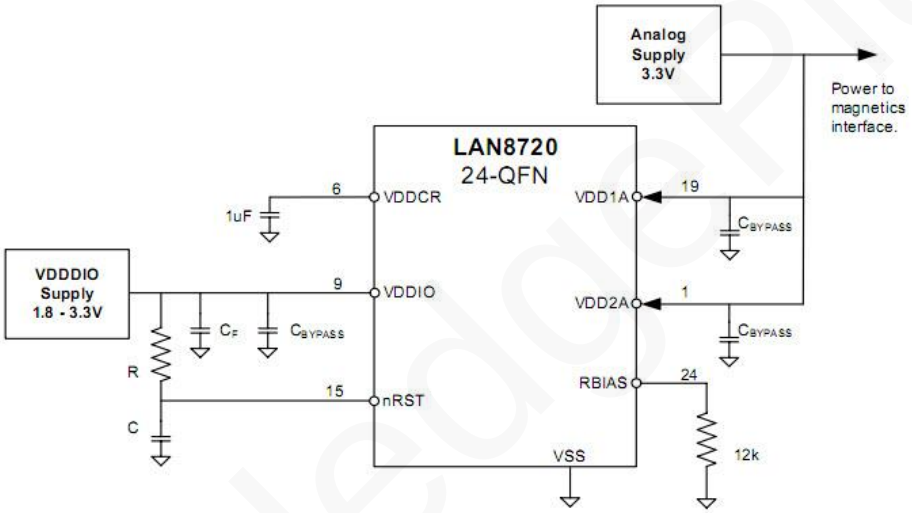


LAN8720/LAN8720i System Block Diagram

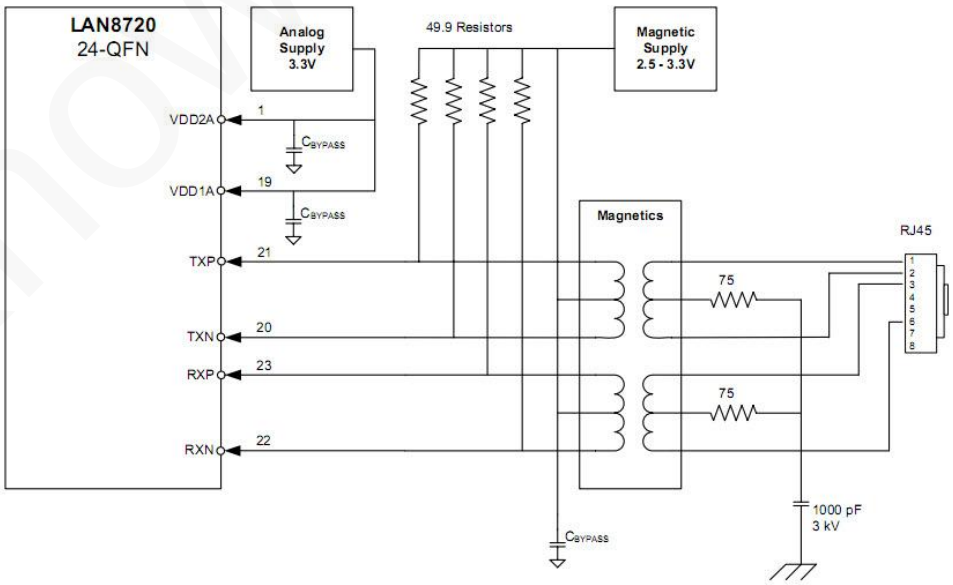
شکل ۶-۴۳



شکل ۶-۴۴

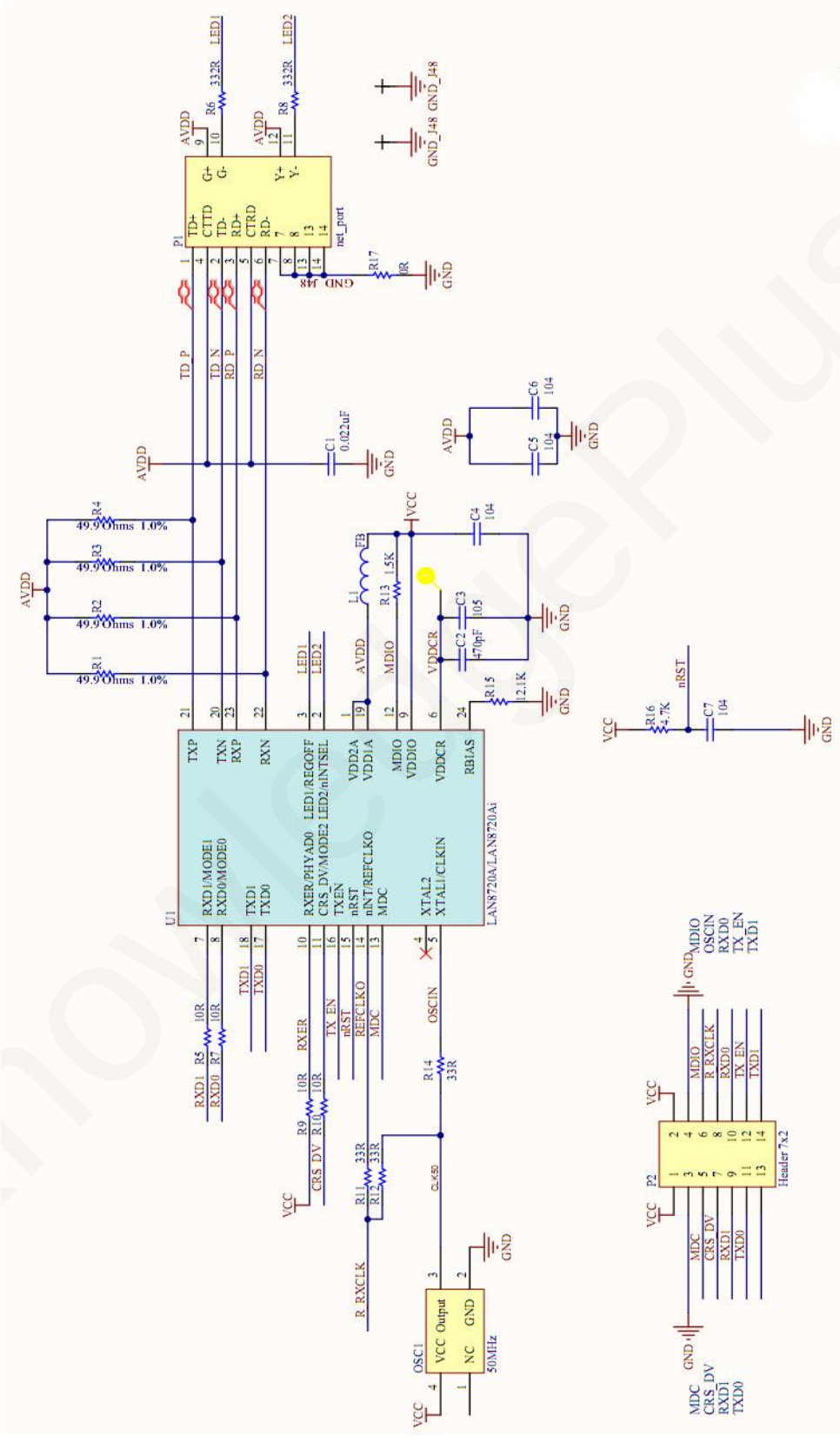


شکل ۶-۴۵



شکل ۶-۴۶

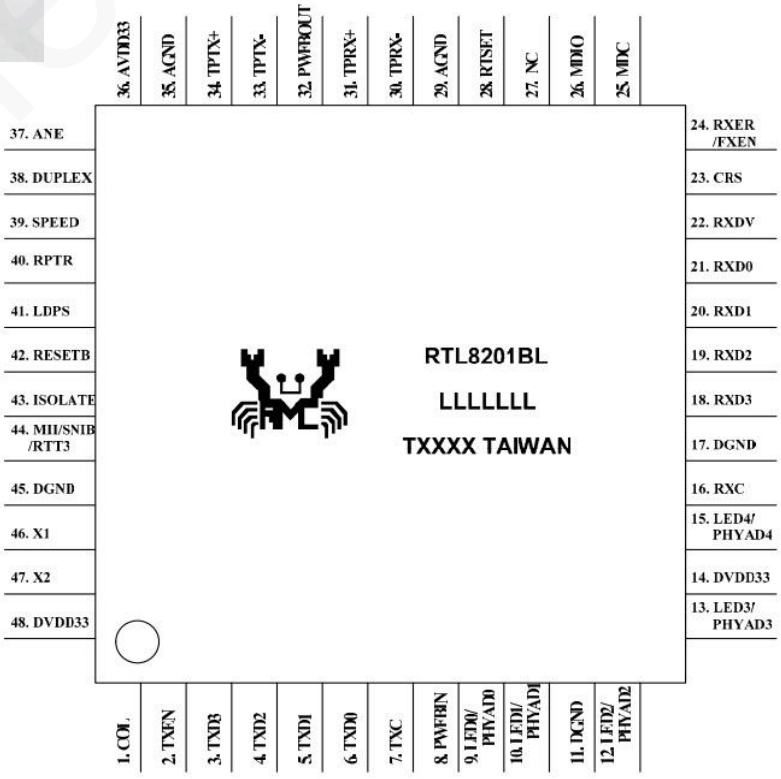
شکل زیر مدار تست شده به صورت عملی را با استفاده از این تراشه برای پیاده سازی لایه فیزیکی نشان می دهد.



شکل ۶-۴۷

٦-٨-٦. تراشه RTL8201BL

- پشتیبانی از سرعت های 10Mbit/s و 100Mbit/s (10Base-Tx ، 100Base-FX)
- پشتیبانی از کابل فیبر نوری (طبق استاندارد 100Base-FX)
- تطبیق با استانداردهای IEEE 802.3 ، IEEE 802.3u
- پشتیبانی از AUTO-MDIX
- تغذیه 3.3v
- قابلیت تحمل ولتاژ 5v در پایه های ورودی
- استفاده از تکنولوژی CMOS و توان مصرفی بسیار پایین
- محافظت در برابر ESD
- بسته بندی QFP ۴۸ پایه



شکل ۶-۴۸

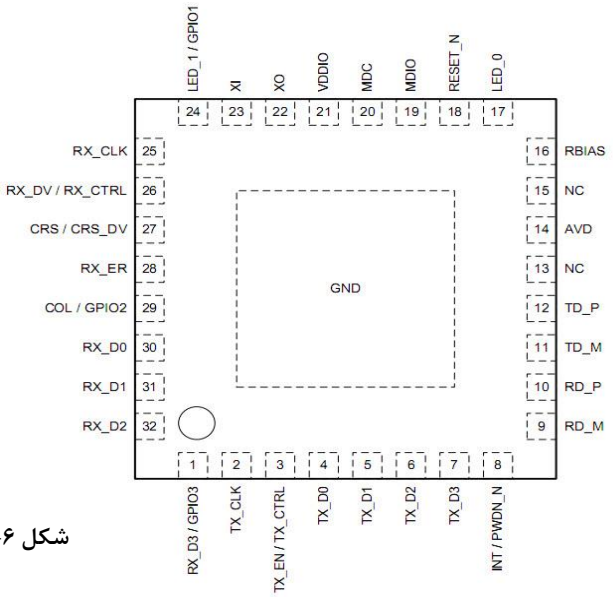
پایه های TPRX- و TPRX+ به همراه پایه های TPTX- و TPTX+ برای اتصال به شبکه به کار می روند.

به پایه های X1 و X2 کریستال 25MHZ متصل می شود.

پایه های TXD0 تا TXD4 و پایه های RXD0 تا RXD3 برای ارسال و دریافت داده به همراه پایه های جانبی دیگر برای اتصال به میکروپروسسور یا میکروکنترلر مورد استفاده قرار می گیرند.

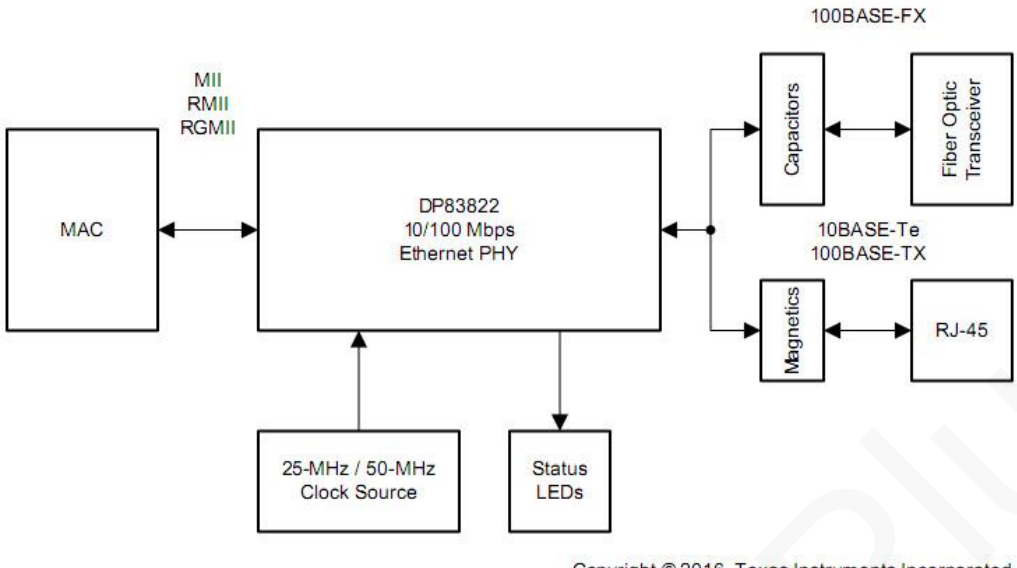
٦-٨-٧. تراشه DP83822

- پشتیبانی از سرعت های 10Mbit/s و 100Mbit/s (10Base-Te ، 100Base-TX و 100Base-FX)
- مطابق استاندارد IEEE 802.3u و IEEE 802.3az
- تشخیص خرابی کابل شبکه
- قابلیت اتصال به فیبر نوری توسط تراشه transceiver واسط
- پشتیبانی از AUTO-MDIX
- محافظت در برابر ESD تا ولتاژ 8Kv
- بسته بندی QFN ۳۲ پایه



شکل ۶-۴۹

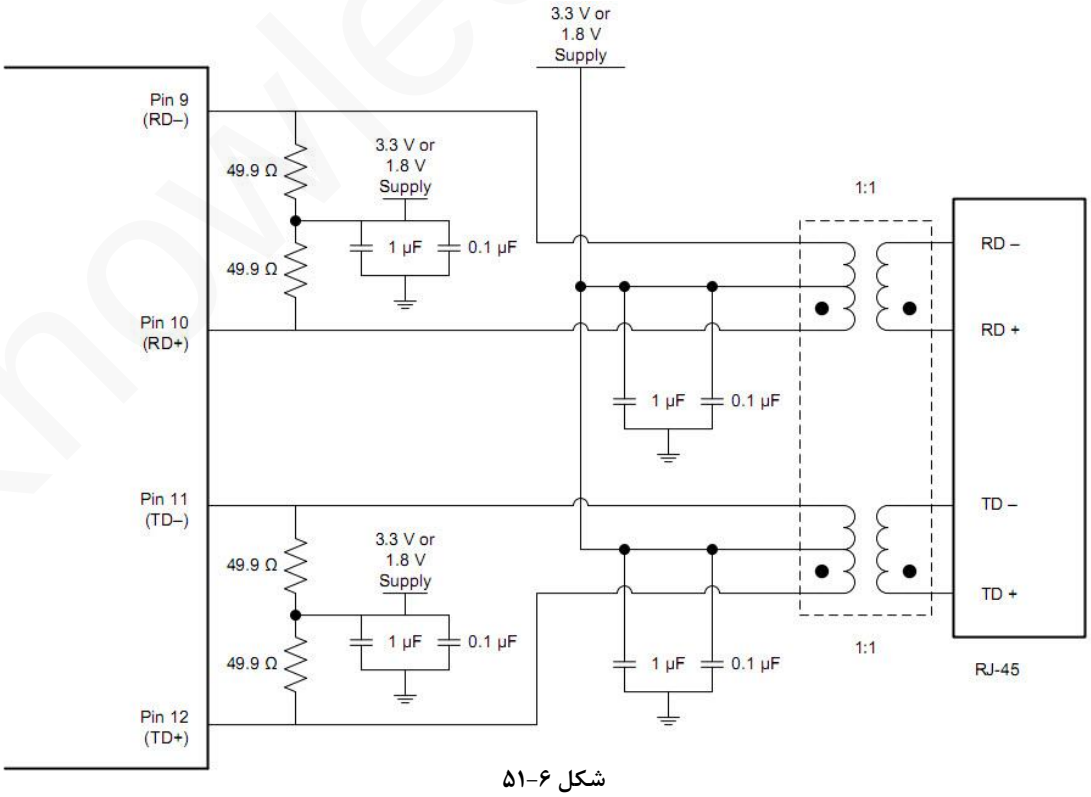
شکل زیر بلوک دیاگرام طراحی با این تراشه را نشان می دهد.



شکل ۶-۵۰

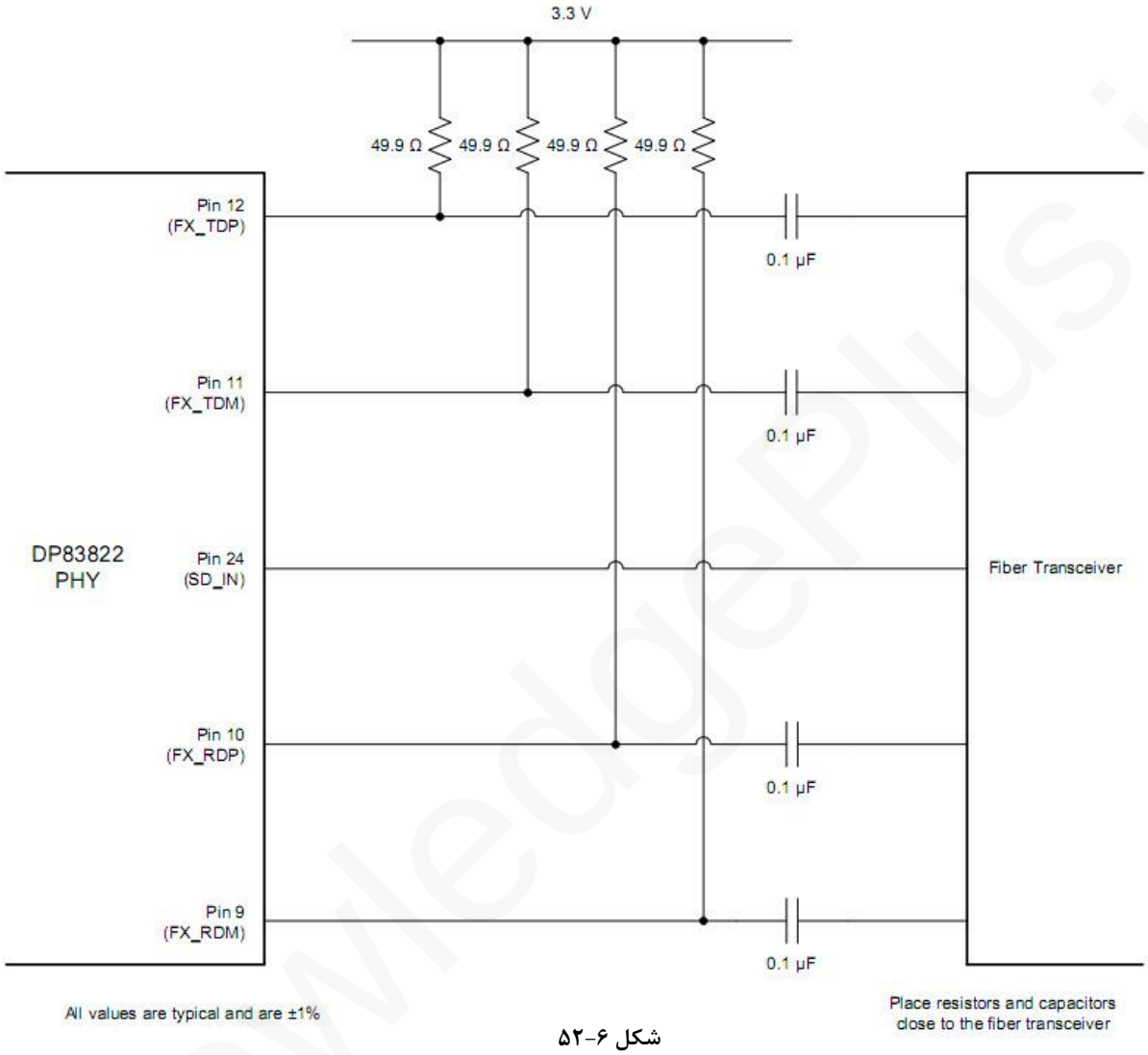
مطابق شکل بالا، برای اتصال به شبکه توسط کابل های CAT (استاندارد 10Base-Te و 100Base-TX) توسط ترانسفورماتور و مدارات مناسب این عمل صورت می پذیرد ولی برای اتصال به فیبر نوری به یک شبکه خازنی و تراشه transceiver مناسب برای فیبر نوری نیاز می باشد.

شکل زیر مدارات مناسب (بلوک magnetics) برای اتصال تراشه به کانکتور RJ-45 را نشان می دهد.



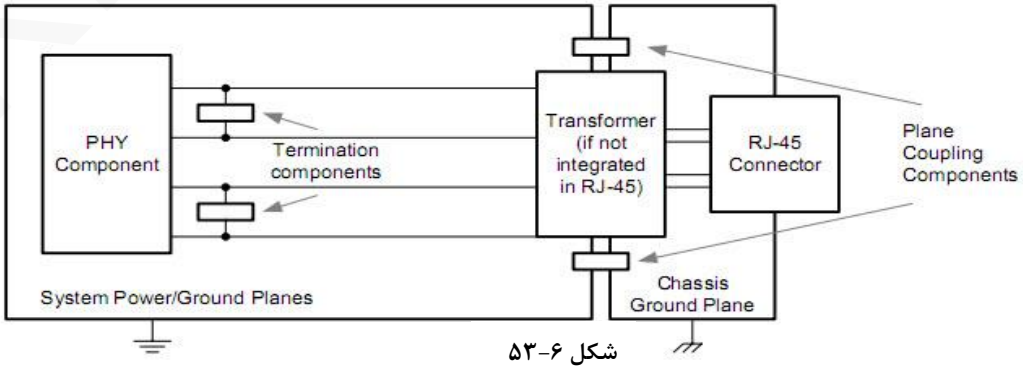
شکل ۶-۵۱

شکل زیر استفاده از شبکه خازنی و تراشه transceiver برای فیبر نوری را نشان می دهد. مقاومت ها و خازن ها باید تا حد امکان به تراشه fiber transceiver نزدیک باشند.



شکل ۶-۵۲

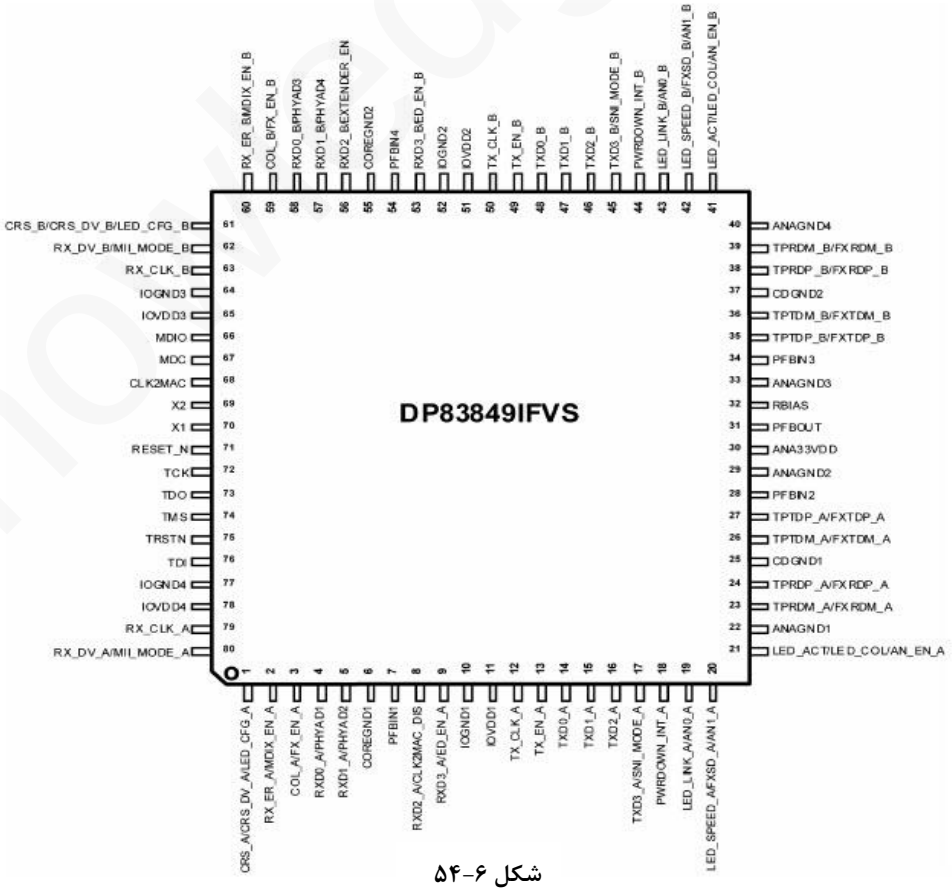
شکل زیر راهنمای طراحی PCB برای این تراشه را نشان می دهد.



شکل ۶-۵۳

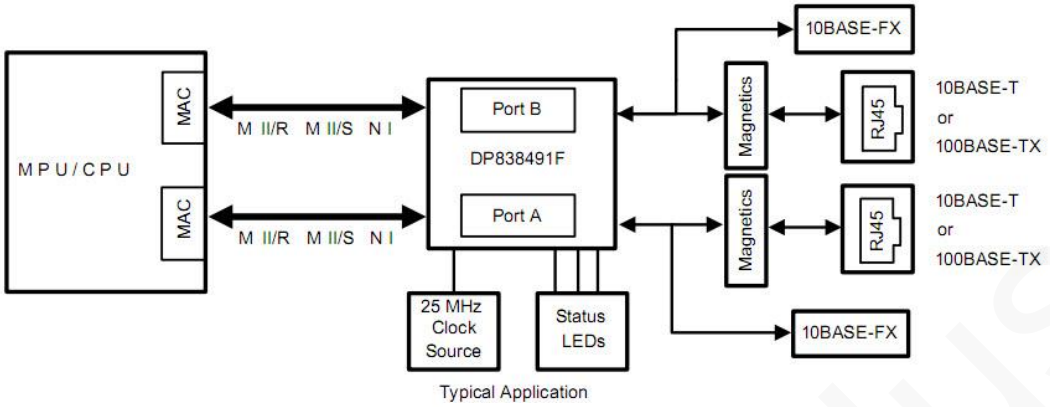
٦-٨-٨. تراشه DP83849IF

- پشتیبانی از سرعت های 10Mbit/s و 100Mbit/s (10Base-T ، 100Base-TX و 100Base-FX)
- تطبیق با استاندارد IEEE 802.3u
- دارای دو پورت Ethernet
- دارای JTAG (IEEE 1149.1) برای طراحی ساده و اشکال زدایی
- پشتیبانی از AUTO-MDIX ، Auto-negotiation
- تشخیص اتصال کابل و مشکلات کابل شبکه
- ولتاژ تغذیه 3.3v
- قابلیت تحمل ولتاژ 3.6v بر روی پایه های I/O
- بسته بندی TQFP ۸۰ پایه
- تراشه نسبتا پیشرفته با امکانات بسیار زیاد



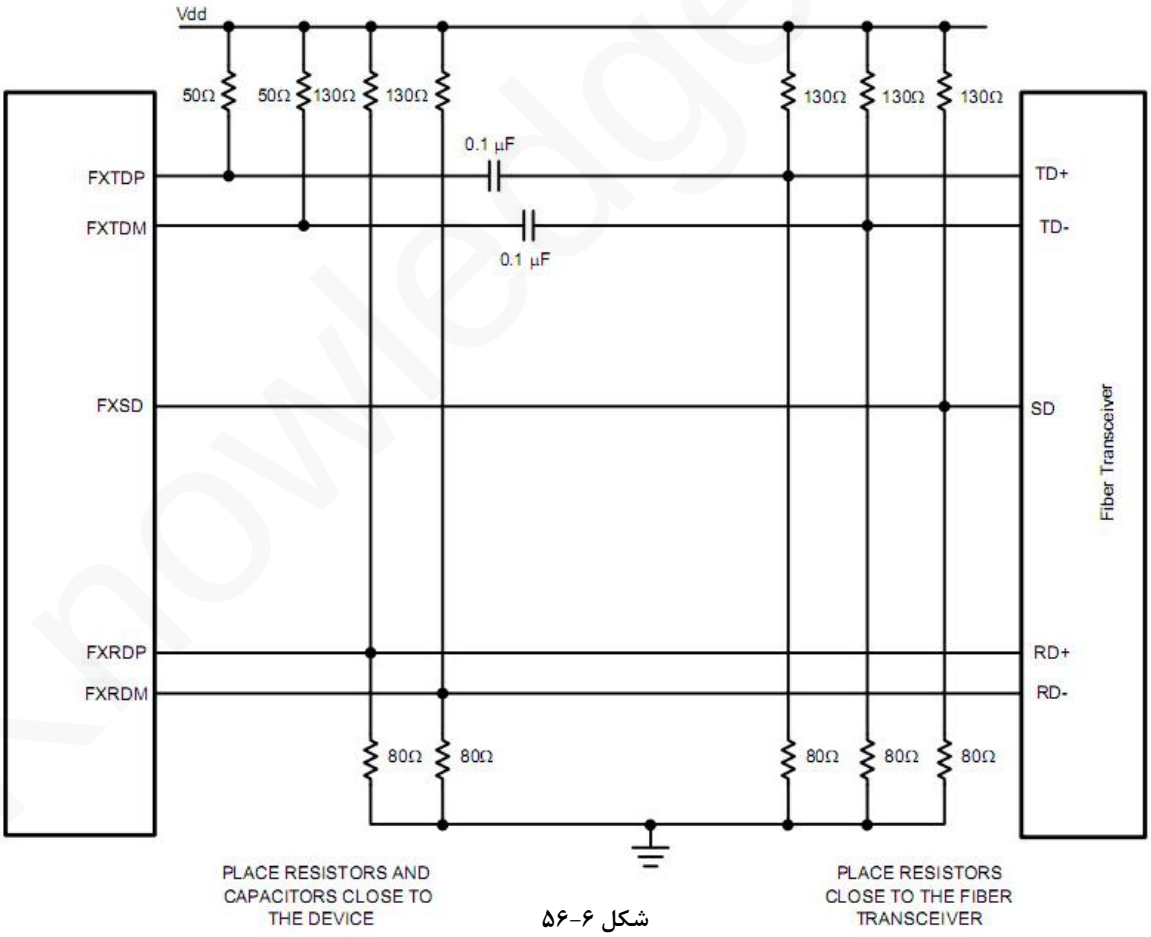
شکل ۶-۵۴

شکل زیر بلوک دیاگرام استفاده از این تراشه را نشان می دهد.



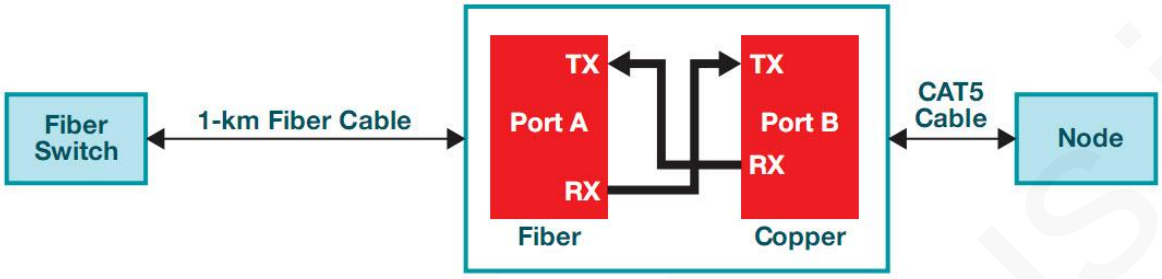
شکل ۶-۵۵

شکل زیر مدار کاربردی استفاده از این تراشه برای سرعت 100Mbit/s با استفاده فیبر نوری را نشان می دهد.



شکل ۶-۵۶

یکی از ویژگی های متمایز کننده این تراشه وجود واحد media conversion در آن است که میتواند به عنوان واسطی بین فیبر نوری و دستگاه ها و تجهیزاتی که از فیبر نوری پشتیبانی نمی کنند عمل کند.



شکل ۶-۵۷

٦-۹. پیاده سازی Gigabit Ethernet

تا این بخش تراشه های معرفی شده مربوط به پیاده سازی اترنت استاندارد (10Mbit/s) و Fast Ethernet (100Mbit/s) می باشد. در این بخش چند تراشه برای پیاده سازی gigabit Ethernet پیشنهاد می شود.

٦-۹-۱. تراشه BCM5482

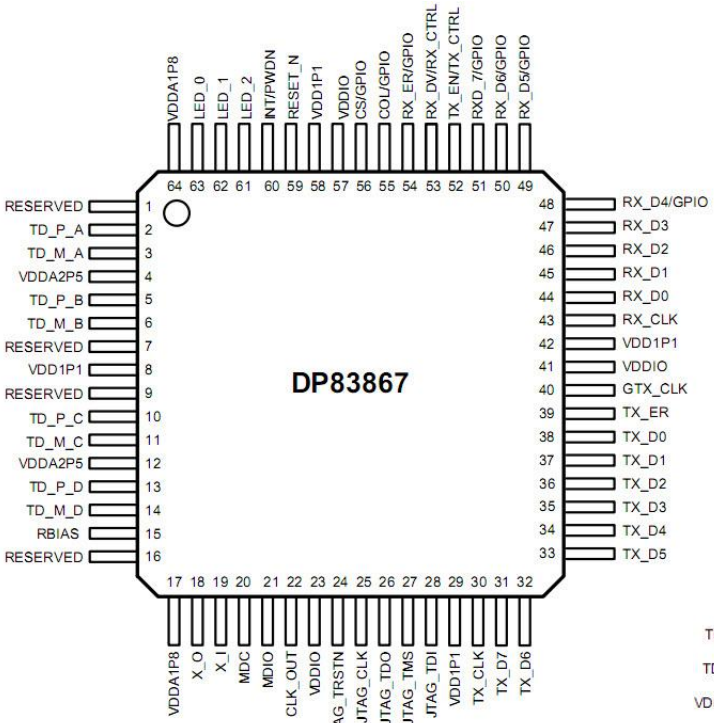
- پشتیبانی از سرعت های 10Mbit/s ، 100Mbit/s و 1000Gbit/s (10Base-T ، 100Base-TX و 1000Base-T)
- دارای دو درگاه (Dual-Port)
- پشتیبانی از AUTO-MDIX
- تشخیص اتصال کابل و عیب های احتمالی (Cable Checker)
- پشتیبانی از JTAG
- بسته بندی BGA
- قابلیت تحمل تخلیه الکتریسیته ساکن (ESD) تا ولتاژ 3Kv

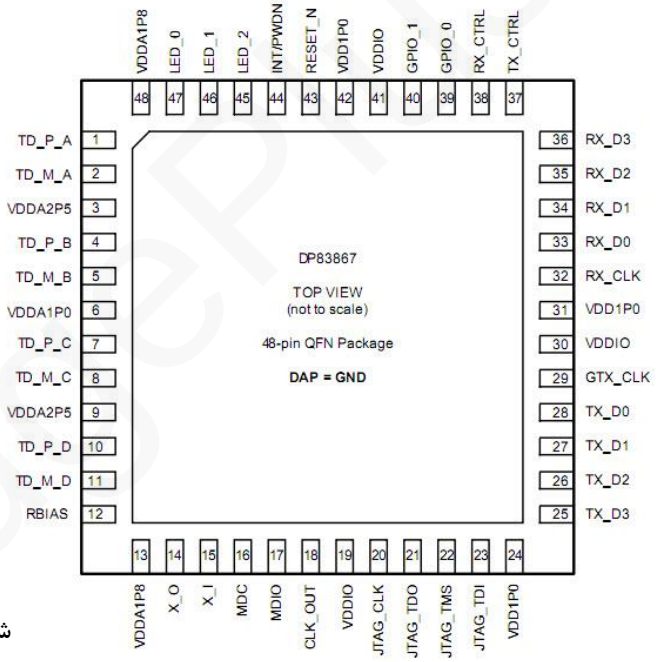
- تشعشعات الکترومغناطیسی (EMI) بسیار کم
- استفاده از تکنولوژی CMOS و توان مصرفی بسیار پایین

شکل ۶-۵۸

۶-۹-۲. تراشه DP83867IR/CR

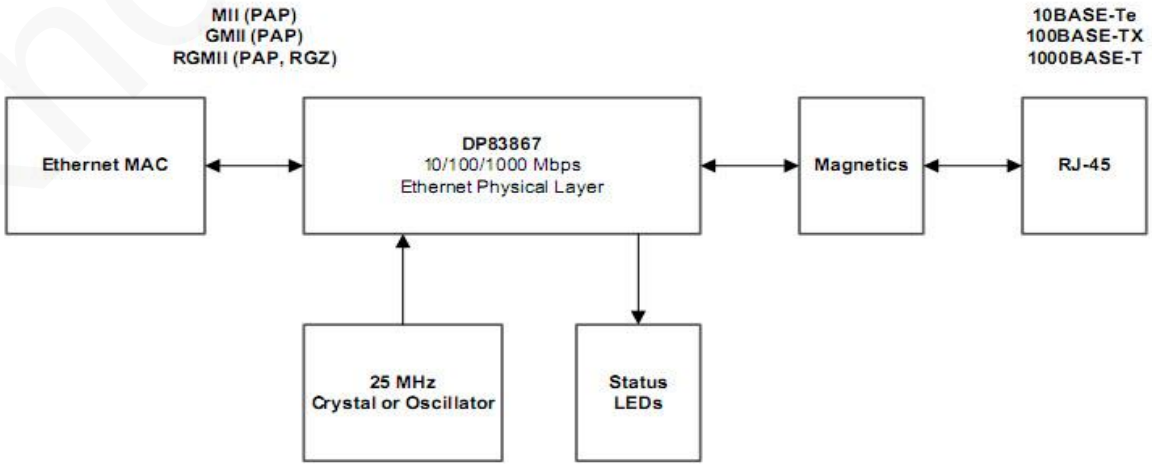
- پشتیبانی از سرعت های 10Mbit/s و 100Mbit/s (10Base-Te ، 100Base-T و 100Base-TX
- منطبق با استاندارد IEEE 802.3
- منطبق با استاندارد IEC 61000-4-2 ESD ، تحمل ولتاژهای بیش از 8Kv
- منطبق با استاندارد EN55011 Class B
- منطبق با استاندارد IEEE 1588
- تشخیص اتصال کابل و مشکلات کابل شبکه
- بسته بندی QFP ۶۴ پایه و QFN ۴۸ پایه





شکل ۶-۵۹

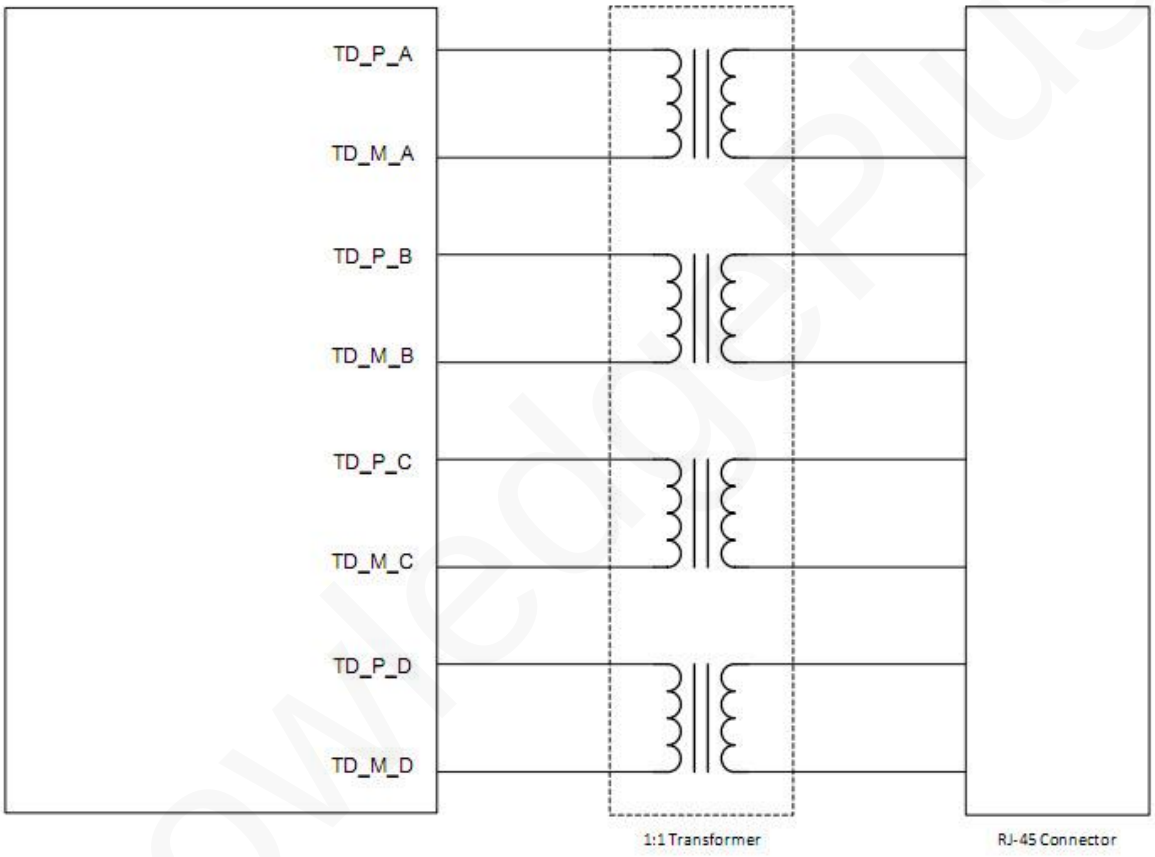
شکل زیر بلوک دیاگرام استفاده از این تراشه را نشان می دهد.



شکل ۶-۶۰

مطابق شکل زیر با توجه به مدارات تعبیه شده در داخل این تراشه، نیازی به استفاده از مقاومت ها و خازن های خارجی در این تراشه وجود ندارد.

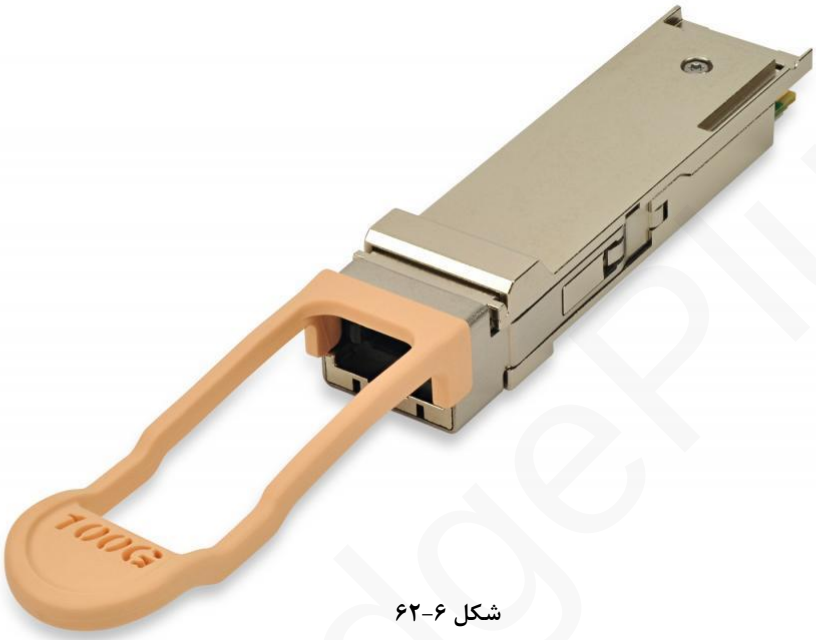
با توجه به توضیحات ارائه شده در فصول قبلی، در سرعت های بالا، نیاز به استفاده از هر ۴ زوج سیم وجود دارد، لذا مطابق شکل زیر باید از چهار ترانسفورماتور برای ایزوله کردن تراشه در هر ۴ زوج سیم استفاده شود.



شکل ۶-۶۱

نکته برای سرعت های بالا مانند 10Gbit/s و بالاتر، باید از transceiver های optical یا نوری و کابل فیبر نوری استفاده کرد.

شکل زیر یک نمونه از این transceiver ها را که توسط شرکت finisar ساخته شده است را نشان می دهد.



شکل ۶-۶۲

مدل FTLC9551REPM :

- کاربرد در لینک های اترنت با سرعت 100Gbit/s با استفاده از فیبر نوری multimode
- مطابق با استاندارد 802.3bm
- قابلیت های تشخیص خطا توسط رابط I2C
- حداکثر توان تلفاتی 3.5W
- تغذیه 3.3V
- حداکثر طول کابل ۱۰۰ متر
- گیرنده دیود PIN

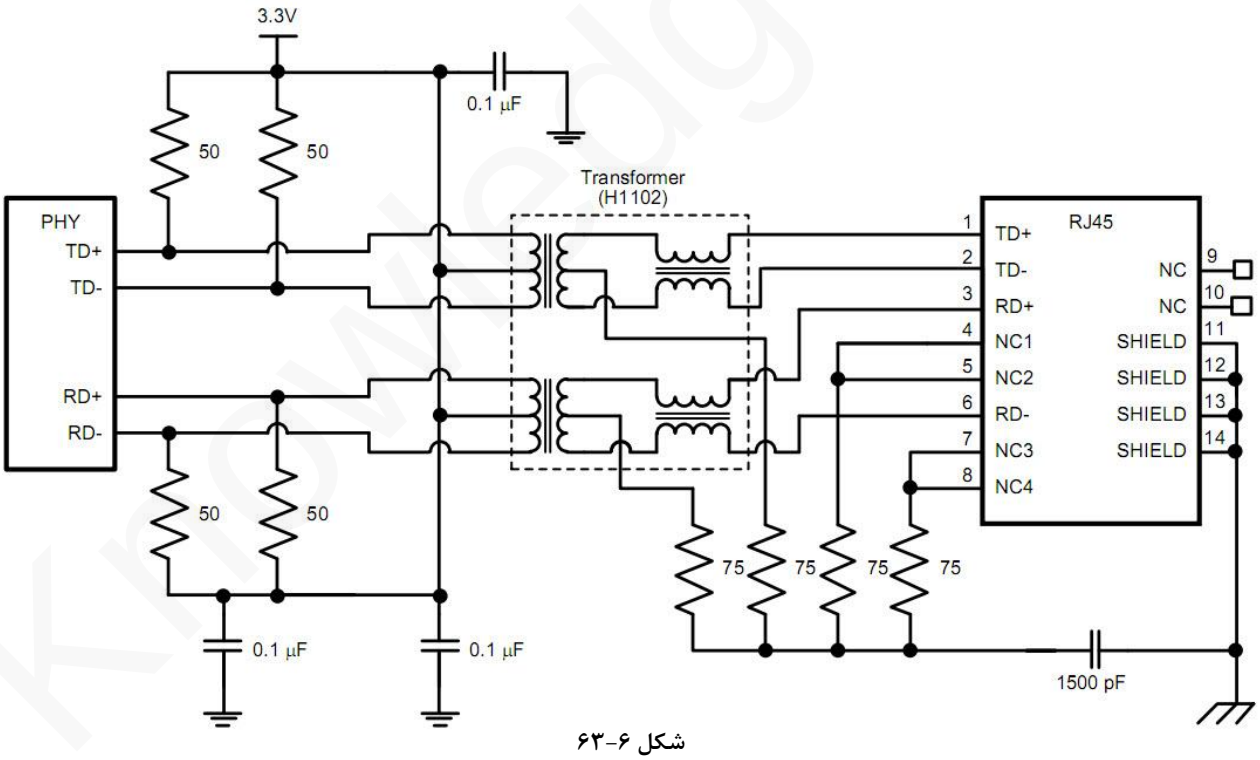
۶-۱۰. کاربردهای بدون ترانسفورماتور

در همه ی مدارات ارائه شده در بخش های قبلی، ترانسفورماتور جزء مشترک همه ی مدارات می باشد. کار ترانسفورماتور ایجاد ایزولاسیون DC بین گره های مختلف شبکه و همچنین بایاس DC پایه های فرستنده و گیرنده است.

در کاربردهایی که هزینه و فضای PCB اهمیت زیادی دارد و مدار باید در شرایط سخت درجه حرارتی کار کند، استفاده از ترانسفورماتور مطلوب نیست.

در این بخش راه کارهایی برای اتصال به شبکه اترنت بدون استفاده از ترانسفورماتور مبتنی بر محصولات دو شرکت Microchip و شرکت TI ارائه خواهد شد.

برای توضیح این روش، ابتدا توپولوژی متداول مورد استفاده توسط ترانسفورماتور را توضیح می دهیم. مطابق شکل زیر، گره ها از شبکه توسط یک ترانسفورماتور 1:1 از نظر DC ایزوله می شوند. همچنین ترانسفورماتور بایاس DC لایه فیزیکی تراشه را به عهده دارد.

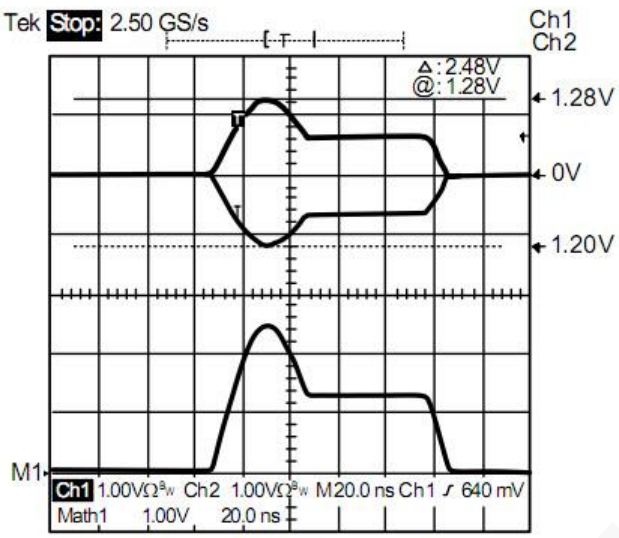


شکل ۶-۶۳

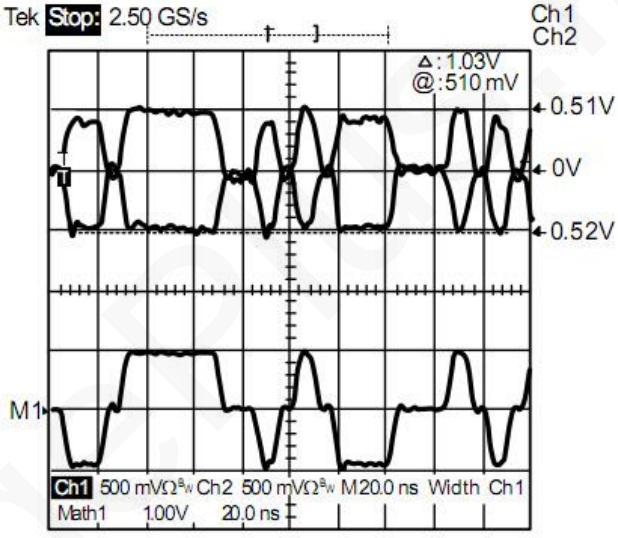
همان طور که در فصل چهارم نیز توضیح داده شد، فرایند auto-negotiation از پالس های اتصال (link puls) برای تشخیص مشخصات شبکه استفاده می کند. پالس های اتصال (در صورتی که بار ۵۰ اهم

متعادل به عنوان خروجی تراشه های transceiver در نظر گرفته شود) در سرعت 10Mbit/s شامل دو سیگنال تفاضلی 2.5v+ و 2.5v- می شود و در سرعت 100Mbit/s شامل سیگنال های تفاضلی 1v+ ، 0v و 1v- می شود.

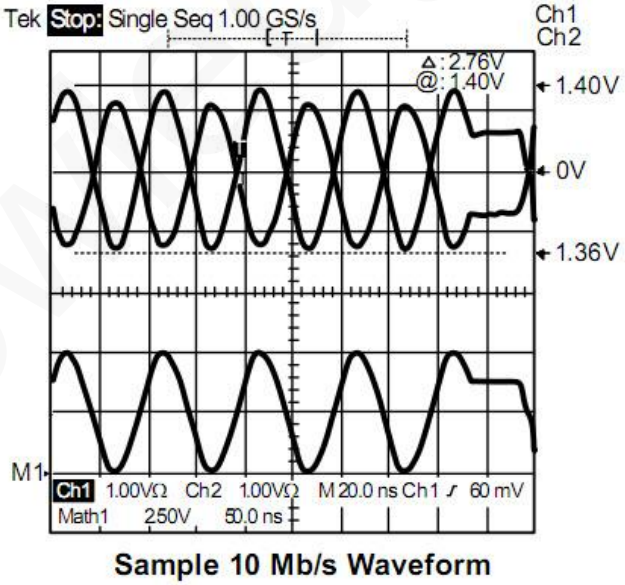
شکل های زیر نشان دهنده ی این سیگنال ها می باشند.



Sample Link Pulse waveform



Sample 100 Mb/s Waveform (MLT-3)



Sample 10 Mb/s Waveform

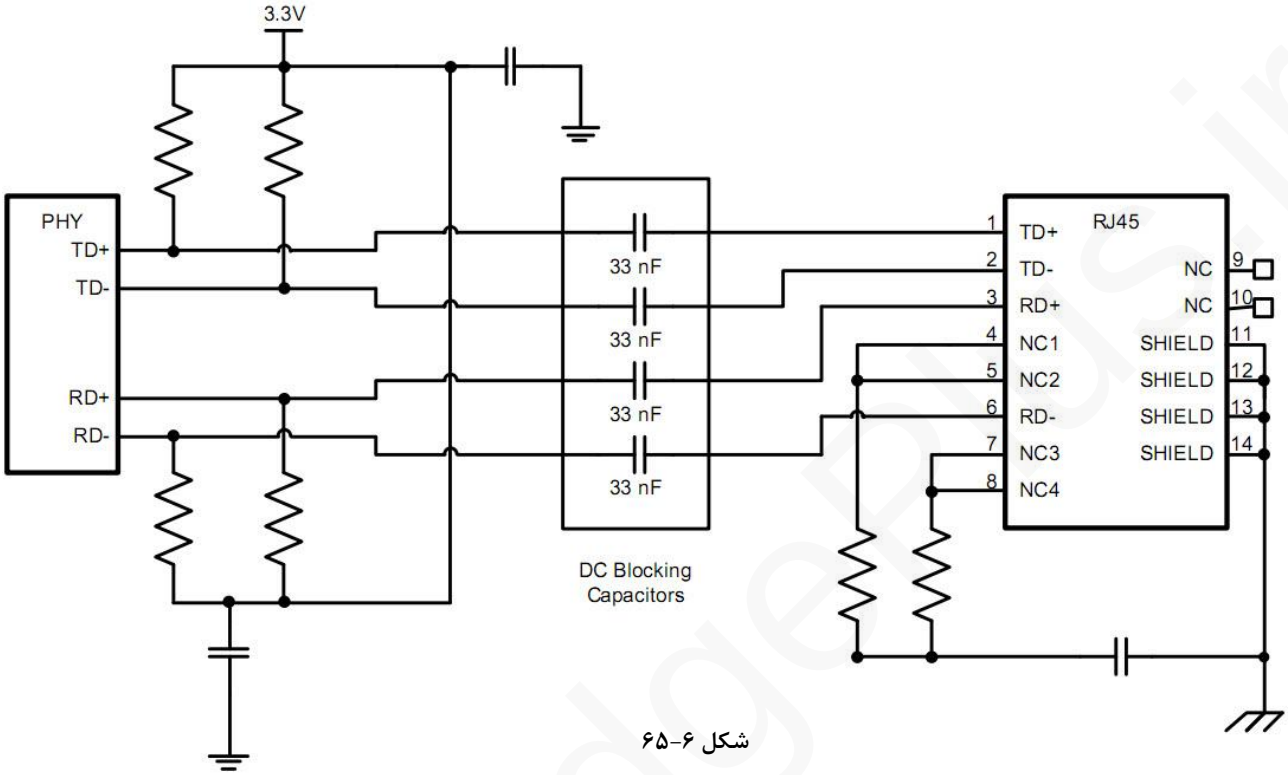
شکل ۶-۶۴

طبق مشخصات استاندارد IEEE 802.3 در بخش ایزوله سازی، بخش فیزیکی باید قابلیت تحمل ولتاژهای 1500v فرکانس 50Hz یا 60Hz و ولتاژ 2250Vdc را برای مدت ۶۰ ثانیه داشته باشد.

در محصولات سری PHYTER شرکت TI، بایاس DC بخش فیزیکی تراشه از طریق سر وسط اولیه

ترانسفورماتورها و همچنین توسط مقاومت های ۵۰ اهمی صورت می پذیرد.

در کاربردهای بدون ترانسفورماتور مطابق شکل زیر، ایزوله سازی توسط خازن های بدون پلاریته صورت می پذیرد.

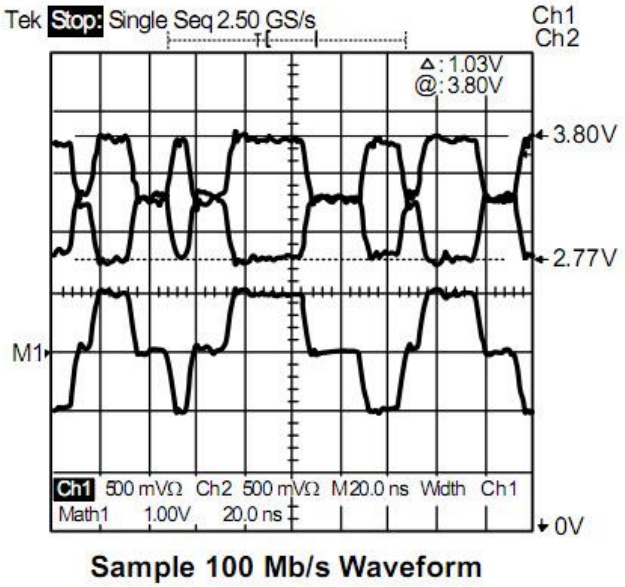


شکل ۶-۶۵

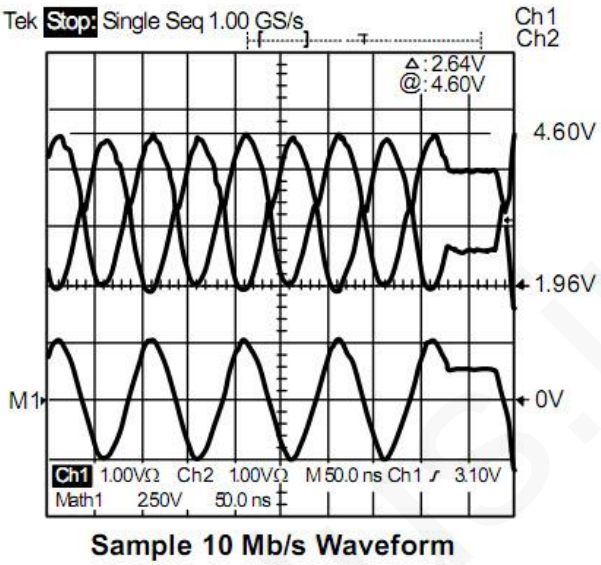
شکل های صفحه بعد، مقایسه سیگنال های شبکه در دو حالت با ترانسفورماتور ایزوله کننده و بدون ترانسفورماتور ایزوله کننده (با خازن های ایزوله کننده) را در دو سرعت 10Mbit/s و 100Mbit/s نشان می دهند.

مطابق شکل ۶-۶۶ ، در سرعت 100Mbit/s در حالت بدون ترانسفورماتور، ولتاژ ها در محدوده 1.8v تا 2.8v هستند و با برعکس شدن پلاریته، ولتاژ های 1v+ و 1v- تولید می شوند.

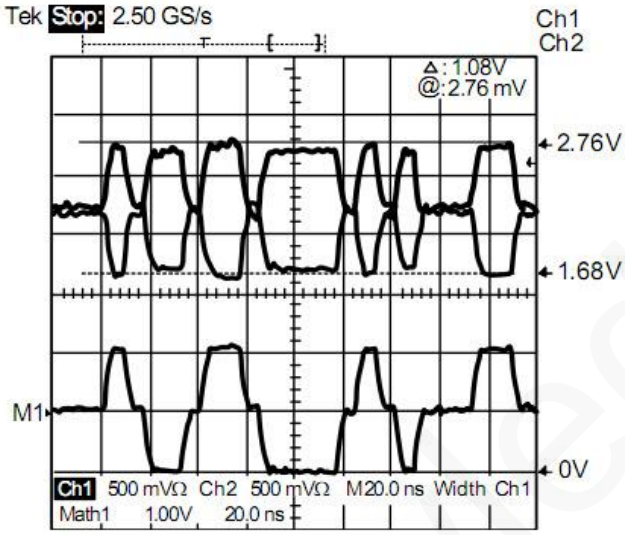
در سرعت 10Mbit/s در حالت بدون ترانسفورماتور، سیگنال ها نامتعادل (نامتقارن) هستند و در مقایسه با حالتی که از ترانسفورماتور استفاده می شود، مقدار EMI القاء شده بر روی کابل ها افزایش پیدا کرده است و در نتیجه پیشنهاد می شود استفاده از کاربرد بدون ترانسفورماتور در سرعت 100Mbit/s و در حالت غیرفعال بودن autonegotiation استفاده شود.



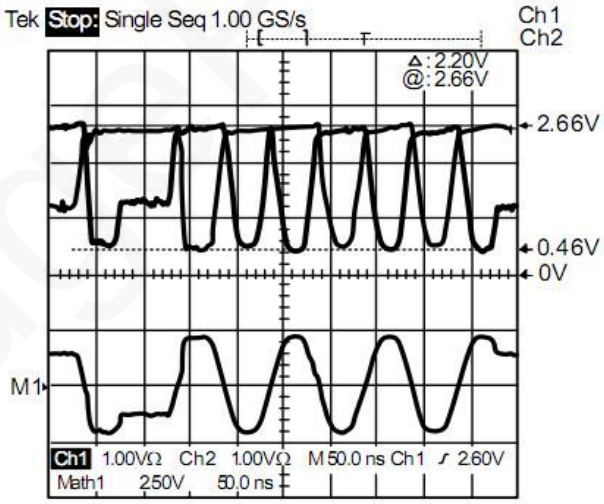
Sample 100 Mb/s Waveform



Sample 10 Mb/s Waveform



Sample 100 Mb/s Waveform With No Transformer



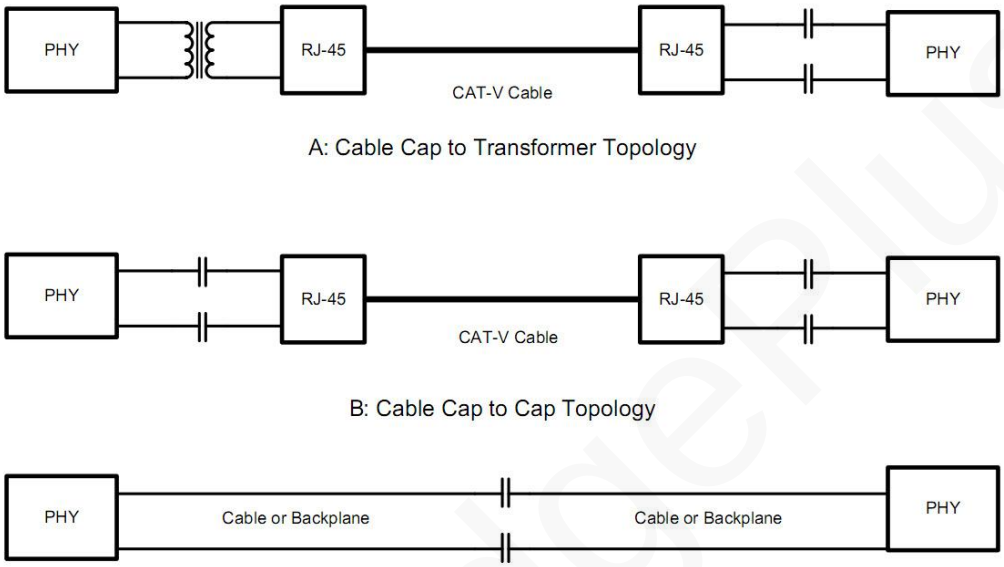
Sample 10 Mb/s Waveform With No Transformer

شکل ۶-۶۶

در بخش گیرنده، سیگنال های دریافت شده در حالت استفاده از ترانسفورماتور و بدون ترانسفورماتور مشابه هستند، در نتیجه با توجه به این که قابلیت MDIX بر مبنای سیگنال های دریافت شده در گیرنده می باشد، استفاده از قابلیت MDIX در حالت بدون ترانسفورماتور نیز مشکلی به وجود نمی آورد.

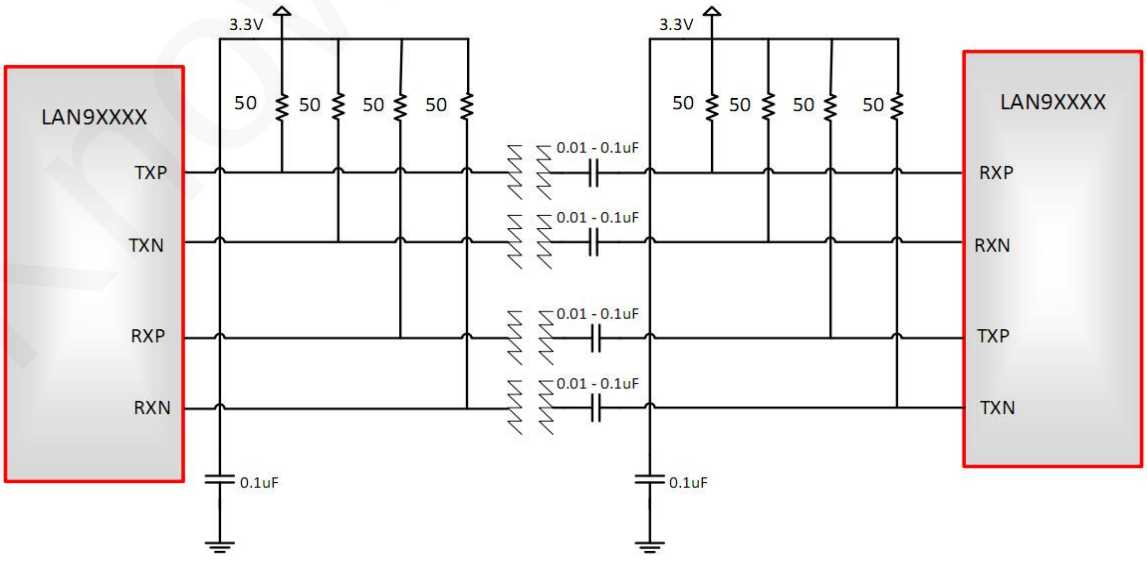
بر مبنای محاسبات انجام شده طبق استاندارد ANSI INCITS 263-1995 TP-PMD، استفاده از خازن های multi-layer ceramic (خازن های عدسی) با ظرفیت حداقل 33nF به عنوان خازن های ایزوله کننده پیشنهاد می شود. حداکثر ظرفیت خازن محدودیتی ندارد اما باید به این نکته توجه داشته باشیم که افزایش ظرفیت خازن باعث کاهش ایزولاسیون DC می شود.

شکل های زیر انواع حالت های مورد استفاده در کاربردهای بدون ترانسفورماتور را نشان می دهند. کابل مورد استفاده معمولا CAT 5 با مقاومت ۱۰۰ اهم یا خطوط PCB با مقاومت ۵۰ اهم می باشد. همچنین پیشنهاد می شود طول کابل حداکثر ۱ متر باشد. طبق استاندارد IEEE 802.3، زوج پایه های فرستنده و گیرنده باید دارای امپدانس تفاضلی ۱۰۰ اهم برای مسافت های طولانی و ۵۰ اهم برای مسافت های کوتاه باشند.



شکل ۶-۶۷

شکل زیر اتصال دو تراشه اترنت سری LAN9XXXX شرکت Microchip بدون استفاده از ترانسفورماتور نشان می دهد.



شکل ۶-۶۸

نکته : در صورت عدم استفاده از ترانسفورماتور، باید قابلیت های Auto-Negotiation و Auto-MDIX در تراشه های Microchip غیرفعال شوند.

فصل هفتم

تراشه ENC28J60

۷-۱. مقدمه

در فصل ششم روش های مختلف پیاده سازی عملی را بررسی کردیم. همان طور که در آن فصل ذکر شد، در این مقاله آموزشی به منظور انجام پروژه های عملی از تراشه ENC28J60 استفاده شده است. علاوه بر دلایل ذکر شده در فصل ششم، یکی دیگر از دلایل استفاده از این تراشه، وجود کدهای مختلف به صورت آماده در کاربردهای مختلف برای پیاده سازی است که میتواند نقطه شروع بسیار مناسبی برای آموزش کار با شبکه اترنت باشد.

در این فصل به توضیح امکانات این تراشه و رجیسترهای آن برای کدنویسی می پردازیم. در فصل آخر این مقاله طبق توضیحات این فصل، در کتابخانه ENC28J60 توابعی برای پیاده سازی هریک از این کارکردها نوشته خواهد شد.

نکته : در این فصل سعی می شود بخش های مهم دیتاشیت که برای کار با این تراشه لازم است بررسی شوند. توصیه می شود در صورتی که می خواهید از این تراشه در پروژه خود استفاده کنید، به بخش های دیگر دیتاشیت مانند مشخصات الکتریکی و دیگر جزئیات دیتاشیت مراجعه فرمایید.

۷-۲. ویژگی های تراشه ENC28J60

۷-۲-۱. ویژگی های بخش کنترل کننده ی Ethernet

- سازگاری با استاندارد IEEE 802.3
- پیاده سازی MAC و لایه ی فیزیکی ارتباط 10Base-T
- قابلیت automatic polarity detection and correction یا تشخیص خودکار پلاریته و تصحیح آن
- پشتیبانی از ارتباط Full-Duplex و Half-Duplex
- قابلیت برنامه ریزی برای ارسال مجدد در صورت ایجاد تصادم
- قابلیت برنامه ریزی برای ایجاد padding و CRC
- قابلیت برنامه ریزی برای نپذیرفتن بسته های داده اشتباه
- ارتباط SPI با حداکثر سرعت 20MHz

۷-۲-۲. ویژگی های بخش Buffer

- ۸ کیلوبایت SRAM برای ارسال و دریافت بسته های داده
- قابلیت برنامه ریزی ظرفیت بافرهای ارسال و دریافت
- بافر دریافت کننده چرخشی (circular) از نوع FIFO با قابلیت کنترل سخت افزاری
- DMA داخلی برای جابه جایی سریع اطلاعات
- انجام محاسبات checksum به کمک سخت افزار برای پروتکل های مختلف شبکه

۷-۲-۳. ویژگی های بخش MAC

- پشتیبانی از بسته های Unicast ، Multicast و Broadcast
- فیلتر کردن قابل برنامه ریزی بسته های دریافتی داده و بیدار شدن (wake-up) هنگام

AND منطقی یا OR منطقی در شرایط :

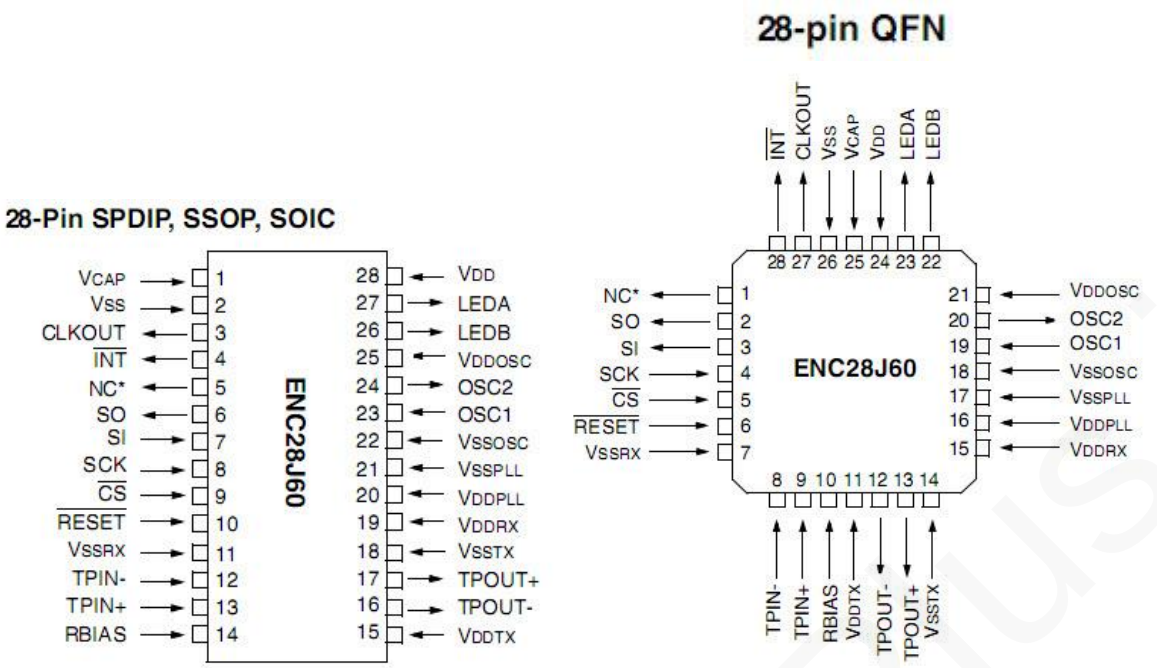
- آدرس مقصد Unicast
- آدرس Multicast
- آدرس Broadcast
- بسته های داده Magic
- آدرس های مقصد گروهی طبق جدول hash ۶۴ بیتی
- ۶۴ بایت الگوهای تطبیقی قابل برنامه ریزی به همراه تنظیم offset

۷-۲-٤. ویژگی های بخش PHY (لایه فیزیکی)

- پشیبانی از حالت Loopback
- دارای دو LED قابل برنامه ریزی برای شرایط LINK ، TX ، RX ، تصادم و وضعیت های full-duplex و half-duplex

۷-۲-۵. ویژگی های عملیاتی (Operational)

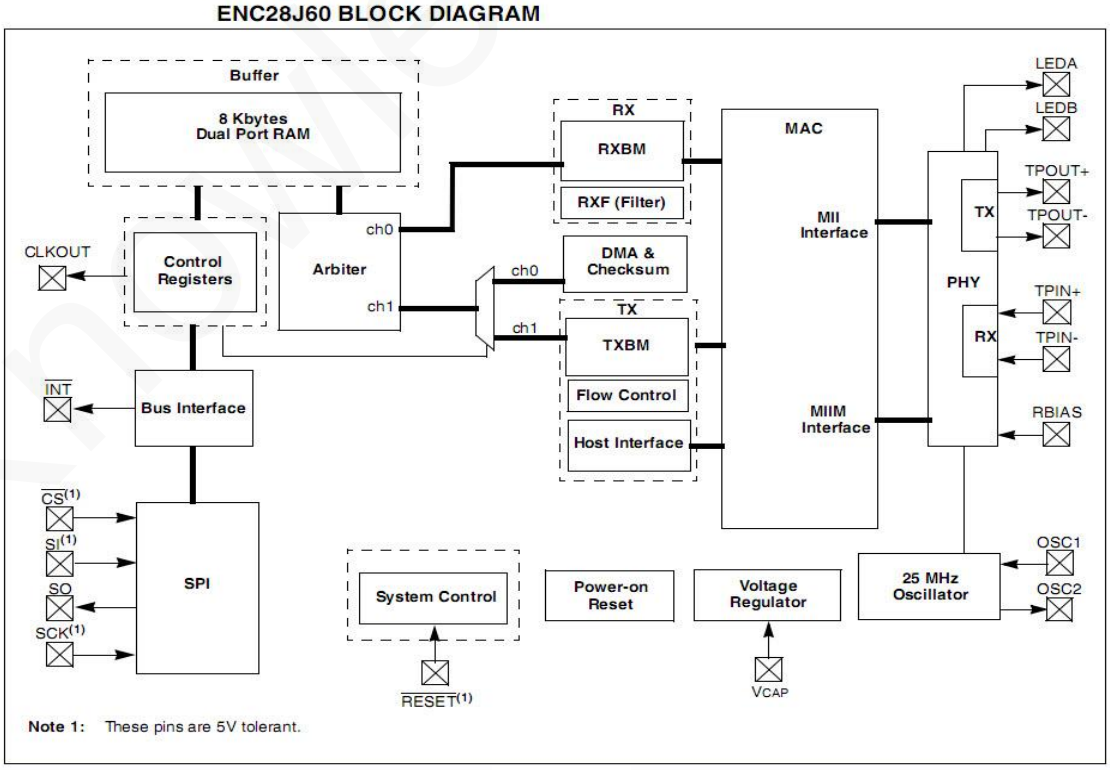
- ۶ منبع وقفه و یک پایه برای وقفه خارجی
- کلاک 25MHz
- پایه ی کلاک خارجی با پیش تقسیم کننده (prescaler) قابل برنامه ریزی
- ولتاژ تغذیه از 3.1v تا 3.6v (مقدار نامی 3.3v)
- قابلیت تحمل ولتاژ 5v در پایه های ورودی
- محدوده ی درجه حرارت : برای کاربردهای صنعتی از 40- درجه تا 85+ درجه ، از 0 درجه تا 70+ درجه برای کاربردهای تجاری (فقط در بسته بندی SSOP)
- دارای ۲۸ پایه در بسته بندی های SPDIP ، SSOP ، SOIC و QFN



شکل۷-۱

۷-۳. بلوک دیاگرام

مطابق شکل زیر تراشه ENC28J60 از ۷ بلوک اصلی تشکیل شده است :



شکل۷-۲

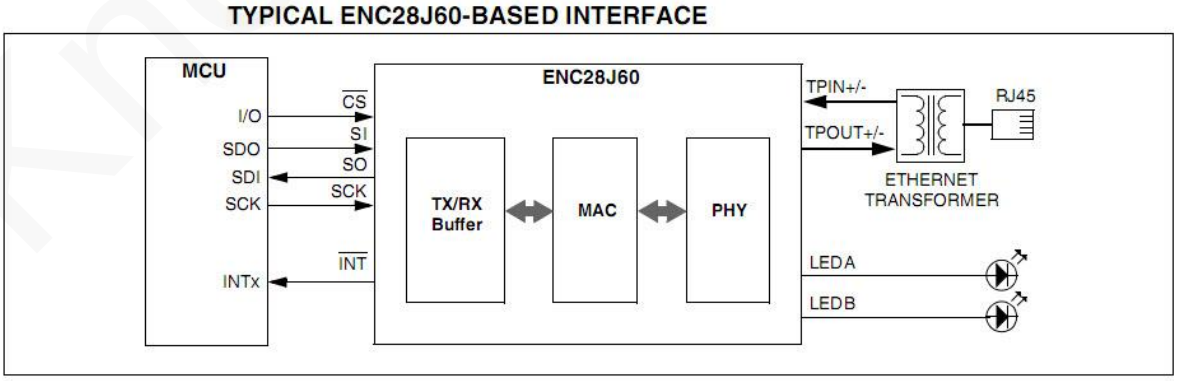
- واسط SPI برای ارتباط بین کنترل کننده و ENC28J60
- رجیسترهای کنترلی برای کنترل و نظارت بر عملکرد ENC28J60
- بافر برای بسته های ارسالی و دریافتی
- واحد arbiter برای کنترل دسترسی به بافر هنگام درخواست از طرف DMA، بلوک های فرستنده و گیرنده
- واسط bus برای تفسیر داده ها و دستورات دریافت شده از طریق واسط SPI
- واحد MAC، پیاده سازی طبق استاندارد IEEE 802.3
- لایه ی فیزیکی (PHY)، برای کد کردن و دیکد کردن داده های آنالوگ موجود روی خط انتقال زوج به هم تابیده

همچنین بلوک های جانبی دیگر مانند اسیلاتور، رگولاتور داخلی، واحد کنترل کننده منطق (control logic) و مبدل سطح (level translator) برای ایجاد قابلیت تحمل ولتاژ 5v روی پایه های I/O

۷-٤. اتصال تراشه ENC28J60 به کنترل کننده

شکل زیر کاربرد متداول این تراشه برای پیاده سازی سخت افزار لازم جهت اتصال به شبکه اترنت را نشان می دهد.

همان طور که مشاهده می شود تنها با دو ترانسفورماتور پالس و تعدادی عنصر غیرفعال می توان میکروکنترلر را به شبکه اترنت متصل کرد.



شکل۷-۳

۷-۵. توضیحات پایه های تراشه ENC28J60

نام پایه	توضیحات
VCAP	خروجی 2.5v رگولاتور داخلی، یک خازن با ESR پایین با مقدار متداول 10uF و مقدار حداقل 1uF باید به این پایه و زمین متصل گردد.
VSS	پایه زمین تراشه
CLKOUT	پایه کلاک خارجی قابل برنامه ریزی
INT	وقفه خارجی
NC	بدون اتصال
SO	پایه داده خروجی برای SPI
SI	پایه داده ورودی برای SPI
SCK	پایه کلاک برای SPI
CS	پایه ورودی انتخاب تراشه برای SPI
RESET	پایه ریست تراشه، فعال با سطح صفر
VSSRX	پایه زمین تراشه برای لایه ی فیزیکی گیرنده
TPIN-	ورودی سیگنال تفاضلی
TPIN+	ورودی سیگنال تفاضلی
RBIAS	جریان بایاس برای لایه ی فیزیکی، باید توسط یک مقاومت به زمین متصل گردد. (در ادامه توضیحات بیشتری داده می شود)
VDDTX	پایه تغذیه مثبت برای لایه فیزیکی بخش فرستنده
TPOUT-	خروجی سیگنال تفاضلی
TPOUT+	خروجی سیگنال تفاضلی
VSSTX	پایه زمین تراشه برای لایه ی فیزیکی فرستنده
VDDRX	پایه تغذیه مثبت 3.3v برای لایه فیزیکی بخش گیرنده
VDDPLL	پایه تغذیه مثبت 3.3v برای لایه فیزیکی بخش PLL
VSSPLL	پایه زمین برای لایه فیزیکی بخش PLL
VSSOSC	پایه زمین برای اسیلاتور
OSC1	ورودی اسیلاتور
OSC2	خروجی اسیلاتور
VDDOSC	پایه تغذیه مثبت 3.3v برای اسیلاتور
LEDB	پایه درایور LEDB
LEDA	پایه درایور LEDA

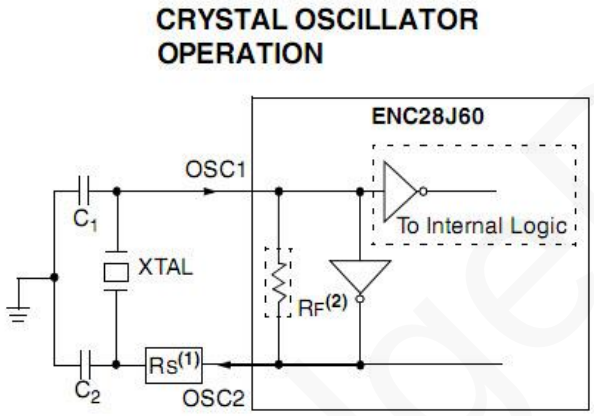
VDD	پایه تغذیه مثبت 3.3v

جدول ۷-۱

۷-٦. واحد اسیلاتور

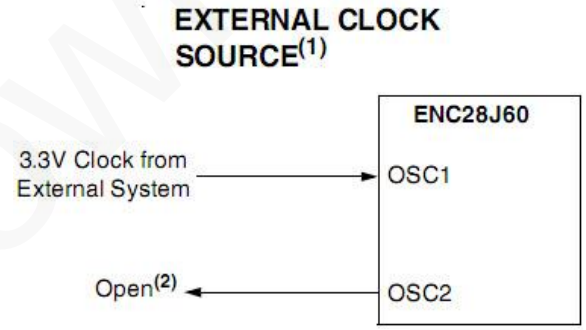
برای تامین کلاک این تراشه به دو روش می توان عمل کرد :

روش اول : استفاده از کریستال 25MHz



شکل۷-۴

روش دوم : اعمال منبع کلاک خارجی به پایه ی OSC1 با دامنه 3.3v ، پایه OSC2 باز



شکل۷-۵

۷-٦-۱. تایمر زمان شروع (Oscillator Start-Up Timer)

پس از اتصال تغذیه یا ریست و خارج شدن از حالت power-down، مدت زمانی معادل 7500 سیکل از کلاک (300us) برای آماده سازی اسیلاتور و لایه ی فیزیکی زمان لازم است. در این مدت زمان

رجیسترها و بافرها قابلیت خواندن یا نوشتن دارند ولی در این مدت زمان نباید قابلیت فرستنده فعال شود (با یک کردن بیت ECON1.TXRTS) و نباید بخش گیرنده نیز فعال شود (با یک کردن بیت ECON1.RXEN) و دسترسی به رجیسترهای MAC ، MII و PHY وجود ندارد.

پس از پایان یافتن این مدت زمان، بیت CLKRDY در رجیستر ESTAT یک خواهد شد، لذا با بررسی این بیت (به روش سرکشی یا polling) می توان از به اتمام رسیدن این مدت زمان مطلع شد.

۷-۷. پایه CLKOUT

از کلاک تولید شده در این پایه می توان برای تامین کلاک میکروکنترلر سیستم یا بخش های دیگر استفاده نمود. همچنین توسط پیش تقسیم کننده های 1,2,3,4,8 می توان فرکانس خروجی این پایه را کاهش داد. برای استفاده و تنظیم این پایه باید رجیستر ECOCON تنظیم شود.

زمانی که کلاک این پایه توسط رجیستر ECOCON غیر فعال شود، سطح این پایه به صفر کاهش پیدا می کند.

فرکانس پیش فرض این پایه 6.25MHz (پیش تقسیم کننده 4)می باشد. در حالت power-down نیز این پایه کلاک تولید می کند.

شکل ۷-۶ مقادیر مختلف این رجیستر را نشان می دهد. همان طور که مشاهده می شود تنها سه بیت کم ارزش تر این رجیستر برای تنظیم فرکانس و فعال کردن این پایه مورد استفاده قرار می گیرد.

در حالتی که هر سه بیت صفر باشند، کلاک این پایه غیرفعال است. مقدار 11x که معادل دو حالت 111 و 110 می باشد نیز نباید استفاده شود.

REGISTER 2-1: ECOCON: CLOCK OUTPUT CONTROL REGISTER

U-0	U-0	U-0	U-0	U-0	R/W-1	R/W-0	R/W-0
—	—	—	—	—	COCON2	COCON1	COCON0
bit 7							bit 0

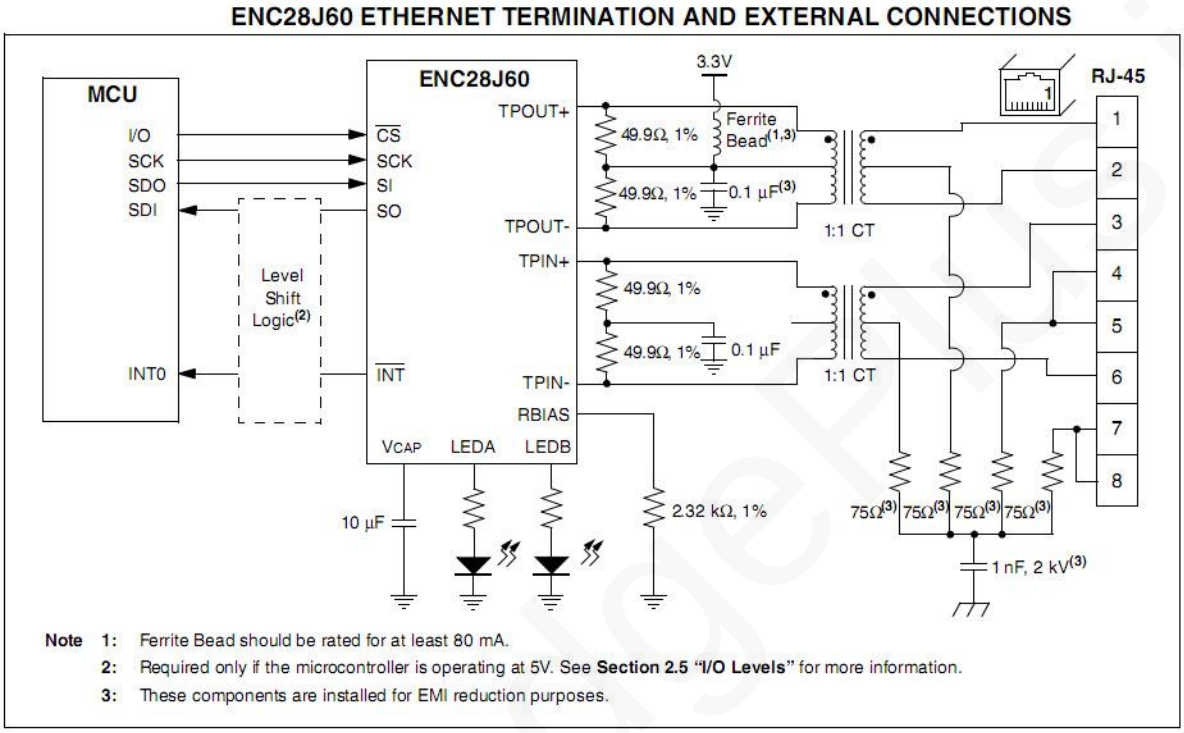
bit 7-3 **Unimplemented**: Read as '0'

bit 2-0 **COCON2:COCON0**: Clock Output Configuration bits
11x = Reserved for factory test. Do not use. Glitch prevention not assured.
101 = CLKOUT outputs main clock divided by 8 (3.125 MHz)
100 = CLKOUT outputs main clock divided by 4 (6.25 MHz)
011 = CLKOUT outputs main clock divided by 3 (8.333333 MHz)
010 = CLKOUT outputs main clock divided by 2 (12.5 MHz)
001 = CLKOUT outputs main clock divided by 1 (25 MHz)
000 = CLKOUT is disabled. The pin is driven low.

شکل۷-۶

۷-۸. مدارات جانبی

برای استفاده از این تراشه و تکمیل سخت افزار لازم برای اتصال به شبکه، چند قطعه خارجی باید مطابق شکل زیر به این تراشه متصل گردد.



شکل۷-۷

مقاومت 2.32K که به پایه RBIASE متصل است باید تاحد امکان به این پایه نزدیک باشد و مسیر سیگنالی در کنار آن وجود نداشته باشد (در طراحی PCB به این نکته باید دقت شود) و ترجیحا از نوع SMD باشد.

عدد نوشته شده به همراه علامت % کنار مقاومت ها تلرانس یا درصد خطای پیشنهادی برای مقاومت ها می باشد.

خازن 10uF که به پایه ی VCAP متصل شده است باید ESR (مقاومت معادل سری) کمی داشته باشد.

فریت بید های متصل شده باید دارای مقدار حداقل جریان 80mA باشند.

از مدار Level Shift Logic زمانی که تغذیه میکروکنترلر 5v است استفاده می شود تا سطح سیگنال های SPI را به 3.3v کاهش بدهد.

بر روی پایه های TPIN+/TPIN- و TPOUT+/TPOUT- باید از ترانسفورماتور پالس از نوع سر

وسط دار با مقادیر مناسب برای شبکه اترنت استفاده شود. این مقادیر که طبق استاندارد IEEE 802.3 تعریف شده است مطابق جدول زیرمی باشد.

REQUIREMENTS FOR EXTERNAL MAGNETICS

Parameter	Min	Norm	Max	Units	Conditions
RX Transformer Turns Ratio	—	1:1	—	—	
TX Transformer Turns Ratio	—	1:1	—	—	Transformer Center Tap = 3.3V
Insertion Loss	0.0	0.6	1.1	dB	
Primary Inductance	350	—	—	µH	8 mA bias
Transformer Isolation	—	1.5	—	kV	
Differential to Common Mode Rejection	40	—	—	dB	0.1 to 10 MHz
Return Loss	-16	—	—	dB	

جدول ۷-۲

به عنوان مثال مطابق جدول بالا، نسبت تعداد دور سیم پیچی هم برای ترانسفورماتور فرستنده (ترانسفورماتور متصل شده به پایه های –TPOUT+/TPOUT) و هم برای ترانسفورماتور گیرنده (ترانسفورماتور متصل شده به پایه های –TPIN+/TPIN) برابر 1/1 است، یعنی تعداد دور اولیه و ثانویه یکسان است و در نتیجه ولتاژ ورودی و خروجی نیز یکسان است و ترانسفورماتور ولتاژ را تغییری نمی دهد. در ستون های بعدی جزئیات بیشتری مانند تلفات و مقدار اندوکتانس ترانسفورماتور ذکر شده است.

همان طور که در فصل قبلی توضیح داده شد، برخی از کانکتور های RJ-45 که استفاده از آنها همراه با کابل های CAT در شبکه های اترنت رایج است، دارای ترانسفورماتور و فیلترهای لازم می باشند و نیازی به استفاده از این قطعات به طور خارجی وجود ندارد.

نکته ۱ : همه ی پایه های تغذیه تراشه باید به منبع تغذیه یکسانی متصل شوند. همچنین همه ی پایه های زمین تراشه نیز باید به زمین یکسانی متصل شوند. همچنین استفاده از خازن های 0.1uF برای هر پایه ی VDD و VSS به عنوان فیلتر تغذیه در نزدیک ترین محل به پایه ها پیشنهاد می شود.

نکته ۲ : قطعاتی که با شماره (3) در شماتیک مشخص شده اند، برای کاهش تداخلات الکترومغناطیسی و افزایش مصونیت مدار در برابر نویز قرار داده شده اند و وجود آنها برای عملکرد مدار در شرایط عادی و کاربردهای آزمایشگاهی ضرورتی ندارد.

مطابق شکل بالا، پایه های ترانسفورماتور باید به کابل CAT و مقاومت های ۷۵ اهمی متصل شود.

پایه های 1 و 2 کانکتور RJ-45 به خروجی ترانسفورماتور بخش فرستنده و پایه های 3 و 6 این کانکتور به خروجی ترانسفورماتور بخش گیرنده متصل شده است. سرهای وسط خروجی ترانسفورماتور ها از طریق مقاومت های ۷۵ اهمی و خازن 1nF به زمین متصل شده اند.

همچنین پایه های 4 و 5 کانکتور به هم و پایه های 7 و 8 کانکتور به هم متصل شده و مشابه سرهای

وسط خروجی ترانسفورماتورها، از طریق مقاومت های ۷۵ اهمی و خازن 1nF به زمین متصل شده اند.

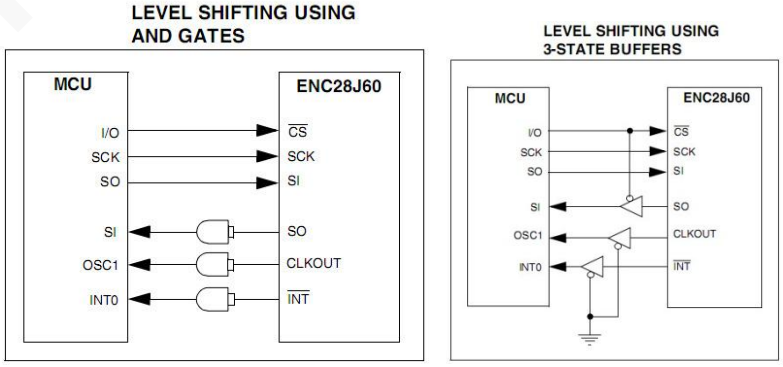
نکته ۳ : با توجه به ولتاژ خازن 1nF که برابر ۲ کیلو ولت است، مدل مناسب برای این خازن نوع پلی استر می باشد.

نکته ۴ : با توجه به این که جریان پایه های متصل شده به کانکتور RJ-45 نسبتا بالا می باشد، هنگام طراحی PCB باید سعی شود خطوط خروجی ترانسفورماتور تا کانکتور RJ-45 تاحد امکان کوتاه و ضخیم باشند (برای کاهش مقاومت مسیر)، اگر امکان کوتاه بودن مسیر وجود ندارد، باید مسیرها به شکلی رسم شوند که مقاومت مشخصه (characteristic impedance) مسیر برابر ۱۰۰ اهم شود.

۷-۹. سطوح منطقی پایه های I/O

علارغم این که ولتاژ تغذیه ENC28J60 برابر 3.3v است، اما پایه های مربوط به ارتباط SPI شامل CS، SCK و SI قابلیت تحمل ولتاژ 5v را دارند، لذا اگر از میکروکنترلری استفاده می کنیم که تغذیه آن 5v است، برای تبدیل ولتاژ این پایه های به 3.3v نیاز به مداری جانبی وجود ندارد ولی از آنجایی که ورودی های میکروکنترلر معمولا باید در محدوده ی تغذیه یعنی 5v باشد تا به عنوان سطح منطقی یک شناخته شود، بدین منظور باید برای پایه های ورودی میکروکنترلر از تراشه مناسب برای تبدیل سطح 3.3v به 5v استفاده کرد. به عنوان مثال مطابق شکل های زیر می توان از گیت های AND سری 74HCT08 یا از بافر های سه حالته 74ACT125 برای اتصال پایه هایی که از تراشه ENC28J60 قرار است به میکروکنترلر وصل شود استفاده نمود.

مزیت استفاده از بافر سه حالته 74ACT125 در کاربردهایی است که به پایه های SPI میکروکنترلر قرار است پایه های دیگری نیز وصل شود و در کاربردهای دیگری استفاده شوند. (به عنوان مثال برای پروگرام کردن میکروکنترلر)



شکل۷-۸

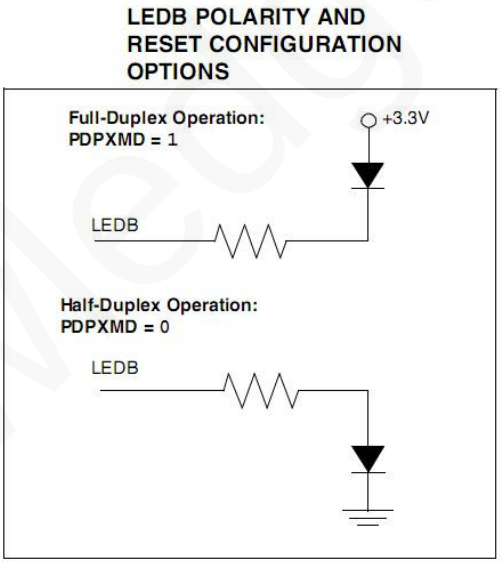
۷-۱۰. پیکربندی LED های وضعیت شبکه

پایه های LEDA و LEDB دارای قابلیت تشخیص خودکار پلاریته می باشند، بدین معنا که LED ها را از هر طرفی که وصل کنیم، تراشه به طور خودکار تشخیص می دهد که کدام پایه متصل شده است. پس از ریست، تراشه به شکل خودکار این تشخیص را انجام می دهد اما اگر در حین وصل بودن تغذیه پایه های LED را تعویض کنیم، برای تشخیص پایه ها باید یک بار تراشه را ریست کنیم.

وضعیت پیشفرض LED ها توسط رجیستر PHLCON مشخص می شود.

LEDB دارای این ویژگی می باشد که اگر پایه آند LED به آن متصل باشد، بیت PDPXMD در رجیستر PHCON1 برابر صفر می شود و وضعیت عملکرد لایه ی فیزیکی اترنت روی حالت half-duplex تنظیم می شود واگر کاتد LED به این پایه متصل باشد، این بیت یک شده و روی حالت full-duplex تنظیم می شود. شکل زیر این دو وضعیت را نشان می دهد.

اگر LED به این پایه متصل نشود، بیت PDPXMD دارای وضعیت نامشخصی خواهد بود.



شکل۷-۹

رجیستر PHLCON یک رجیستر دو بایتی برای تنظیمات مربوط به LEDA و LEDB می باشد.

REGISTER 2-2: PHLCON: PHY MODULE LED CONTROL REGISTER

R/W-0	R/W-0	R/W-1	R/W-1	R/W-0	R/W-1	R/W-0	R/W-0
r	r	r	r	LACFG3	LACFG2	LACFG1	LACFG0
bit 15							bit 8

R/W-0	R/W-0	R/W-1	R/W-0	R/W-0	R/W-0	R/W-1	R/W-x
LBCFG3	LBCFG2	LBCFG1	LBCFG0	LFRQ1	LFRQ0	STRCH	r
bit 7							bit 0

شکل۷-۱۰

بیت های LACFG0 ، LACFG1 ، LACFG2 و LACFG3 مربوط به تنظمات پیکربندی LEDA و بیت های LBCFG0 ، LBCFG1 ، LBCFG2 و LBCFG3 مربوط به تنظمات پیکربندی LEDB می باشد. توسط این بیت ها می توان تنظیماتی مانند روشن شدن LED در شرایط ارسال، دریافت، بروز تصادم، روشن بودن دائم، خاموش بودن دائم یا چشمک زدن و ... را تنظیم کرد. به عنوان مثال با نوشتن مقدار 0100 در این چهار بیت، LED وضعیت Link یا اتصال به شبکه را نشان خواهد داد.

بیت های 0 ، 12 ، 13 ، 14 و 15 کاربردی ندارند.

توسط بیت های LFRQ0 و LFRQ1 می توان مدت زمان روشن بودن هر دو LED را طبق جدول زیر در سه حالت تنظیم کرد. (نوشتن عدد 11 در این دو بیت کاربردی ندارد)

LED BLINK STRETCH LENGTH

Stretch Length	Typical Stretch (ms)
TNSTRCH (normal)	40
TMSTRCH (medium)	70
TLSTRCH (long)	140

جدول ۷-۳

برای استفاده از این قابلیت باید ابتدا در بیت STRCH مقدار یک نوشته شود.

مقادیر مختلف برای تنظیم شرایط مختلف LED ها که بخشی از آن در این قسمت توضیح داده شد، در جدول مربوط به بخش LED Configuration دیتاشیت آمده است که برای تنظیم دقیق وضعیت این دو LED می توان به این بخش مراجعه کرد.

۷-۱۱. مدیریت حافظه

حافظه این تراشه به سه قسمت تقسیم شده است :

رجیسترهای کنترلی : برای تنظیم و پیکربندی و همچنین بررسی وضعیت تراشه به کار میروند.

بافر اترنت : شامل بافر فرستنده و گیرنده برای نگه داری اطلاعات، اندازه این بخش قابل برنامه ریزی است.

رجیسترهای بخش فیزیکی (PHY) : شامل تنظیمات بخش فیزیکی است. دسترسی به این رجیسترها برخلاف رجیسترهای کنترلی و بافر اترنت، از طریق دستورات SPI امکان پذیرد نیست و از طریق بخش MIIM است. در بخش های بعدی توضیحات بیشتری در مورد دسترسی به این رجیسترها خواهیم داد.

۷-۱۲. رجیسترهای کنترلی

رجیسترها ی کنترلی مطابق جدول ۷-۴ به ۴ بخش (bank) تقسیم بندی شده اند و برای دسترسی به هر بخش باید مقدار مناسب در بیت های BSEL0 و BSEL1 در رجیسترهای ECON1 نوشته شود.

مطابق شکل، ۵ رجیستر آخر هر بخش (آدرس 1Bh تا 1Fh) شامل رجیسترهای مشترک به نام های EIE ، EIR ، ESTAT ، ECON2 و ECON1 می باشد. این رجیسترها برای بررسی وضعیت های مختلف تراشه مورد استفاده قرار می گیرند که برنامه نویس می تواند در شرایط مختلف برای اطلاع از وضعیت تراشه، مقدار آنها را بخواند. به دلیل کاربرد زیاد این رجیسترها، به صورت مشترک در تمام بخش ها وجود دارند که برای دسترسی به آنها برنامه نویس لازم نباشد که بخش فعلی را توسط بیت های BSEL0 و BSEL1 تغییر دهد تا به این رجیسترها دسترسی پیدا کند.

رجیسترهایی که با حرف E شروع می شوند مربوط به تنظیمات بخش اترنت هستند، رجیسترهایی که با حرف M شروع می شوند مربوط به تنظیمات بخش MAC هستند و رجیسترهایی که با حرف M شروع می شوند مربوط به تنظیمات بخش MII هستند.

ENC28J60 CONTROL REGISTER MAP

Bank 0 Address	Name	Bank 1 Address	Name	Bank 2 Address	Name	Bank 3 Address	Name
00h	ERDPTL	00h	EHT0	00h	MACON1	00h	MAADR5
01h	ERDPTH	01h	EHT1	01h	Reserved	01h	MAADR6
02h	EWRPTL	02h	EHT2	02h	MACON3	02h	MAADR3
03h	EWRPTH	03h	EHT3	03h	MACON4	03h	MAADR4
04h	ETXSTL	04h	EHT4	04h	MABBIPG	04h	MAADR1
05h	ETXSTH	05h	EHT5	05h	—	05h	MAADR2
06h	ETXNDL	06h	EHT6	06h	MAIPGL	06h	EBSTSD
07h	ETXNDH	07h	EHT7	07h	MAIPGH	07h	EBSTCON
08h	ERXSTL	08h	EPMM0	08h	MACLCON1	08h	EBSTCSL
09h	ERXSTH	09h	EPMM1	09h	MACLCON2	09h	EBSTCSH
0Ah	ERXNDL	0Ah	EPMM2	0Ah	MAMXFLL	0Ah	MISTAT
0Bh	ERXNDH	0Bh	EPMM3	0Bh	MAMXFLH	0Bh	—
0Ch	ERXRDPTL	0Ch	EPMM4	0Ch	Reserved	0Ch	—
0Dh	ERXRDPTH	0Dh	EPMM5	0Dh	Reserved	0Dh	—
0Eh	ERXWRPTL	0Eh	EPMM6	0Eh	Reserved	0Eh	—
0Fh	ERXWRPTH	0Fh	EPMM7	0Fh	—	0Fh	—
10h	EDMASTL	10h	EPMCSL	10h	Reserved	10h	—
11h	EDMASTH	11h	EPMCSH	11h	Reserved	11h	—
12h	EDMANDL	12h	—	12h	MICMD	12h	EREVID
13h	EDMANDH	13h	—	13h	—	13h	—
14h	EDMADSTL	14h	EPMOL	14h	MIREGADR	14h	—
15h	EDMADSTH	15h	EPMOH	15h	Reserved	15h	ECOCON
16h	EDMACSL	16h	Reserved	16h	MIWRL	16h	Reserved
17h	EDMACSH	17h	Reserved	17h	MIWRH	17h	EFLOCON
18h	—	18h	ERXFCON	18h	MIRDL	18h	EPAUSL
19h	—	19h	EPKTCNT	19h	MIRDH	19h	EPAUSH
1Ah	Reserved	1Ah	Reserved	1Ah	Reserved	1Ah	Reserved
1Bh	EIE	1Bh	EIE	1Bh	EIE	1Bh	EIE
1Ch	EIR	1Ch	EIR	1Ch	EIR	1Ch	EIR
1Dh	ESTAT	1Dh	ESTAT	1Dh	ESTAT	1Dh	ESTAT
1Eh	ECON2	1Eh	ECON2	1Eh	ECON2	1Eh	ECON2
1Fh	ECON1	1Fh	ECON1	1Fh	ECON1	1Fh	ECON1

جدول ۷-۴

۷-۱۳. رجیستر ECON1

همان طورکه در بخش رجیسترهای کنترلی اشاره شد، برای تغییر بخش به منظور دسترسی به رجیسترهای کنترلی مختلف، نیاز به مقدار دهی مناسب دو بیت BSEL0 و BSEL1 در این رجیستر می باشد. از دیگر بیت های این رجیستر شامل فعال سازی دریافت، درخواست ارسال و کنترل واحد DMA است.

ECON1: ETHERNET CONTROL REGISTER 1

R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0
TXRST	RXRST	DMAST	CSUMEN	TXRTS	RXEN	BSEL1	BSEL0
bit 7							bit 0

شکل۷-۱۱

ار بیت های مهم این رجستر عبارت اند از :

- TXRTS : ایجاد درخواست برای ارسال بیت
- RXEN : فعال سازی دریافت بسته های داده
- BSEL1:BSEL0: انتخاب بخش (بخش های رجیسترهای کنترلی) :
  - 11 : انتخاب بخش ۳
  - 10 : انتخاب بخش ۲
  - 01 : انتخاب بخش 1
  - 00 : انتخاب بخش ۰

۷-۱۴. حافظه بافر اترنت

شامل ۸ کیلوبایت حافظه برای فرستنده و گیرنده که مکان و ظرفیت هر بخش توسط میکروکنترلر قابل برنامه ریزی است.

مقدار ظرفیت بخش بافر گیرنده را می توان توسط رجیسترهای ERXSTH:ERXSTL و ERXNDH:ERXNDL تعیین کرد.

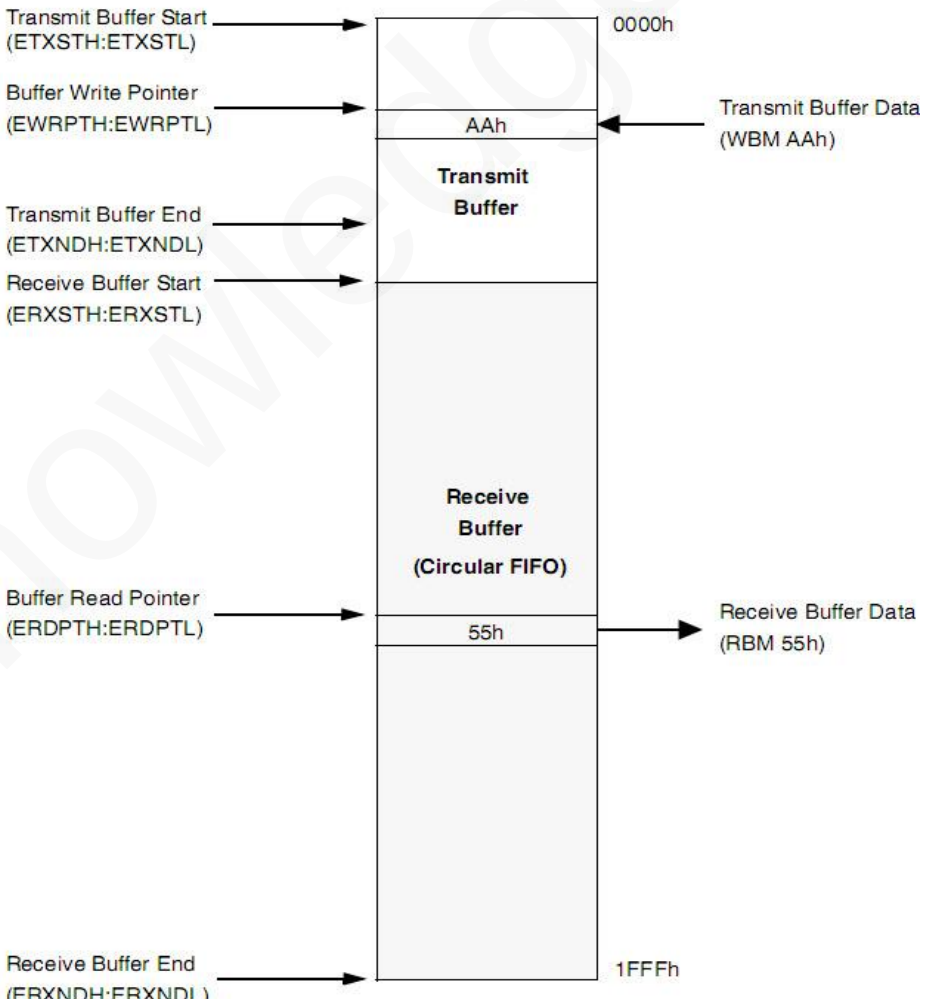
بافر بخش گیرنده به صورت FIFO چرخشی است. آدرس نوشته شده در رجیستر اشاره گر ERXRDPT حد ظرفیت بافر گیرنده را مشخص می کند. اطلاعات دریافت شده تا مرز این آدرس در بافر قرار می گیرند. اگر اطلاعات جدیدی دریافت شوند و این ظرفیت به اتمام رسیده باشد، اطلاعات دریافت شده ذخیره نمی گردد و از دست می رود. برای جلوگیری از این مسئله میکروکنترلر باید در هنگام دریافت اطلاعات، دائما (به صورت متناوب مثلا هر 1ms) مقدار این اشاره گر ر ا افزایش دهد.

ظرفیت بافر گیرنده توسط رجیسترهای ذکر شده در بالا مشخص می شود. بخش باقی ماند از ۸

کیلوبایت بافر اترنت، به بافر فرستنده تخصیص می یابد. (علارغم بافر گیرنده آدرس آن توسط رجیسترها مشخص نمی شود)

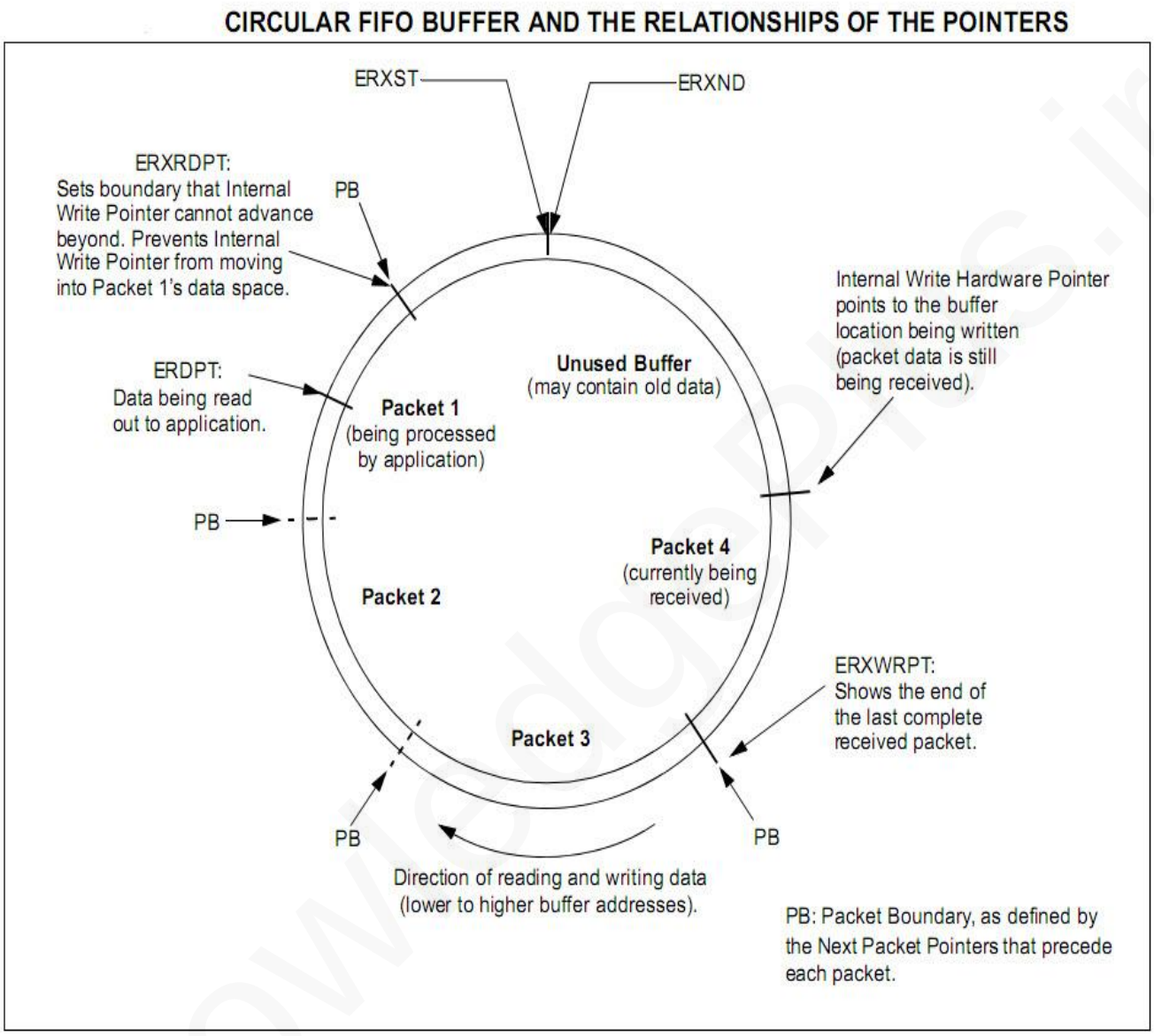
محل قرار گیری هر بسته داده (که قرار است توسط واحد فرستنده به شبکه ارسال شود) توسط خطوط برنامه و مقداردهی مناسب به اشاره گرهای ETXST و ETXND تعیین می شود. در واقع هر وقت که خواستیم بسته ای را در شبکه ارسال کنیم، باید توسط دادن آدرس بخشی از بافر فرستنده به این دو اشاره گر، بسته را در آن بخش از بافر فرستنده قرار دهیم. نکته ای که وجود دارد این است که ممکن است برنامه نویس به اشتباه آدرس بخشی از بافر گیرنده را در این دو اشاره گر تعیین کند، در اینصورت سخت افزار هیچ بررسی به این منظور انجام نمی دهد و نمیتوان مثلا از بررسی یک بیت به این مشکل پی برد؛ بنابراین باید به این مسئله دقت شود که برای احتیاط مقدار اشاره گر ETXND به حافظه تعیین شده برای بافر گیرنده خیلی نزدیک نشود.

شکل زیر بافر اترنت و حافظه های گیرنده و فرستنده در آن را نشان می دهد.



شکل۷-۱۲

برای درک بهتر ساختار FIFO حافظه می توان از شکل زیر استفاده کرد.



شکل۷-۱۳

۷-۱۵. رجیسترهای لایه فیزیکی (PHY)

تمام رجیسترهای این بخش ۱۶ بیتی هستند. علارغم رجیسترهای کنترلی یا حافظه ی بافر که در قسمت قبل توضیح داده شد، دسترسی به رجیسترهای این بخش توسط دستورات SPI امکان پذیر نیست و باید توسط رجیسترهای کنترلی MAC که در واقع یک نوع MIIM را تشکیل می دهند صورت گیرد.

مراحل خواندن رجیسترهای بخش PHY عبارت است از :

- نوشتن آدرس رجیستر مورد نظر در رجیستر MIREGADR
- نوشتن یک در بیت MICMD.MIIRD (رجیستر MICMD و بیت MIIRD)، بیت MISTAT.BUSY به نشانه ی شروع عملیات خواندن، یک می شود.
- خواندن رجیستر MISTAT.BUSY برای اطمینان از اتمام عملیات خواندن (صفر شدن این بیت به نشانه ی اتمام عملیات است، معمولا زمانی حدود 10.24us)
- نوشتن صفر در MICMD.MIIRD
- خواندن رجیسترهای MIRDL و MIRDH (ترتیب خواندن اهمیتی ندارد)

نکته : خواندن رجیسترها به صورت ۱۶ بیتی صورت می پذیرد.

مراحل نوشتن در رجیسترهای بخش PHY عبارت است از :

- نوشتن آدرس رجیستر مورد نظر در رجیستر MIREGADR
- نوشتن یک در بیت MICMD.MIIRD (رجیستر MICMD و بیت MIIRD)، بیت MISTAT.BUSY به نشانه ی شروع عملیات خواندن، یک می شود.
- نوشتن ۸ بیت پایین تر اطلاعات در رجیستر MIWRL
- نوشتن ۸ بیت پر ارزش تر در رجیستر MIWRH ، با نوشتن در این رجیستر عملیات نوشتن به صورت خودکار شروع می شود لذا حتما باید عملیات نوشتن در این رجیستر بعد از نوشتن در رجیستر MIWRL صورت پذیرد، بیت MISTAT.BUSY با شروع عملیات نوشتن یک می شود.
- خواندن رجیستر MISTAT.BUSY برای اطمینان از اتمام عملیات خواندن (صفر شدن این بیت به نشانه ی اتمام عملیات است، معمولا زمانی حدود 10.24us)

نکته : نوشتن در رجیسترها به صورت ۱۶ بیتی صورت می پذیرد.

۷-۱۵-۱. پویش (Scan) رجیسترهای بخش فیزیکی

در مواقعی که نیاز است وضعیت رجیسترهای PHY به صورت متناوب بررسی شود، می توان عملیات خواندن را به شکل زیر به صورت پشت سر هم انجام داد که این مطلب باعث کاهش خطوط برنامه می شود :

- نوشتن آدرس رجیستر مورد نظر در رجیستر MIREGADR
- نوشتن یک در بیت MICMD.MIISCAN ، بیت MISTAT.BUSY به نشانه ی شروع عملیات خواندن یک می شود. علمیات خواندن حدود 10.24us زمان خواهد برد، خواندن های بعدی به صورت پی در پی و خودکار صورت می پذیرد تا این که کاربر آن را لغو کند. برای اطلاع از پایان اولین مرحله از خواندن، می توان بیت MISTAT.NVALID را بررسی کرد.

نکته : در هنگام انجام این عملیات، نباید بر روی رجیستر MIWRH نوشته شود یا عملیات MIIRD انجام شود.

- با نوشتن صفر در بیت MICMD.MIISCAN و بررسی بیت MISTAT.BUSY برای صفر شدن، عملیات خواندن های پی در پی به پایان می رسد.

نکته : پس از یک کردن بیت MIISCAN ، رجیسترهای MIRDL و MIRDH هر 10.24us به صورت خودکار با مقادیر جدید خوانده شده آپدیت می شوند. هیچ مکانیزمی برای فهمیدن این مسئله که رجیسترهای MIRD چه زمانی آپدیت می شوند وجود ندارد. از آنجایی که در هر لحظه میکروکنترلر تنها میتواند یک رجیستر MII را از طریق دستورات SPI بخواند، لذا مقدار خوانده شده در هر لحظه به معنای مقدار این رجیسترها در آن لحظه نیست!

۷-۱۵-۲. رجیستر MICMD

REGISTER 3-3: MICMD: MII COMMAND REGISTER

U-0	U-0	U-0	U-0	U-0	U-0	R/W-0	R/W-0
—	—	—	—	—	—	MIISCAN	MIIRD
bit 7							bit 0

شکل۷-۱۴

MIISCAN : فعال سازی قابلیت خواندن های پی در پی با نوشتن یک در این بیت

MIIRD : فعال سازی قابلیت خواندن رجیسترهای PHY (از طریق MII) با نوشتن یک در این بیت

۷-۱۵-۳. رجیستر MISTAT

REGISTER 3-4: MISTAT: MII STATUS REGISTER

U-0	U-0	U-0	U-0	R-0	R-0	R-0	R-0
—	—	—	—	r	NVALID	SCAN	BUSY
bit 7							bit 0

شکل۷-۱۵

NVALID : یک بودن این بیت نشان دهنده ی صحیح نبودن اطلاعات رجیستر MIRD می باشد.
(برای خواندن رجیستر MIRD باید منتظر ماند تا این بیت صفر شود)

SCAN : یک بودن این بیت نشان دهنده ی در حال اجرا بودن عملیات خواندن های پی در پی می باشد.

BUSY : یک بودن این بیت نشان دهنده ی در حال اجرا بودن عملیات خواندن یا نوشتن در یکی از رجیسترهای PHY است.

۷-۱۶. رجیسترهای PHSTAT

رجیسترهای فقط خواندنی (Read-only) PHSTAT1 و PHSTAT2 نشان دهنده ی وضعیت بخش فیزیکی (PHY) به ویژه اتصال به شبکه می باشند.

صفر شدن بیت LLSTAT از رجیستر PHSTAT1 نشان دهنده ی قطع شدن ارتباط از شبکه است. اگر از وقفه تغییر وضعیت اتصال (link change) به شبکه استفاده نمی کنیم، می توانیم با روش سرکشی (polling) این بیت در خطوط نرم افزار، از وضعیت اتصال به شبکه با خبر شویم.

همچنین یک شدن بیت jabber در این رجیستر نشان دهنده ی اتفاق افتادن شرایط jabber در شبکه است. (وضعیتی است که داده های ارسال شده توسط یک گره در شبکه بیش از حد استاندارد بوده و اجازه ارتباط با شبکه به گره های دیگر داده نمی شود)

رجیستر PHSTAT2 حاوی بیت های وضعیت اتصال شبکه و همچنین وضعیت ارسال و دریافت داده ها می باشد.

۷-۱۶-۱. رجیستر PHSTAT1

REGISTER 3-5: PHSTAT1: PHYSICAL LAYER STATUS REGISTER 1

U-0	U-0	U-0	R-1	R-1	U-0	U-0	U-0
—	—	—	PFDPX	PHDPX	—	—	—
bit 15							bit 8

U-0	U-0	U-0	U-0	U-0	R/LL-0	R/LH-0	U-0
—	—	—	—	—	LLSTAT	JBSTAT	—
bit 7							bit 0

شکل۷-۱۶

از بیت های مهم این رجیستر عبارت اند از :

- LLSTAT : یک بودن این بیت نشان دهنده ی برقرار بودن اتصال به شبکه است.
- JBSTAT : یک بودن این بیت نشان دهنده ی رخ دادن شرایط Jabber در شبکه است.

۷-۱۶-۲. رجیستر PHSTAT2

REGISTER 3-6: PHSTAT2: PHYSICAL LAYER STATUS REGISTER 2

U-0	U-0	R-0	R-0	R-0	R-0	R-x	U-0
—	—	TXSTAT	RXSTAT	COLSTAT	LSTAT	DPXSTAT(1)	—
bit 15							bit 8

U-0	U-0	R-0	U-0	U-0	U-0	U-0	U-0
—	—	PLRITY	—	—	—	—	—
bit 7							bit 0

شکل۷-۱۷

از بیت های مهم این رجستر عبارت اند از :

- TXSTAT : یک بودن این بیت نشان دهنده ی در حال اجرا بودن عملیات ارسال است.
- RXSTAT : یک بودن این بیت نشان دهنده ی در حال اجرا بودن عملیات دریافت است.
- COLSTAT : یک بودن این بیت نشان دهنده ی اتفاق افتادن تصادم در شبکه است.
- LSTAT : یک بودن این بیت نشان دهنده ی برقرار بودن اتصال به شبکه است.
- DPXSTAT : یک بودن این بیت نشان دهنده ی ارتباط در مد full-duplex (یک بودن بیت هشتم رجیستر PHCON1) و صفر بودن این بیت نشان دهنده ی ارتباط در مد half- duplex (صفر بودن بیت هشتم رجیستر PHCON1) می باشد.
- PLRITY : یک بودن این بیت نشان دهنده ی جا به جایی پلاریته ی سیگنال های TPIN- و TPIN+ می باشد و صفر بودن نشانه ی صحیح بودن پلاریته و عوض نکردن جای این دو سیگنال است.

نکته : همان طور که در بخش های قبلی نیز اشاره شد، مقدار پیشفرض دوطرفه یا یک طرفه بودن ارتباط (نوع Duplex) به جهت LED متصل شده به پایه ی LEDB بستگی دارد.

۷-۱۷. رابط SPI

این تراشه تنها حالت 00 در SPI را پشتیبانی می کند و خط SCK در حالت بی کار (Idle) باید صفر باشد. همچنین پایه CS هنگام ارسال و دریافت داده باید صفر باشد.

دستوراتی که باید میکروکنترلر به تراشه بدهد از طریق دستورات SPI که در جدول زیر مشاهده می شود صورت می پذیرد. به طور کلی ۷ دستور وجود دارد.

بایت اولی که باید برای تراشه از طریق ارتباط SPI ارسال شود شامل دو بخش است. بخش اول شامل سه بیت به نام Opcode (مخفف operational code یا کد عملیاتی) می باشد که مشخص کننده ی نوع دستور است. در بخش دوم به نام Argument باید به جای حروف a آدرس رجیستر مورد نظر که دستور را برای آن اجرا می کنیم (مثلا خواندن یک رجیستر یا نوشتن در یک رجیستر) نوشته شود.

در بایت دوم N/A (Not/Assigned) به معنای عدم لزوم به استفاده از بایت دوم می باشد و به جای حروف d باید داده ی مورد نظر مرتبط با آن دستور نوشته شود. (به عنوان مثال مقداری که در رجیستر می خواهیم بنویسیم)

SPI INSTRUCTION SET FOR THE ENC28J60

Instruction Name and Mnemonic	Byte 0		Byte 1 and Following
	Opcode	Argument	Data
Read Control Register (RCR)	0 0 0	a a a a a	N/A
Read Buffer Memory (RBM)	0 0 1	1 1 0 1 0	N/A
Write Control Register (WCR)	0 1 0	a a a a a	d d d d d d d d
Write Buffer Memory (WBM)	0 1 1	1 1 0 1 0	d d d d d d d d
Bit Field Set (BFS)	1 0 0	a a a a a	d d d d d d d d
Bit Field Clear (BFC)	1 0 1	a a a a a	d d d d d d d d
System Reset Command (Soft Reset) (SRC)	1 1 1	1 1 1 1 1	N/A

جدول ۷-۵

دستورات جدول بالا عبارت اند از :

- Read Control Register : خواندن یک رجیستر کنترلی
- Read Buffer Memory : خواندن حافظه بافر
- Write Control Register : نوشتن در یک رجیستر کنترلی

- Write Buffer Memory : نوشتن در حافظه بافر
- Bit Field Set : یک کردن یک بیت مشخص
- Bit Field Clear : صفر کردن یک بیت مشخص
- System Reset Command : ریست کردن تراشه به صورت نرم افزاری

در بخش های بعدی به توضیح هریک از این دستورات و نحوه ی استفاده از آنها می پردازیم.

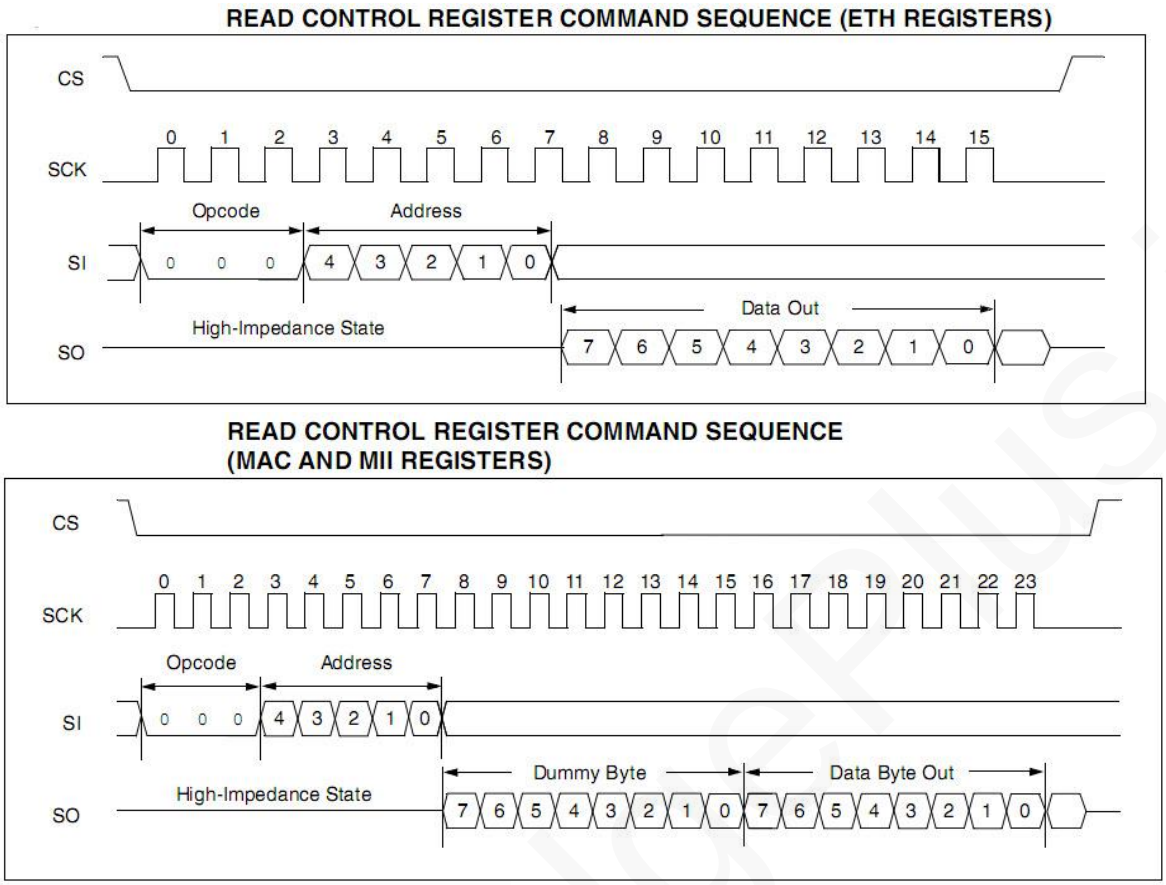
۷-۱۷-۱. دستور READ CONTROL REGISTER

از این دستور برای خواندن هر یک از رجیسترهای اترنت، MAC و MII (برای خواندن رجیسترهای PHY) استفاده می شود.

مراحل استفاده از این دستور عبارت است از :

- صفر کردن پایه ی CS تراشه
- نوشتن opcode مربوط به این دستور (000) و آدرس رجیستر مورد نظر در بخش فعلی (ابتدا باید در بخش رجیستر مورد نظر قرار بگیریم) که می خواهیم مقدار موجود در آن را بخوانیم در بایت اول ارسالی.
- اگر قصد خواندن رجیسترهای اترنت را داریم، بلافاصله پس از ارسال دستور بالا، داده های موجود در این رجیستر از طریق پایه ی SO برای میکروکنترلر ارسال می شود. (به ترتیب بیت های پر ارزش تر (MSB first))
- اگر قصد خواندن رجیسترهای MAC یا MII را داریم، پس از ارسال بایت اول که در بالا توضیح داده شد، ابتدا یک بایت بدون استفاده (dymmy byte) ارسال می شود و در بایت بعدی داده های موجود در این رجیستر از طریق پایه ی SO برای میکروکنترلر ارسال می شود. (به ترتیب بیت های پر ارزش تر (MSB first))
- با یک کردن پایه CS این دستور به پایان می رسد.

شکل ۷-۱۸ فرایند توضیح داده شده را برای رجیسترهای اترنت و رجیسترهای MAC و MII به صورت جداگانه نشان می دهد.



شکل۷-۱۸

۷-۱۷-۲. دستور READ BUFFER MEMORY

توسط این دستور می توان حافظه بافر فرستنده و گیرنده را که مجموعا ۸ بایت است خواند.

نکته : اگر بیت AUTOINC در رجیستر ECON2 برابر یک باشد، با خواندن هر بایت از حافظه، به شکل خودکار آدرس بایت بعدی (آدرس بایت خوانده شده به علاوه یک) در اشاره گر ERDPT قرار می گیرد و لازم به انجام این کار به صورت دستی نیست. کاربرد این ویژگی در مواقعی است که می خواهیم بایت های حافظه را به صورت پشت سر هم بخوانیم. مثلا خواندن داده های دریافت شده در بافر گیرنده به صورت پشت سر هم. ویژگی دیگری که در این بخش وجود دارد این است که اگر اشاره گر به انتهای بافر گیرنده برسد، مجددا به ابتدای بافر گیرنده می آید و این ویژگی برای خواندن دائم بافر گیرنده به منظور دریافت بسته های داده دریافت شده از شبکه بسیار مفید است.

مراحل استفاده از این دستور عبارت است از :

- صفر کردن پایه ی CS تراشه

- نوشتن opcode مربوط به این دستور (001) و عدد ثابت 1Ah
- بلافاصله پس از ارسال دستور بالا، داده های موجود در آدرس اشاره گر ERDPT از طریق پایه ی SO برای میکروکنترلر ارسال می شود. (به ترتیب بیت های پر ارزش تر)
- اگر پس از ارسال بایت اول، پایه ی CS صفر بماند و کلاک توسط میکروکنترلر روی پایه ی SCK تامین شود (که به شکل خودکار نیز چنین است) و اگر طبق توضیحات بالا بیت AUTOINC یک باشد، به صورت پشت سر هم و توسط افزایش آدرس اشاره گر ERDPT به شکل خودکار، داده های موجود در حافظه به سمت میکروکنترلر ارسال می شود.
- با یک کردن پایه CS این دستور به پایان می رسد.

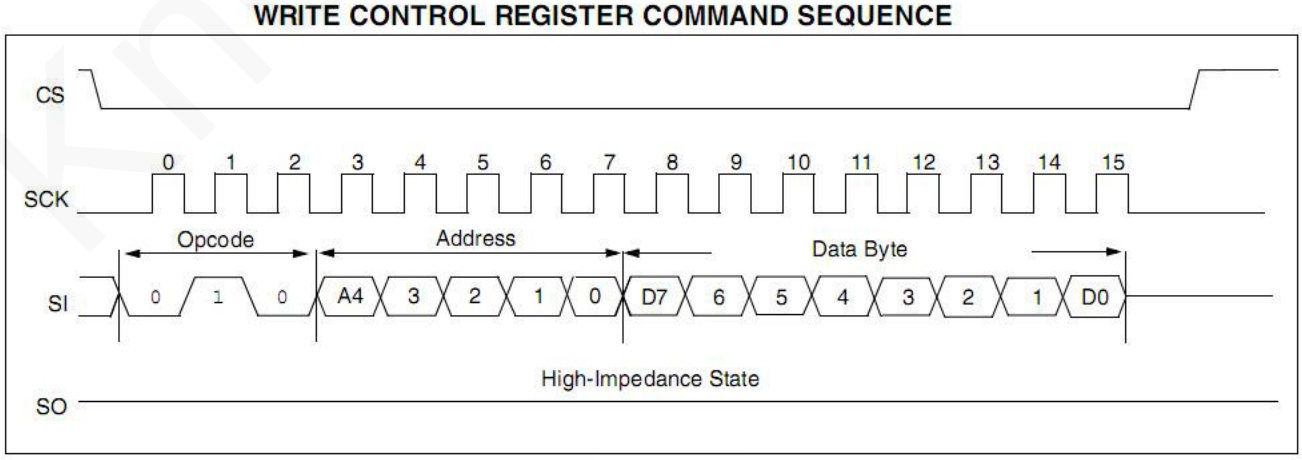
۷-۱۷-۳. دستور WRITE CONTROL REGISTER

توسط این دستور می توان در هر یک از رجیسترهای اترنت، MAC و MII مقدار دلخواه را بنویسیم.
مراحل استفاده از این دستور عبارت است از :

- نوشتن opcode مربوط به این دستور (010) و آدرس رجیستر مورد نظر در بخش فعلی (ابتدا باید در بخش رجیستر مورد نظر قرار بگیریم) که می خواهیم در آن را بنویسیم در بایت اول ارسالی.
- پس از ارسال دستور بالا، مقداری که می خواهیم در رجیستر انتخاب شده بنویسیم را ارسال می کنیم.
- با یک کردن پایه CS این دستور به پایان می رسد.

شکل زیر این دستور را نشان می دهد.

WRITE CONTROL REGISTER COMMAND SEQUENCE



شکل۷-۱۹

۷-۱۷-٤. دستور WRITE BUFFER MEMORY

توسط این دستور می توان در حافظه بافر فرستنده و گیرنده که مجموعا ۸ بایت است، داده های مورد نظر را نوشت.

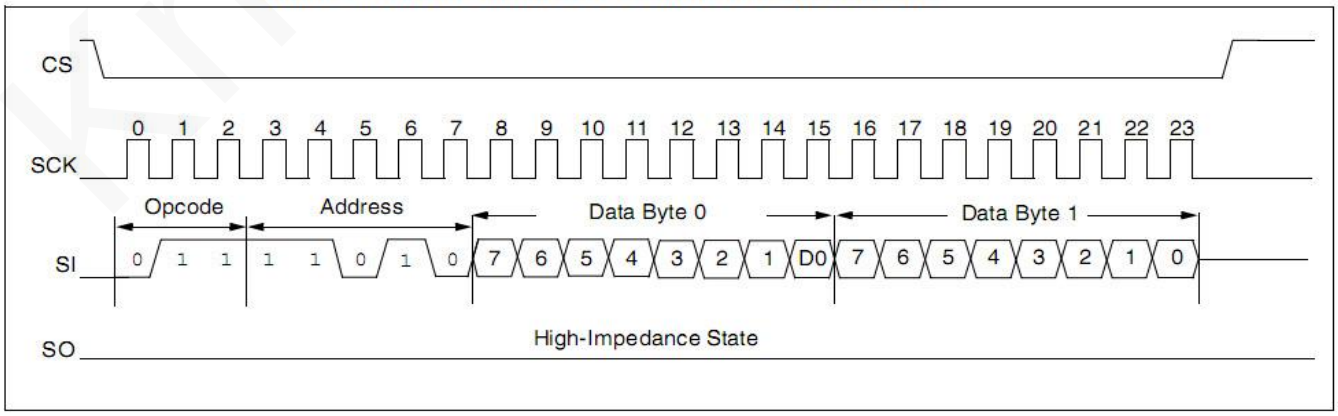
نکته : مشابه توضیحاتی که در بخش خواندن از بافر فرستنده و گیرنده ارائه شد، اگر بیت AUTOINC در رجیستر ECON2 برابر یک باشد، با نوشتن هر بایت از حافظه، به شکل خودکار آدرس بایت بعدی (آدرس بایت نوشته شده به علاوه یک) در اشاره گر EWRPT قرار می گیرد و لازم به انجام این کار به صورت دستی نیست.

مراحل استفاده از این دستور عبارت است از :

- صفر کردن پایه ی CS تراشه
- نوشتن opcode مربوط به این دستور (011) و عدد ثابت 1Ah
- پس از ارسال دستور بالا، باید داده های مورد نظر را ارسال کرد و این داده ها در آدرس اشاره گر EWRPT از حافظه بافر نوشته می شود.
- مشابه دستور خواندن از حافظه بافر که در بخش های قبل توضیح داده شد، اگر پس از ارسال بایت اول، پایه ی CS صفر بماند و کلاک توسط میکروکنترلر روی پایه ی SCK تامین شود (که به شکل خودکار نیز چنین است) و اگر طبق توضیحات بالا بیت AUTOINC یک باشد، به صورت پشت سر هم و توسط افزایش آدرس اشاره گر EWRPT به شکل خودکار، می توان در مکان های پشت سر هم حافظه داده های موجود را نوشت.
- با یک کردن پایه CS این دستور به پایان می رسد.

شکل زیر این دستور را نشان می دهد.

WRITE BUFFER MEMORY COMMAND SEQUENCE



شکل۷-۲۰

۷-۱۷-۵. دستورات BIT FIELD SET ، BIT FIELD CLEAR ، SYSTEM RESET

دستور BIT FIELD SET برای انجام عملیات OR منطقی بر روی یک رجیستر و دستور BIT FIELD CLEAR برای انجام عملیات NAND منطقی بر روی یک رجیستر مورد استفاده قرار می گیرد. (این دو دستور فقط برای رجیسترهای اترنت قابل استفاده هستند)

دستور SYSTEM RESET نیز برای انجام ریست نرم افزاری تراشه مورد استفاده قرار می گیرد.

این دستورات نیز مطابق جدول ارائه شده در بخش اول دستورات این مقاله و توضیحات ارائه شده در دستورات قبلی قابل اجرا هستند.

۷-۱۸. ساختار بسته های اترنت

در مورد ساختار بسته ها یا فریم های شبکه اترنت در فصل چهارم به طور کامل توضیح داده شد. در این بخش به نکاتی که در دیتاشیت این تراشه در مورد ساختار بسته ها آمده اشاره می کنیم.

۷-۱۸-۱. PREAMBLE/START OF FRAME DELIMITER

بخش های preamble ، SFD ، padding و CRC از فریم به صورت خودکار توسط تراشه هنگام ارسال یا دریافت داده ها تولید یا حذف می شوند و نیازی به تنظیمات خاصی توسط میکروکنترلر وجود ندارد. بخش های padding و CRC در بافر دریافت ذخیره می شوند و میتوان در صورت نیاز آنها را مورد بررسی قرار داد.

۷-۱۸-۲. DESTINATION ADDRESS

اگر کم ارزش ترین بیت از اولین بایت یک باشد، آدرس MAC از نوع multicast می باشد،

به عنوان مثال آدرس های 01-00-00-00-F0-00 و 33-45-67-89-AB-CD از نوع multicast هستند اما آدرس های 00-00-00-00-F0-00 و 32-45-67-89-AB-CD از این نوع نمی باشند.

آدرس FF-FF-FF-FF-FF-FF از نوع broadcast است، به این معنا که همه ی گره های شبکه آن را دریافت و مورد بررسی قرار می دهند.

اگر کم ارزش ترین بیت از اولین بایت صفر باشد، آدرس از نوع unicast است و برای گره ای خاص در شبکه ارسال می شود.

تراشه ENC28J60 دارای فیلترهایی در بخش گیرنده است که میتوانند به شکلی تنظیم شوند که بسته های خاص با آدرس هایی مشخص را دریافت کند، در این صورت در خطوط برنامه نیازی به بررسی آدرس بسته ها نداریم و حجم پردازش کمتر می شود. هنگام ارسال باید آدرس مقصد توسط میکروکنترلر تنظیم شود.

۷-۱۸-۳. SOURCE ADDRESS

هنگام ارسال بسته ها، میکروکنترلر باید آدرس MAC گره را در بافر فرستنده بنویسد.

نکته : تراشه ENC28J60 محتویات رجیستر MAADR که برای فیلترهای گیرنده آدرس unicast تنظیم شده است را به صورت خودکار ارسال نمی کند.

۷-۱۸-٤. DATA

به صورت پیشفرض حداکثر ظرفیت بخش داده 1500 بایت است و اگر تعداد بایت های بیشتری در این بخش قرار گیرد، توسط تراشه ارسال نمی شود. البته اگر بیت HFRMEN به نام Huge Frame Enable یا فعال سازی فریم های بزرگ را در رجیستر MACON3 فعال کنیم، می توانیم در بخش داده بایت های بیشتری را ارسال کنیم.

۷-۱۸-۵. PADDING

برای این که در صورت نیاز (کمتر بودن طول داده از ۴۶ بایت) به صورت خودکار padding به بسته ها اضافه شود، باید بیت PADCFG در رجیستر MACON3 یک شود، در غیر اینصورت این کار باید

توسط میکروکنترلر و نوشتن خطوط کد مناسب صورت پذیرد.

۷-۱۸-۶. CRC

کد CRC استاندارد، 4 بایت است. اگر بیت CRCEN در رجیستر ERXFCON یک شود، خود تراشه به صورت خودکار کدهای CRC بسته های دریافتی را بررسی می کند و در صورتی که اشتباه باشند، آن بسته را دریافت و ذخیره نمی کند. در صورتی که این بیت صفر باشد، برنامه نویس می تواند از طریق خواندن بردار وضعیت دریافت (receive status vector) صحت کد CRC را به صورت دستی بررسی کند.

همچنین اگر بیت PADCFG در رجیستر MACON3 یک باشد، هنگام ارسال بسته ها، تراشه به صورت خودکار کد CRC متناسب با بسته را محاسبه و به آن اضافه می کند. در صورتی که این بیت صفر باشد، برنامه نویس باید به صورت دستی برای هر بسته از داده که می خواهد در شبکه ارسال کند، کد CRC متناسب با آن بسته را محاسبه کرده و در بافر فرستنده به همراه داده بنویسد.

نکته مهم : با توجه به پیچیدگی محاسبه ی کد CRC پیشنهاد می شود این کار به صورت دستی و نوشتن کد انجام نشود و بیت PADCFG را یک کنیم تا تراشه خودش به صورت خودکار این محاسبات را انجام دهد.

۷-۱۹. پیکربندی اولیه (INITIALIZATION)

قبل از این که تراشه شروع به ارسال و دریافت بسته ها در شبکه بکند، بسته به کاربرد مورد نظر شاید لازم باشد برخی از تنظیمات پیشفرض تراشه تغییر پیدا کنند که در این بخش به این تنظیمات می پردازیم.

۷-۱۹-۱. بافر گیرنده

قبل از شروع به دریافت بسته ها از شبکه، باید محلی را برای ذخیره سازی بسته های دریافتی تعیین کنیم. برای تعیین بخشی از حافظه به عنوان بافر گیرنده باید رجیسترهای اشاره گر ERXST و

ERXND را با آدرس های مناسبی از حافظه مقدار دهی کنیم. آدرس های بین این دو مقدار برای بافر گیرنده در نظر گرفته می شود.

پیشنهاد می شود که رجیستر ERXST روی آدرس های زوج تنظیم شود.

برای کاربردهایی که تعداد بسته های دریافتی و سرعت دریافت آنها زیاد است، باید بیشتر حافظه بافر را به بافر گیرنده اختصاص داد، از طرفی اگر نیاز به ارسال تعداد زیادی بسته به شبکه داریم، باید بخش بافر گیرنده را کوچک تر در نظر بگیریم تا بقیه ی حافظه را به بافر فرستنده تخصیص دهیم.

۷-۱۹-۲. بافر فرستنده

پس از تعیین بخشی از حافظه به عنوان بافر گیرنده، بقیه ی حافظه به صورت خودکار به بافر فرستنده اختصاص پیدا می کند و نیاز به تنظیم رجیستر خاصی وجود ندارد.

۷-۱۹-۳. فیلترهای گیرنده

در صورتی که می خواهیم آدرس بسته ها به صورت خودکار توسط فیلترهای گیرنده ای که در تراشه وجود دارد بررسی شود، می توان تنظیمات این بخش را در رجیستر ERXFCON انجام داد. در بخش فیلترهای گیرنده نحوه تنظیم این رجیستر توضیح داده خواهد شد.

۷-۱۹-٤. زمان OST

همان طور که قبلا هم اشاره شد، پس از وصل شدن تغذیه یا ریست شدن تراشه، مدت زمانی باید بگذرد تا بتوانیم تنظیماتی مانند تنظیم رجیسترهای MAC و PHY را انجام دهیم. برای اطمینان از گذشتن این زمان می توانیم بیت CLKRDY در رجیستر ESTAT را بررسی کنیم. (روش polling)

(در بخش های قبلی در مورد این مسئله توضیح داده شده است)

۷-۱۹-۵. تنظیمات اولیه MAC

رجیسترهایی از MAC که باید در ابتدا تنظیم شوند عبارت اند از : (ترتیب تنظیم رجیسترها مهم نیست)

یک کردن بیت MARXEN در رجیستر MACON1 برای ایجاد امکان دریافت فریم ها در لایه ی MAC . اگر ارتباط در حالت full duplex تنظیم شده است، باید بیت های TXPAUS و RXPAUS نیز برای ایجاد قابلیت کنترل جریان فریم ها به منظور تطابق با استاندارد IEEE یک شوند.

تنظیم بیت های PADCFG ، TXCRCEN و FULDPX از رجیستر MACON3 . این بیت ها به منظور تنظیم امکان تولید خوکار pad و CRC می باشد. با یک کردن بیت FRMLNEN گزارش طول فریم ها نیز فعال می شود. با یک کردن بیت FULDPX ارتباط در حالت full duplex قرار می گیرد.

یک کردن بیت DEFER در رجیستر MACON4 به منظور تطابق با استانداردهای IEEE .

تنظیم رجیستر MAMXFL با حداکثر طول فریم ها. در حالت معمول حداکثر طول فریم ها 1518 بایت است.

تنظیم رجیستر MABBIPG ، در اغلب کاربردها، برای حالت Full-Duplex با مقدار 15h و برای حالت Half-Duplex با مقدار 12h تنظیم می شود.

تنظیم رجیستر MAIPGL ، در اغلب کاربردها با مقدار 12h تنظیم می شود.

اگر در حالت half duplex هستیم، باید رجیستر MAIPGH را تنظیم کنیم، در اغلب کاربردها با مقدار 0Ch تنظیم می شود.

اگر در حالت half duplex هستیم، می توان رجیسترهای MACLCON1 و MACLCON2 را تنظیم کنیم. در اغلب کاربردها نیازی به تغییر مقدار پیشفرض این رجیسترها وجود ندارد. اگر از کابل شبکه بلند استفاده می کنیم، نیاز به افزایش مقدار پیشفرض رجیستر MACLCON2 داریم.

تنظیم رجیسترهای MAADR1 تا MAADR6 با آدرس MAC .

در این بخش رجیسترهای MAC که در قسمت قبل توضیحات آن را دادیم آورده شده و بیت های مهم آن به همراه مقادیر متداول برای بیت ها مورد بررسی قرار میگیرند.

۷-۱۹-۶. رجیستر MACON1

REGISTER 6-1: MACON1: MAC CONTROL REGISTER 1

U-0	U-0	U-0	R-0	R/W-0	R/W-0	R/W-0	R/W-0
—	—	—	r	TXPAUS	RXPAUS	PASSALL	MARXEN
bit 7							bit 0

شکل۷-۲۱

مطابق توضیحات ارائه شده، بیت های MARXEN ، TXPAUS و RXPAUS را یک می کنیم.

۷-۱۹-۷. رجیستر MACON3

REGISTER 6-2: MACON3: MAC CONTROL REGISTER 3

R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0
PADCFG2	PADCFG1	PADCFG0	TXCRCEN	PHDREN	HFRMEN	FRMLNEN	FULDPX
bit 7							bit 0

شکل۷-۲۲

مطابق توضیحات ارائه شده، یک کردن بیت های PADCFG0 ، PADCFG1 و PADCFG2 برای تنظیم خوکار pad و CRC ، یک کردن بیت TXCRCEN برای فعال کردن ارسال CRC با بسته ها.

۷-۱۹-۸. رجیستر MACON4

REGISTER 6-3: MACON4: MAC CONTROL REGISTER 4

U-0	R/W-0	R/W-0	R/W-0	U-0	U-0	R-0	R-0
—	DEFER	BPEN	NOBKOFF	—	—	r	r
bit 7							bit 0

شکل۷-۲۳

مطابق توضیحات ارائه شده، یک کردن بیت DEFER

۷-۱۹-۹. رجیستر MABBIPG

REGISTER 6-4: MABBIPG: MAC BACK-TO-BACK INTER-PACKET GAP REGISTER

U-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0
—	BBIPG6	BBIPG5	BBIPG4	BBIPG3	BBIPG2	BBIPG1	BBIPG0
bit 7							bit 0

شکل۷-۲۴

مطابق توضیحات ارائه شده، چون حالت half-duplex را انتخاب کردیم، با مقدار 12h رجیستر را تنظیم می کنیم.

۷-۱۹-۱۰. تنظیمات اولیه رجیسترهای لایه فیزیکی

همان طور که در بخش های قبل ذکر شده، با تنظیم جهت LED، بیت PDPXMD در رجیستر PHCON1 به منظور ارتباط full-duplex یا half-duplex تنظیم می شود. اگر نخواهیم از LED استفاده کنیم یا به شکل صحیحی تنظیم نشود، می توانیم به صورت دستی با تغییر این بیت این کار را انجام دهیم. همچنین می توان با خواندن مقدار بیت PDPXMD ، مقدار بیت FULDPX را تنظیم کنیم.

نکته : مقدار بیت PDPXMD با مقدار بیت FULDPX در رجیستر MACON3 باید تطابق داشته باشد. (از نظر full-duplex یا half-duplex بودن ارتباط)

اگر ارتباط را از نوع half-duplex تنظیم کرده ایم، می توان با یک کردن بیت HDLDIS در رجیستر PHCON2 از loopback خودکار هنگام ارسال داده جلوگیری کرد.

REGISTER 6-5: PHCON2: PHY CONTROL REGISTER 2

U-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0
—	FRCLNK	TXDIS	r	r	JABBER	r	HDLDIS
bit 15							bit 8

شکل۷-۲۵

مطابق توضیحات ارائه شده، در حالت half-duplex می توان بیت HDLDIS را یک کرد تا از loopback خودکار جلوگیری شود.

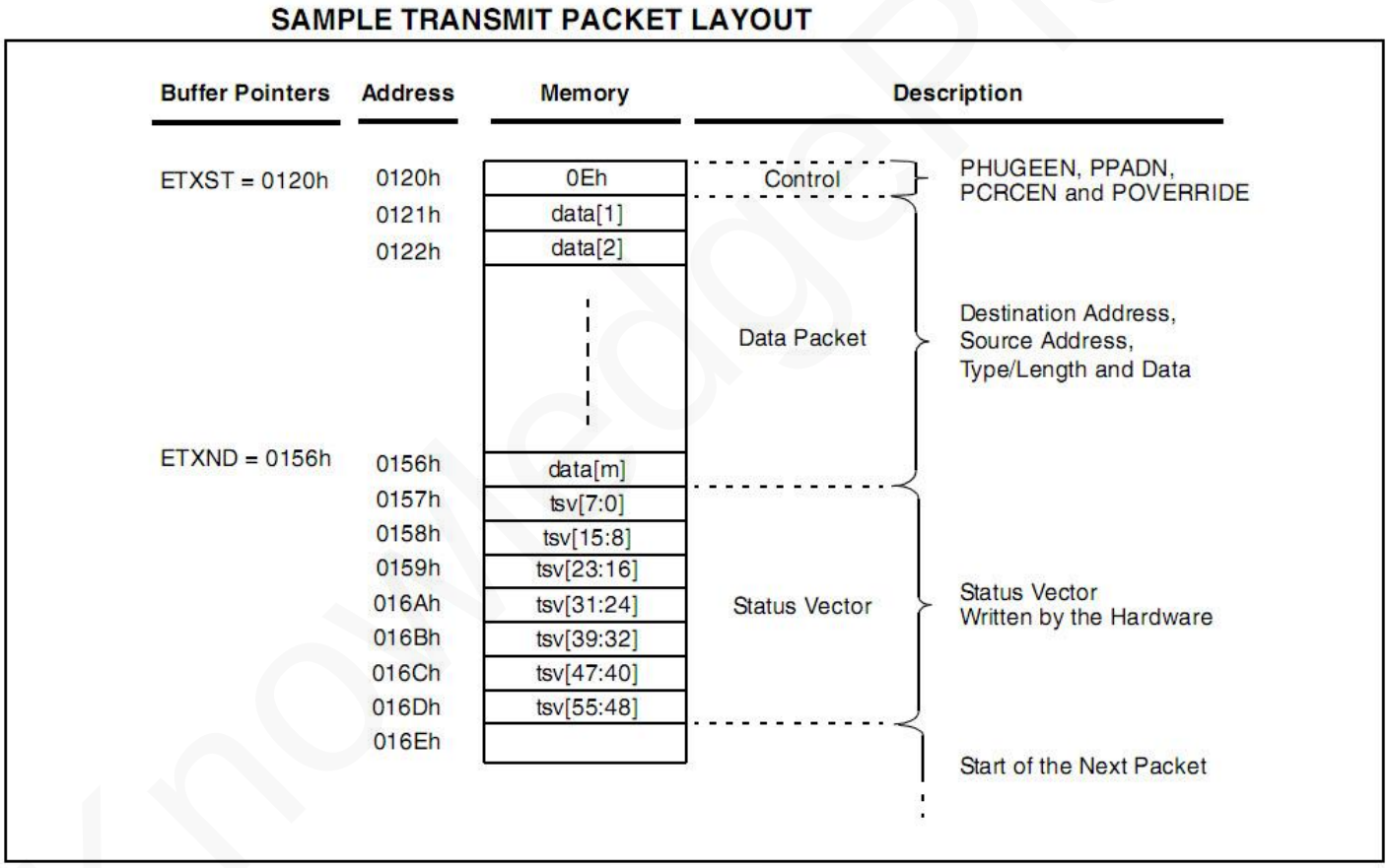
۷-۲۰. بسته های ارسالی و دریافتی

۷-۲۰-۱. بسته های ارسالی

بخش MAC تراشه به شکل خودکار بخش های preamble و start-of-frame delimiter و در صورت تنظیم شدن، بخش های padding و CRC یک فریم را تولید می کند. بخش های دیگر یک فریم باید توسط میکروکنترلر در حافظه بافر نوشته شود.

به ازای ارسال هر بسته، یک بایت به عنوان اطلاعات کنترلی ارسال می شود.

شکل زیر ساختار کامل بسته های ارسالی را نشان می دهد.



شکل۷-۲۶

در ابتدای بسته، یک بایت کنترلی قرار می گیرد. در بخش بعدی آدرس فرستنده، آدرس گیرنده و داده های مورد نظر کاربر قرار می گیرد. در بخش status vector یا بردار وضعیت، اطلاعاتی که به شکل خودکار توسط تراشه تولید می شود و در بخش های قبلی توضیح داده شد قرار می گیرد.

برای تشکیل دادن بسته های ارسالی، می توان مراحل زیر را دنبال کرد :

- اشاره گر ETXST را با آدرس مناسب از بخش خالی از حافظه تنظیم می کنیم. مطابق شکل ۷-۲۶، بایت کنترل از این آدرس شروع می شود. در این مثال آدرس برابر 0120h تنظیم می شود. پیشنهاد می شود از آدرس های زوج استفاده کنید.
- از دستور WBM مربوط به SPI برای نوشتن بایت کنترلی (در صورت نیاز)، آدرس مقصد، آدرس فرستنده و بخش های دیگر فریم استفاده می کنیم.
- در اشاره گر ETXND ، آدرس آخرین داده نوشته شده در فریم را قرار می دهیم. در این مثال آدرس برابر 0156h تنظیم شده است.
- در صورتی که میخواهید از سیگنال وقفه تراشه استفاده کنید، بیت TXIF در رجیستر EIR را صفر و بیت های TXIE و INTIE در رجیستر EIE را یک تنظیم کنید.
- برای شروع ارسال، بیت TXRTS در رجیستر ECON1 را یک می کنیم.

نکته : در هنگام ارسال بسته ها به شبکه (پس از یک کردن بیت TXRTS)، نباید رجیسترهای حاشیه زده شده در شکل زیر را تغییر دهید. همچنین نباید بایت هایی که در حال ارسال است خوانده یا نوشته شوند. (برای انصراف از ارسال بسته می توان بیت TXRTS را صفر کرد)

SUMMARY OF REGISTERS USED FOR PACKET TRANSMISSION

Register Name	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0	Reset Values on page
EIE	INTIE	PKTIE	DMAIE	LINKIE	TXIE	r	TXERIE	RXERIE	13
EIR	—	PKTIF	DMAIF	LINKIF	TXIF	r	TXERIF	RXERIF	13
ESTAT	INT	BUFER	r	LATECOL	—	RXBUSY	TXABRT	CLKRDY	13
ECON1	TXRST	RXRST	DMAST	CSUMEN	TXRTS	RXEN	BSEL1	BSEL0	13
ETXSTL	TX Start Low Byte (ETXST<7:0>)								13
ETXSTH	—	—	—	TX Start High Byte (ETXST<12:8>)					13
ETXNDL	TX End Low Byte (ETXND<7:0>)								13
ETXNDH	—	—	—	TX End High Byte (ETXND<12:8>)					13
MACON1	—	—	—	r	TXPAUS	RXPAUS	PASSALL	MARXEN	14
MACON3	PADCFG2	PADCFG1	PADCFG0	TXCRCEN	PHDREN	HFRMEN	FRMLNEN	FULDPX	14
MACON4	—	DEFER	BPEN	NOBKOFF	—	—	r	r	14
MABBIPG	—	Back-to-Back Inter-Packet Gap (BBIPG<6:0>)							14
MAIPGL	—	Non-Back-to-Back Inter-Packet Gap Low Byte (MAIPGL<6:0>)							14
MAIPGH	—	Non-Back-to-Back Inter-Packet Gap High Byte (MAIPGH<6:0>)							14
MACLCON1	—	—	—	—	Retransmission Maximum (RETMAX<3:0>)				14
MACLCON2	—	—	Collision Window (COLWIN<5:0>)						14
MAMXFLL	Maximum Frame Length Low Byte (MAMXFL<7:0>)								14
MAMXFLH	Maximum Frame Length High Byte (MAMXFL<15:8>)								14

جدول ۷-۶

پس از پایان ارسال (یا انصراف از آن)، بیت TXRTS در رجیستر ECON1 صفر می شود، بیت TXIF در رجیستر EIR یک شده و وقفه در صورت فعال سازی ایجاد می شود.

همچنین هفت بایت مربوط به بخش بردار وضعیت در آدرس ETXND + 1 توسط سخت افزار نوشته می شود.

برای بررسی از صحت ارسال بسته، بیت TXABRT در رجیستر ESTAT را بررسی می کنیم. اگر یک بود یعنی مشکلی در ارسال بسته به وجود آمده است و برای بررسی علت آن باید بیت LATECOL در رجیستر ESTAT و بیت های مربوط به فرستنده در بخش بردار وضعیت را که توسط سخت افزار تولید شده است بررسی کنیم.

۷-۲۰-۲. بسته های دریافتی

پس از انجام تنظیمات اولیه که در بخش های قبلی توضیح داده شد :

برای فعال سازی وقفه دریافت پیام ها (در صورت نیاز) بیت های PKTIE و INTIE در رجیستر EIE را یک می کنیم.

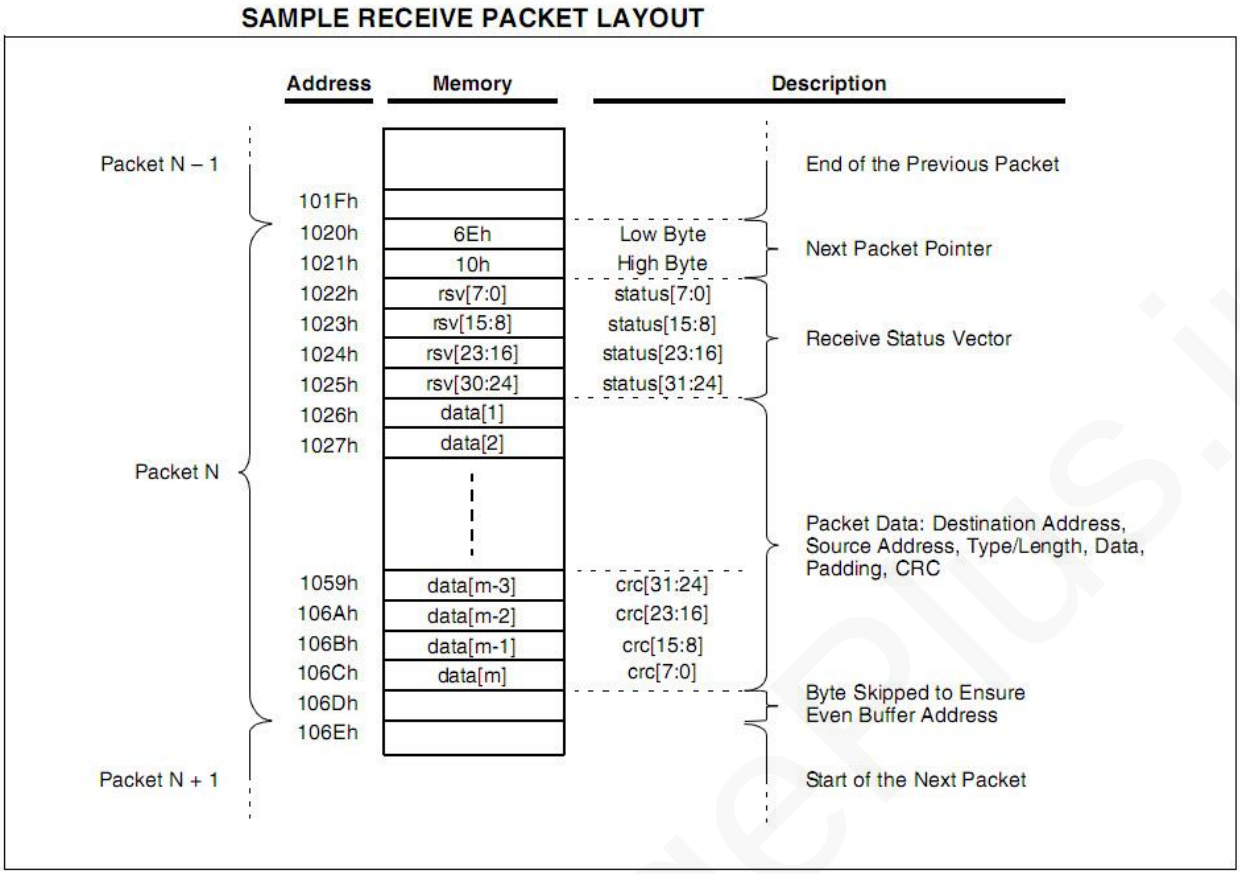
برای فعال سازی وقفه ذخیره نشدن پیام در بافر به دلیل نبودن فضای کافی، بیت RXERIF در رجیستر EIR را صفر و بیت های RXERIE و INTIE در رجیستر EIE را یک می کنیم.

فعال سازی دریافت با یک کردن بیت RXEN در رجیستر ECON1 .

نکته : پس از فعال سازی بیت RXEN ، حالت duplex و اشاره گرهای تعیین کننده ی شروع و پایان بافر دریافت نباید تغییر کنند. همچنین برای عدم دریافت پیام های اشتباه، پیشنهاد می شود قبل از تغییر تنظیمات فیلترهای گیرنده و آدرس MAC ، بیت RXEN را صفر کنید.

پس از دریافت هر بسته داده و نوشته شدن آن در بافر گیرنده ، بیت PKTIF در رجیستر EIR یک می شود و در صورت فعال سازی وقفه، سیگنال وقفه نیز تولید می شود.

شکل ۷-۲۷ ساختار کامل بسته های دریافتی را نشان می دهد که ساختاری مشابه بسته های ارسالی دارد.



شکل۷-۲۷

۷-۲۰-۳. خواندن بسته های دریافتی

برای خواندن بسته های دریافتی باید از دستور RBM مربوط به SPI استفاده نمود. در این حالت باید ابتدا بردار وضعیت خوانده و رجیسترهای لازم در این قسمت ذخیره شود، سپس داده های موجود در بسته خوانده شود. برای خواندن بسته بدون نیاز به تغییر رجیستر ERDPT و به صورت پشت سر هم، باید بیت AUTOINC در رجیستر ECON2 را یک کنیم.

۷-۲۰-٤. آزادسازی فضای بسته های دریافتی

پس از خواندن هر بسته (یا بخشی از آن)، برای آزاد سازی فضای اختصاص داده شده به آن، باید اشاره گر ERXRDPT که برای خواندن داده های بافر گیرنده استفاده می شود را افزایش دهیم. اگر تراشه بخواهد بسته دریافت شده را به جای بسته فعلی بنویسد، بسته دریافت شده جدید نوشته نخواهد شد و بیت RXERIF در رجیستر EIR یک شده و سیگنال وقفه در صورت فعال سازی تولید خواهد شد، در نتیجه

تراشه هیچ وقت بسته های دریافت شده را به جای بسته های خوانده نشده نمی نویسد.

نکته : برای نوشتن در اشاره گر ERXRDPT ، ابتدا باید در بایت پایینی آن یعنی ERXRDPTL نوشت و سپس در بایت بالایی آن یعنی ERXRDPTH اما در خواندن رعایت این ترتیب اهمیتی ندارد.

برای افزایش مقدار اشار گر ERXRDPT پس از خواندن کامل یک بسته، باید بیت PKTDEC در رجیستر ECON2 را یک کنیم. با این کار مقدار رجیستر EPKTCNT یک واحد کاهش پیدا می کند. اگر مقدار EPKTCNT برابر صفر شود، بیت PKTIF در رجیستر EIR به شکل خودکار صفر می شود و نشانگر نبودن بسته ای جدید در بافر گیرنده برای خواندن آن است و اگر مقدار EPKTCNT با یک واحد کاهش برابر صفر نشد، بیت PKTIF برابر یک خواهد ماند و نشان دهنده ی وجود بسته های خوانده نشده در حافظه ی بافر گیرنده است.

نکته : اگر مقدار رجیستر EPKTCNT به عدد ۲۵۵ برسد، بسته های جدید دریافت نمی شوند (حتی اگر حافظه کافی برای دریافت بسته ها وجود داشته باشد) و بیت RXERIF در رجیستر EIR به نشانه خطا یک خواهد شد و سیگنال وقفه نیز در صورت فعال سازی تولید خواهد شد. برای جلوگیری از این اتفاق باید دائما مقدار رجیستر EPKTCNT توسط میکروکنترلر بعد از خواندن هر بسته دریافتی، کاهش پیدا کند.

اگر قصد ذخیره ی یک بسته دریافتی را داریم تا بعدا آن را بخوانیم، باید بسته را در بخشی از فضای استفاده نشده از حافظه ذخیره کنیم. برای انجام بهینه این فرایند می توان از کنترل کننده ی DMA موجود در تراشه استفاده کنیم.

۷-۲۱. فیلترهای گیرنده

به منظور کاهش بار پردازشی میکروکنترلر، می توان از فیلترهای گیرنده برای دریافت نکردن بسته هایی که مورد نیاز نیست استفاده نمود. در این تراشه ۶ نوع فیلتر گیرنده پیاده سازی شده است که عبارت اند از :

- Unicast
- Pattern Match
- Magic Packet
- Hash Table
- Multicast
- Broadcast

تنظیمات هر فیلتر توسط رجیستر ERXFCON صورت می پذیرد. قابلیت استفاده همزمان از چندین فیلتر نیز وجود دارد. همچنین این قابلیت وجود دارد که فیلترها به شکلی تنظیم شوند که در صورتی که بسته دریافتی با معیارهای چندین فیلتر سازگار بود دریافت شود. در صورت صفر کردن رجیستر ERXFCON ، فیلترها غیرفعال شده و تمامی بسته ها دریافت می شوند.

۷-۲۱-۱. رجیستر ERXFCON

REGISTER 8-1: ERXFCON: ETHERNET RECEIVE FILTER CONTROL REGISTER

R/W-1	R/W-0	R/W-1	R/W-0	R/W-0	R/W-0	R/W-0	R/W-1
UCEN	ANDOR	CRCEN	PMEN	MPEN	HTEN	MCEN	BCEN
bit 7							bit 0

شکل۷-۲۸

اگر بیت ANDOR یک باشد، تنها بسته هایی دریافت می شود که با شرایط تمام فیلترهای فعال شده تطابق داشته باشد (منطق AND)، در صورتی که این بیت صفر باشد، اگر بسته تنها با یکی از فیلترها تطابق داشته باشد، دریافت می شود. (منطق OR)

با یک کردن بیت UCEN ، فیلتر آدرس MAC فعال می شود، به این معنا که اگر آدرس MAC بسته با آدرس MAC تنظیم شده برابر نباشد، بسته دریافت نمی شود.

در صورت یک بودن بیت CRCEN ، تنها بسته های با CRC صحیح دریافت می شوند.

با یک کردن بیت PMEN ، فیلتر مقایسه الگو (Pattern Match) فعال می شود.

با یک کردن بیت MPEN ، فیلتر بسته های Magic یا غیرعادی فعال می شود.

با یک کردن بیت HTEN ، فیلتر جدول Hash فعال می شود.

با یک کردن بیت MCEN ، فیلتر Multicast فعال می شود، به این معنا که بیت کم ارزش آدرس مقصد بسته باید برابر یک باشد تا بسته دریافت شود.

با یک کردن بیت BCEN ، فیلتر Broadcast فعال می شود، به این معنا که فقط بسته های با آدرس مقصد FF-FF-FF-FF-FF-FF دریافت می شوند.

۷-۲۱-۲. فیلتر Unicast

در صورت فعال سازی این فیلتر، فقط بسته هایی که آدرس مقصد آنها با آدرس نوشته شده در رجیستر MAADR یکسان باشند دریافت می شوند.

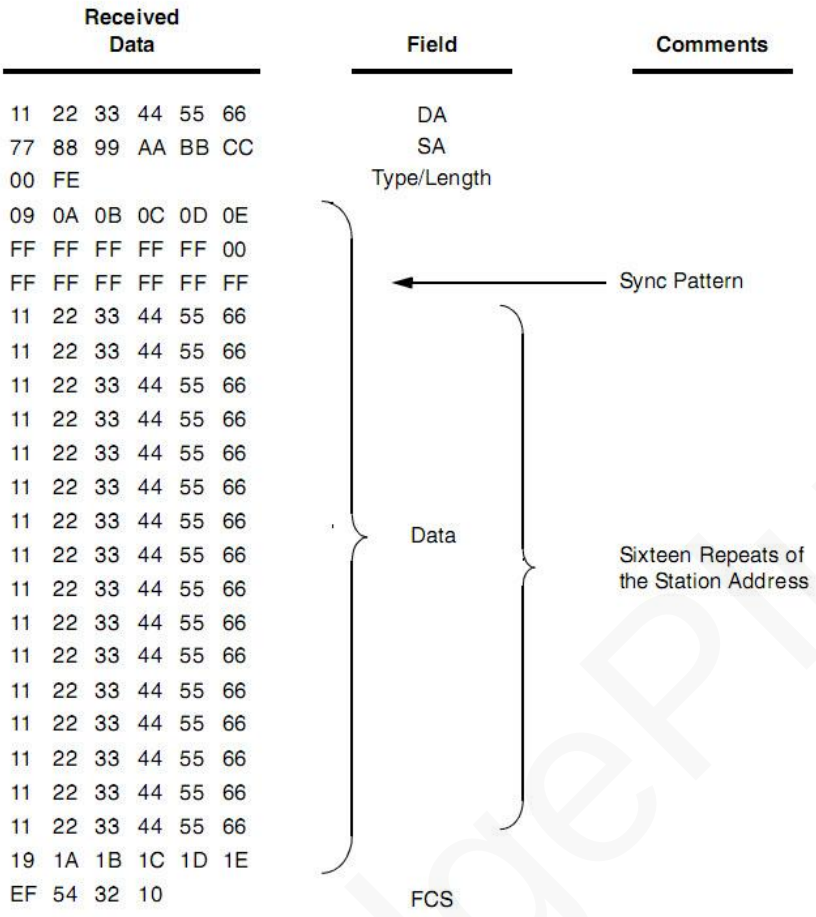
۷-۲۱-۳. فیلتر تطبیق الگو (Pattern Match Filter)

از این فیلتر برای دریافت بسته هایی با داده های مشخص استفاده می شود. این عمل از طریق محاسبه ی IP checksum برای تعدادی از بایت های دریافت شده و مقایسه ی آن با رجیستر EPMCS صورت می پذیرد.

۷-۲۱-٤. فیلتر Magic Packet

در صورت فعال سازی این فیلتر، فقط بسته هایی که آدرس مقصد آنها با آدرس نوشته شده در رجیستر MAADR یکسان باشد و در بخش داده حاوی یک بسته magic یا غیرعادی باشد دریافت می شود.

بسته غیر عادی یا magic به بسته ای گفته می شود که دارای شش بایت هماهنگ (sync) با محتوای 0xFF باشد و در ادامه ی آن آدرس مقصد برای ۱۶ بار تکرار شود. شکل ۷-۲۹ مثالی از این نوع بسته را نشان می دهد.



شکل۷-۲۹

۷-۲۱-۵. فیلتر جدول Hash

در صورت فعال سازی این فیلتر، ابتدا یک عملیات یا محاسبات CRC در بخش آدرس مقصد بسته صورت می پذیرد و سپس نتایج آن به عنوان یک اشاره گر برای رجیستر EHT به کار می رود، اگر بیتی که اشاره گر به آن اشاره می کند یک باشد، بسته دریافت می شود و در غیر اینصورت بسته دریافت نمی شود. به عنوان مثال اگر حاصل عملیات CRC برابر 0x5 بشود، بیت پنجم در جدول hash مورد بررسی قرار می گیرد و در صورت یک بودن این بیت، بسته دریافت می شود.

بدیهی است اگر همه ی بیت های جدول hash یک باشد، همه ی بسته ها دریافت می شود و اگر همه ی بیت های جدول hash صفر باشد، بسته ای دریافت نمی شود.

۷-۲۱-۶. فیلتر Multicast

در صورت فعال سازی این فیلتر، فقط بسته هایی که بیت کم ارزش تر آدرس مقصد آنها یک باشد دریافت می شوند.

۷-۲۱-۷. فیلتر Broadcast

در صورت فعال سازی این فیلتر، فقط بسته هایی که آدرس مقصد آنها برابر FF-FF-FF-FF-FF-FF باشد دریافت می شوند.

نکته مهم : در توضیحات فیلترهای بالا لفظ "دریافت شدن" یا "دریافت نشدن" بسته ها به معنای رعایت شدن شرایط آن فیلتر و یا رعایت نشدن شرایط آن فیلتر می باشد و تصمیم نهایی سخت افزار برای دریافت و یا عدم دریافت بسته ها همان طور که در بخش رجیستر ERXFCON نیز توضیح داده شد، به نوع منطق مورد استفاده (تنظیم بیت ANDOR روی منطق OR یا AND) بستگی دارد.

نکته : تراشه ENC28J60 از قابلیت automatic duplex negotiation پشتیبانی نمی کند، به این معنا که اگر به یک switch ، router یا گره ی دیگر از شبکه وصل شود که از این قابلیت پشتیبانی می کند، نوع ارتباط half-duplex تنظیم خواهد شد. برای استفاده از نوع ارتباط Full-Duplex ، باید تراشه ENC28J60 و سیستم های دیگر متصل به شبکه به صورت دستی تنظیم شوند.

۷-۲۲. منابع وقفه

تراشه ENC28J60 دارای منابع وقفه متعددی است و توسط پایه خارجی نیز در مواقع مختلف با ایجاد یک سیگنال پایین رونده در موقعیت های مختلف، امکان استفاده از آن برای میکروکنترلر را فراهم می کند. دو رجیستر مربوط به تنظیمات وقفه ها در این تراشه در نظر گرفته شده است.

رجیستر EIE که برای فعال یا غیرفعال سازی وقفه ها به کار می رود و رجیستر EIR شامل پرچم ها (flag) نشان دهنده ی اتفاق افتادن وقفه است.

هنگامی که وقفه ای اتفاق می افتد، پرچم مربوطه در رجیستر EIR یک شده و در صورت یک بودن بیت فعال سازی همگانی وقفه (INTIE)، پایه وقفه خارجی INT صفر می شود.

تا زمانی که پرچم مربوطه یک می باشد، پایه ی INT صفر باقی می ماند. در صورت فعال شدن بیش از یک وقفه به صورت همزمان، برای تشخیص منبع وقفه ایجاد شده باید رجیستر EIR در خطوط برنامه بررسی شود.

نکته : پیشنهاد می شود برای صفر کردن پرچم های وقفه در رجیستر EIR ، به جای استفاده از دستور Write Control Register یا WCR ، از دستور Bit Field Clear یا BFC استفاده شود. این مسئله برای پاک نشدن پرچم های دیگر هنگام عملیات WCR اهمیت دارد. این دستورات SPI در بخش های قبلی توضیح داده شده اند.

نکته : به غیر از وقفه LINKIF ، تمامی پرچم های وقفه حتی در صورت فعال نبودن وقفه مربوطه در رجیستر EIE و فعال نبودن بیت INTIE ، با اتفاق افتادن وقفه یک می شوند و در صورتی که از روش سرکشی برای تشخیص وقفه استفاده می شود، باید پرچم های یک شده برای استفاده مجدد از وقفه ها در نرم افزار صفر شوند.

قبل از سرویس دهی به وقفه، باید برای یک شدن پایه ی INT ، بیت INTIE صفر شود. این کار باعث می شود که اگر وقفه ی دیگری هنگام سرویس دهی به وقفه جاری اتفاق افتاد، از دست نرود.

پس از سرویس دهی به وقفه، بیت INTIE یک میشود.

۷-۲۲-۱. رجیستر ESTAT

REGISTER 12-1: ESTAT: ETHERNET STATUS REGISTER

R-0	R/C-0	R-0	R/C-0	U-0	R-0	R/C-0	R/W-0
INT	BUFER	r	LATECOL	—	RXBUSY	TXABRT	CLKRDY
bit 7							bit 0

شکل۷-۳۰

یک بودن بیت INT نشان دهنده ی اتفاق افتادن وقفه است.

یک بودن بیت BUFER نشان دهنده ی اتفاق افتادن خطا در بخش بافر در اثر عملیات خواندن یا نوشتن است.

یک بودن بیت LATECOL نشان دهنده ی اتفاق افتادن یک تصادم است.

یک بودن بیت RXBUSY نشان دهنده ی دریافت یک بسته است.

یک بودن بیت TXABRT نشان دهنده ی لغو عملیات ارسال است.

یک بودن بیت CLKRDY نشان دهنده ی فعال بودن بخش فرستنده برای ارسال و پایان یافتن زمان OST می باشد.

۷-۲۲-۲. رجیستر EIE

REGISTER 12-2: EIE: ETHERNET INTERRUPT ENABLE REGISTER

R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0
INTIE	PKTIE	DMAIE	LINKIE	TXIE	r	TXERIE	RXERIE
bit 7							bit 0

شکل۷-۳۱

یک کردن بیت INTIE باعث فعال سازی پایه وقفه خارجی INT می شود.

یک کردن بیت PKTIE باعث فعال سازی وقفه دریافت می شود.

یک کردن بیت DMAIE باعث فعال سازی وقفه DMA می شود.

یک کردن بیت LINKIE باعث فعال سازی وقفه تغییر اتصال لایه ی PHY می شود.

یک کردن بیت TXIE باعث فعال سازی وقفه ارسال می شود.

یک کردن بیت TXERIE باعث فعال سازی وقفه خطا در ارسال می شود.

یک کردن بیت RXERIE باعث فعال سازی وقفه خطا در دریافت می شود.

۷-۲۲-۳. رجیستر EIR

REGISTER 12-3: EIR: ETHERNET INTERRUPT REQUEST (FLAG) REGISTER

U-0	R-0	R/C-0	R-0	R/C-0	R-0	R/C-0	R/C-0
—	PKTIF	DMAIF	LINKIF	TXIF	r	TXERIF	RXERIF
bit 7							bit 0

شکل۷-۳۲

یک بودن بیت PKTIF نشان دهنده ی وجود بسته یا بسته هایی در حافظه بافر هست که خوانده نشده اند، با یک شدن بیت PKTDEC این بیت صفر می شود.

یک بودن بین DMAIF نشان دهنده ی به اتمام رسیدن محاسبات کپی یا checksum است.

یک بودن بیت LINKIF نشان دهنده ی تغییر در وضعیت اتصال (link) می باشد، با خواندن رجیستر PHIR این بیت صفر می شود.

یک بودن بیت TXIF نشان دهنده ی پایان یافتن درخواست ارسال است.

یک بودن بیت TXERIF نشان دهنده ی رخ دادن خطای ارسال است.

یک بودن بیت RXERIF نشان دهنده ی رخ دادن خطای دریافت در اثر پر شدن حافظ بافر گیرنده یا رسیدن شمارنده بسته به عدد ۲۵۵ می باشد.

۷-۲۲-٤. رجیستر PHIE

REGISTER 12-4: PHIE: PHY INTERRUPT ENABLE REGISTER

R-0	R-0	R-0	R-0	R-0	R-0	R-0	R-0
r	r	r	r	r	r	r	r
bit 15							bit 8

R-0	R-0	R/W-0	R/W-0	R-0	R-0	R/W-0	R/W-0
r	r	r	PLNKIE	r	r	PGEIE	r
bit 7							bit 0

شکل۷-۳۳

یک بودن کردن بیت PLNKIE باعث فعال شدن وقفه تغییر در اتصال (link) می شود.

یک کردن بیت PGEIE باعث فعال شدن وقفه های لایه فیزیکی یا PHY می شود.

۷-۲۲-۵. رجیستر PHIR

REGISTER 12-5: PHIR: PHY INTERRUPT REQUEST (FLAG) REGISTER

R-x	R-x	R-x	R-x	R-x	R-x	R-x	R-x
r	r	r	r	r	r	r	r
bit 15							bit 8

R-x	R-x	R-0	R/SC-0	R-0	R/SC-0	R-x	R-0
r	r	r	PLNKIF	r	PGIF	r	r
bit 7							bit 0

شکل۷-۳۴

یک بودن کردن بیت PLNKIF نشان دهنده ی تغییر کردن وضعیت اتصال در لایه فیزیکی است، با خواندن این بیت مقدارش صفر می شود.

یک بودن بیت PGIF نشان دهنده ی اتفاق افتادن یکی از وقفه های لایه فیزیکی است. با خواندن این بیت مقدارش صفر می شود.

فصل هشتم

پیاده سازی عملی اترنت

۸-۱. مقدمه

در فصل های قبلی این مقاله با روش های و راه کار های مختلف طراحی سخت افزار لازم برای اتصال مدارات مختلف به شبکه اترنت و همچنین کاربردهای مختلف قابل پیاده سازی در چارچوب سیستم های embedded آشنا شدیم.

در این فصل با توجه به مطالب آموخته شده تا اینجای مقاله، قصد طراحی یک پروژه به منظور آشنایی عملی با نحوه ی طراحی سخت افزار و کدنویسی های لازم جهت ایجاد کاربردهای مورد نظر را داریم.

در بخش سخت افزار، از تراشه ENC28J60 شرکت Microchip برای پیاده سازی نقش Ethernet Controller یا کنترل کننده اترنت و transceiver یا واحد فرستنده و گیرنده استفاده می کنیم. در واقع این تراشه لایه فیزیکی و لایه پیوند داده را پیاده سازی می کند. برای پیاده سازی لایه های بالاتر از میکروکنترلر AVR مدل Atmega32 استفاده می کنیم.

برای استفاده از تراشه ENC28J60 سه روش مختلف پیشنهاد داده شده است که میتوانید با توجه به هزینه و زمان، یکی از این روش های را انتخاب کنید. برای هر سه روش پیشنهاد شده فایل های شماتیک و PCB ارائه شده است که می توانید برای ساخت پروژه از آنها استفاده کنید.

در بخش نرم افزار ۳ نوع پروژه ارائه شده است. در پروژه اول با استفاده از توابع سطح پایین نوشته شده برای تراشه ENC28J60 در شبکه اترنت به ارسال و دریافت داده می پردازیم. این سطح از کدنویسی

مشابه کار با پروتکل های دیگر مانند SPI یا I2C می باشد و برای آشنایی با اصول ارسال و دریافت داده در شبکه اترنت مناسب می باشد.

در پروژه دوم هدف پیاده سازی یک وب سرور ساده می باشد، به این ترتیب که مدار طراحی شده از طریق کابل به کامپیوتر متصل شده و یک صفحه وب ساده در مرورگر اینترنتی طراحی می کنیم و اطلاعاتی را بین میکروکنترلر و کامپیوتر رد و بدل می کنیم.

پروژه سوم مشابه پروژه اول است با این تفاوت که صفحه وب طراحی شده قابلیت های بیشتری دارد. در این پروژه از کد نویسی پیشرفته تری استفاده شده است و هر کدام از پروتکل های استفاده شده (مانند TCP ، HTTP و...) در کتابخانه های جداگانه ای استفاده شده و به برنامه اصلی الحاق شده اند.

در پروژه دوم، ساده و خلاصه بودن کدها مد نظر قرار گرفته است. برای ساده سازی توابع مربوط به تراشه ENC28J60 و پیاده سازی پروتکل های مختلف، همه ی توابع تنها درون یک فایل قرار گرفته اند، همچنین برای عدم نیاز به اضافه کردن فایل c. به صورت جداگانه، تمامی این توابع در فایل h. به طور کامل پیاده سازی شده اند (برخلاف روال معمول که در این فایل معمولا فقط توابع تعریف می شوند)، بدین ترتیب تنها کافی است این فایل به برنامه اصلی اضافه شود و به راحتی می توان از همه ی توابع آن استفاده نمود.

در پروژه سوم، پیاده سازی همه ی پروتکل ها، توابع تراشه ENC28J60 ، ساخت منوها، فعال سازی ADC میکروکنترلر و... به صورت جداگانه و در فایل های مختلف با پسوند های h. و c. صورت پذیرفته است. بدین ترتیب می توان از این توابع برای برنامه نویسی پیشرفته و در پروژه های مختلف بر حسب نیاز استفاده کرد.

بخش های مهم کدهای نوشته شده در هر سه پروژه به صورت کامل و خط به خط توضیح داده شده اند، همچنین در خود متن کدها نیز از comment برای توضیح بخش های مختلف کد استفاده شده است.

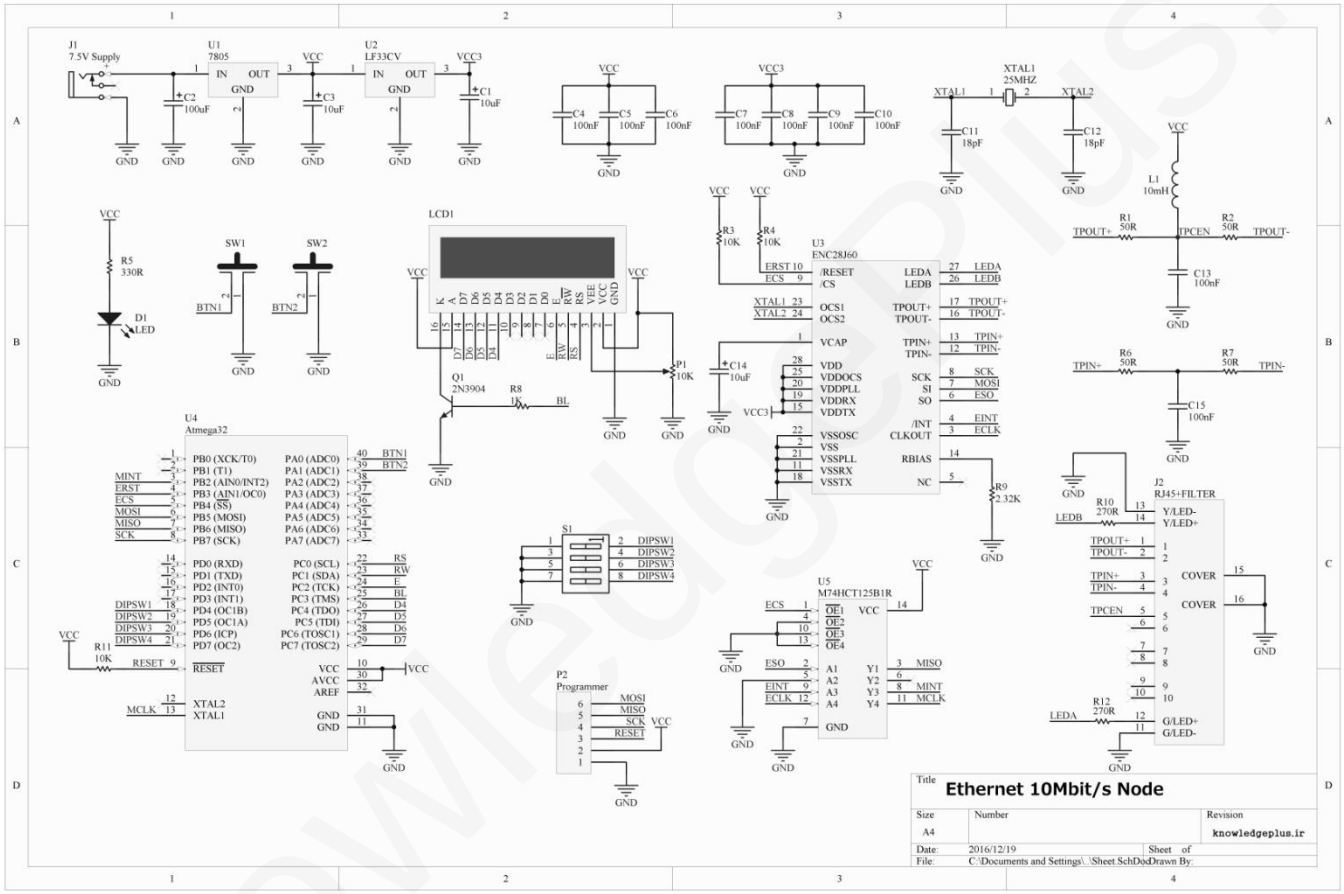
کد پروژه دوم برای سخت افزار شماره ۱ و کد دوم پروژه سوم برای سخت افزار شماره ۳ نوشته شده است. یکی از ویژگی های کدهای نوشته شده این است که به شکل ماژولار (مبتنی بر توابع کتابخانه ای) نوشته شده و از دستورهای #define برای تعریف شناسه ها در ابتدای برنامه استفاده شده است، لذا از این کدها به راحتی می توان در هر آرایش سخت افزاری یا حتی تراشه ها و قطعات دیگر نیز استفاده نمود.

کدهای نوشته شده بر مبنای زبان C و کامپایلر های CodeVisionAVR و WinAVR نوشته شده اند. همچنین برای کامپایلرهای دیگر مانند AVR STUDIO و BASCOM نیز نمونه کدهایی به همراه این مقاله ارائه می شود.

۸-۲. سخت افزار پروژه

۸-۲-۱. روش اول

شکل زیر سخت افزار طراحی شده در این روش را نشان می دهد. (شماتیک و PCB به همراه مقاله ارائه شده است)



شکل ۸-۱

از آنجا که تراشه اترنت به کار رفته با استفاده از ارتباط SPI با میکروکنترلر ارتباط برقرار می کند، میکروکنترلر مورد استفاده باید این قابلیت را داشته باشد. همچنین با توجه به مقدار حافظه لازم برای پردازش بسته های داده اترنت و نیاز به پایه های دیگر برای اتصال سایر قطعات پروژه، میکروکنترلر ATmega32 انتخاب شده است.

برای کانکتور و فیلتر نیز از یک کانکتور دارای فیلتر و چراغ وضعیت ارتباط استفاده شده است، به عنوان مثال کانکتور مدل P65-101-1AK9 . (با رعایت ترتیب پایه ها میتوان از هر کانکتور دیگری که دارای

فیلتر داخلی است نیز استفاده نمود)

همان طور که در فصل هفتم توضیح داده شد، تراشه ENC28J60 با ولتاژ 3.3V کار می کند. در این مدار به علت استفاده از LCD ، ولتاژ 5V نیز لازم است. به این ترتیب می توان طراحی را به دو صورت انجام داد :

میکروکنترلر با ولتاژ 3.3V کار کند و برای ارتباط LCD از مدار تبدیل سطح استفاده شود.

میکروکنترلر با ولتاژ 5V کار کند و ارتباط میکروکنترلر و تراشه شبکه از طریق مدار تبدیل سطح برقرار شود.

در این سخت افزار با توجه به فرکانس نسبتا بالای کلاک، برای میکروکنترلر ATmega32 از تغذیه 5v و برای تراشه ENC28J60 از تغذیه 3.3v استفاده کرده ایم.

به منظور انتقال سیگنال از تراشه به میکروکنترلر، از یک بافر برای تغییر سطح استفاده شده است (74HCT125) اما در جهت عکس یعنی برای انتقال سیگنال از میکروکنترلر به تراشه هیچ نوع واسطی به کار نرفته است و اتصال مستقیم برقرار شده است و علت آن توانایی تحمل سیگنال های 5V در ورودی های تراشه ENC28J60 است.

توجه داشته باشید که میکروکنترلرهای خانواده AVR با ولتاژ 5V کار می کنند و به همین دلیل در صورت اتصال مستقیم خروجی های تراشه ENC28J60 به ورودی های آن، هیچ کدام از نظر الکتریکی آسیب نمی بینند اما به دلیل پیروی میکروکنترلر از مشخصات خانواده HC ، در صورت اتصال مستقیم احتمال بروز خطا در خواندن سطح یک منطقی وجود دارد و در نتیجه استفاده از مدار تبدیل سطح ولتاژ ضروری است.

جدول ۸-۱ مشخصات الکتریکی ورودی و خروجی بافر 74HCT125 را نشان می دهد.

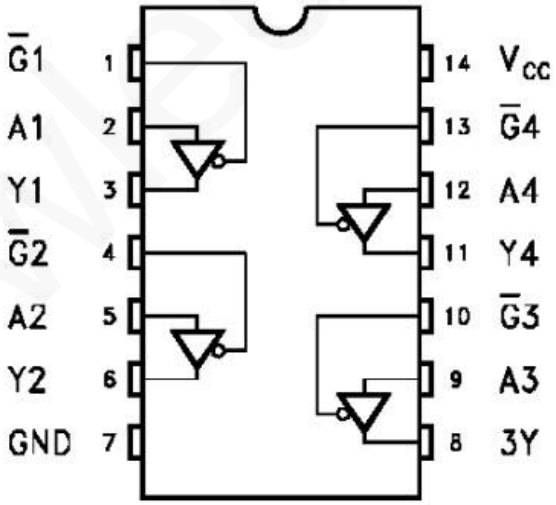
DC SPECIFICATIONS

Symbol	Parameter	Test Conditions			Value							Unit
		Vcc (V)			T_A = 25 °C 54HC and 74HC			-40 to 85 °C 74HC		-55 to 125 °C 54HC		
					Min.	Typ.	Max.	Min.	Max.	Min.	Max.	
V_{IH}	High Level Input Voltage	4.5 to 5.5			2.0			2.0		2.0		V
V_{IL}	Low Level Input Voltage	4.5 to 5.5					0.8		0.8		0.8	V
V_{OH}	High Level Output Voltage	4.5	V_I = V_{IH} or V_{IL}	I_O=-20 µA	4.4	4.5		4.4		4.4		V
				I_O=-6.0 mA	4.18	4.31		4.13		4.10		
V_{OL}	Low Level Output Voltage	4.5	V_I = V_{IH} or V_{IL}	I_O= 20 µA		0.0	0.1		0.1		0.1	V
				I_O= 6.0 mA		0.17	0.26		0.33		0.4	

جدول ۸-۱

به عنوان مثال مطابق جدول بالا، حداقل ولتاژ ورودی لازم برای سطح یک منطقی برابر 2V است که کمتر از 3.3V می باشد.

از سوی دیگر، ساختار داخلی این بافر که در شکل زیر ارائه شده است نشان می دهد که این بافر از نوع سه حالته (Tri-State) می باشد و از ین خاصیت برای آزاد کردن پایه ی MISO به هنگام برنامه ریزی میکروکنترلر استفاده شده است.



شکل ۸-۲

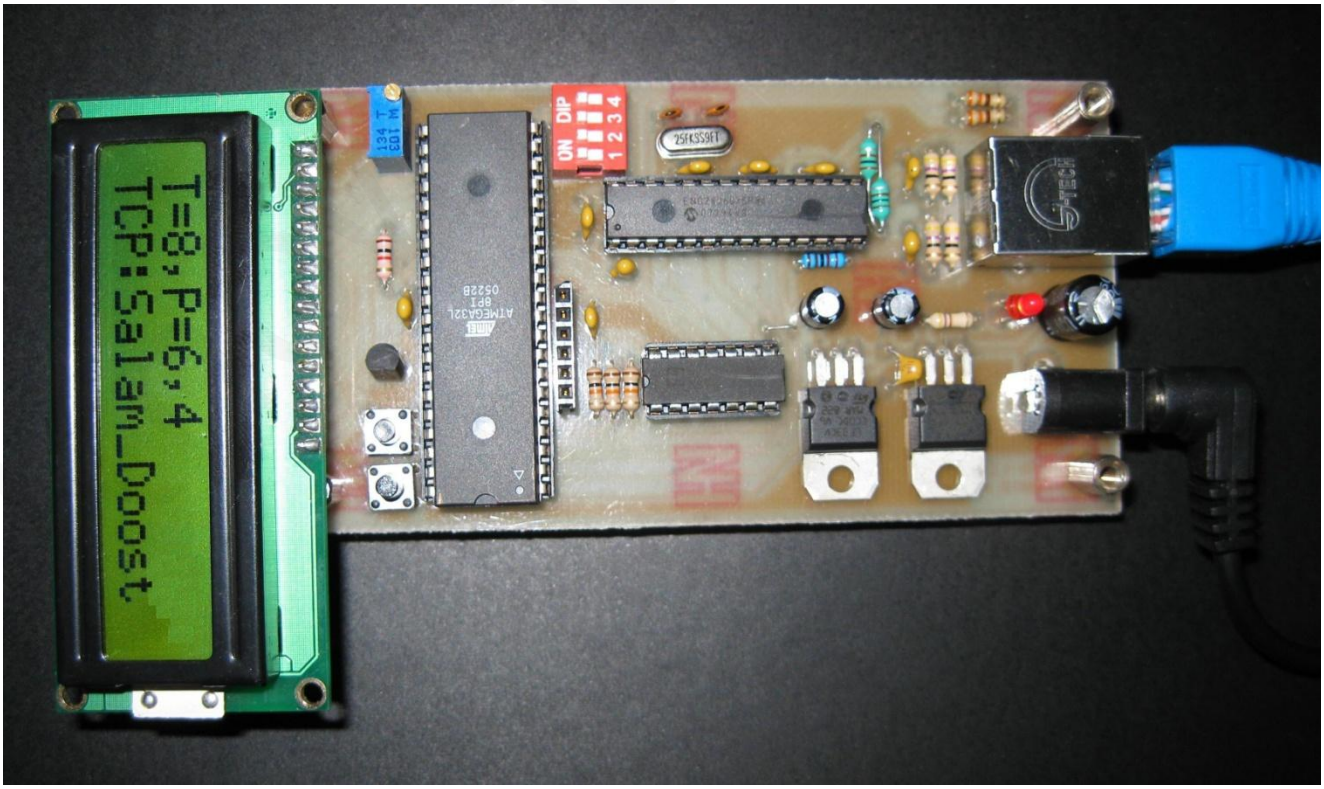
علاوه بر خروجی MISO که از تراشه به میکروکنترلر متصل شده است، خروجی CLKOUT تراشه نیز به عنوان کلاک میکروکنترلر به کار رفته است و از طریق این بافر به پایه XTAL1 میکروکنترلر وصل شده است. با این کار، کلاک میکروکنترلر کسری از فرکانس همان کریستال تراشه شبکه خواهد بود و لزوم استفاده از کریستال مجزا را برای داشتن زمان بندی دقیق رفع می کند. دلیل دیگر این کار، رفع لزوم

استفاده از کلاک سنکرون برای ارتباط SPI در نسخه های قدیمی تراشه ENC28J60 می باشد.

نکته : با توجه به اینکه در این پروژه برای تامین کلاک میکروکنترلر از منبع کلاک خارجی استفاده شده است، باید فیوزبیت های مربوط به کلاک میکروکنترلر با توجه به توضیحات دیتاشیت برنامه ریزی شود.

پایه INT تراشه نیز از طریق بافر به INT2 میکروکنترلر متصل می شود که هنگام دریافت بسته داده از شبکه فعال می گردد. این کار صرفا برای کامل بودن طراحی سخت افزاری انجام شده است و در نرم افزار از این قابلیت استفاده نمی شود و دریافت با کنترل پی در پی وضعیت تراشه تشخیص داده می شود.(روش سرکشی)

LCD استفاده شده یک LCD معمولی 2 × 16 می باشد که به پورت C میکروکنترلر متصل شده است. پایه های ۱۵ و ۱۶ LCD به نور زمینه اختصاص دارند و همان طور که در نقسه شماتیک مدار مشاهده می کنید، پایه منفی آن توسط یک ترانزیستور سوییچ می شود که از پایه PC3 میکروکنترلر فرمان می گیرد. همچنین یک دیپ سوییچ ۴ تایی در مدار تعبیه شده است که وضعیت آن توسط میکروکنترلر خوانده می شود و برای تغییر آدرس های MAC و IP دستگاه به کار می رود. به این ترتیب اتصال ۱۶ دستگاه مشابه در یک شبکه بدون نیاز به تغییر سخت افزاری و نرم افزاری فراهم می آید. نحوه انتخاب آدرس IP و آدرس MAC در بخش نرم افزار توضیح داده خواهد شد. مدار پیاده سازی شده پروژه روی PCB در شکل زیر ارائه شده است.



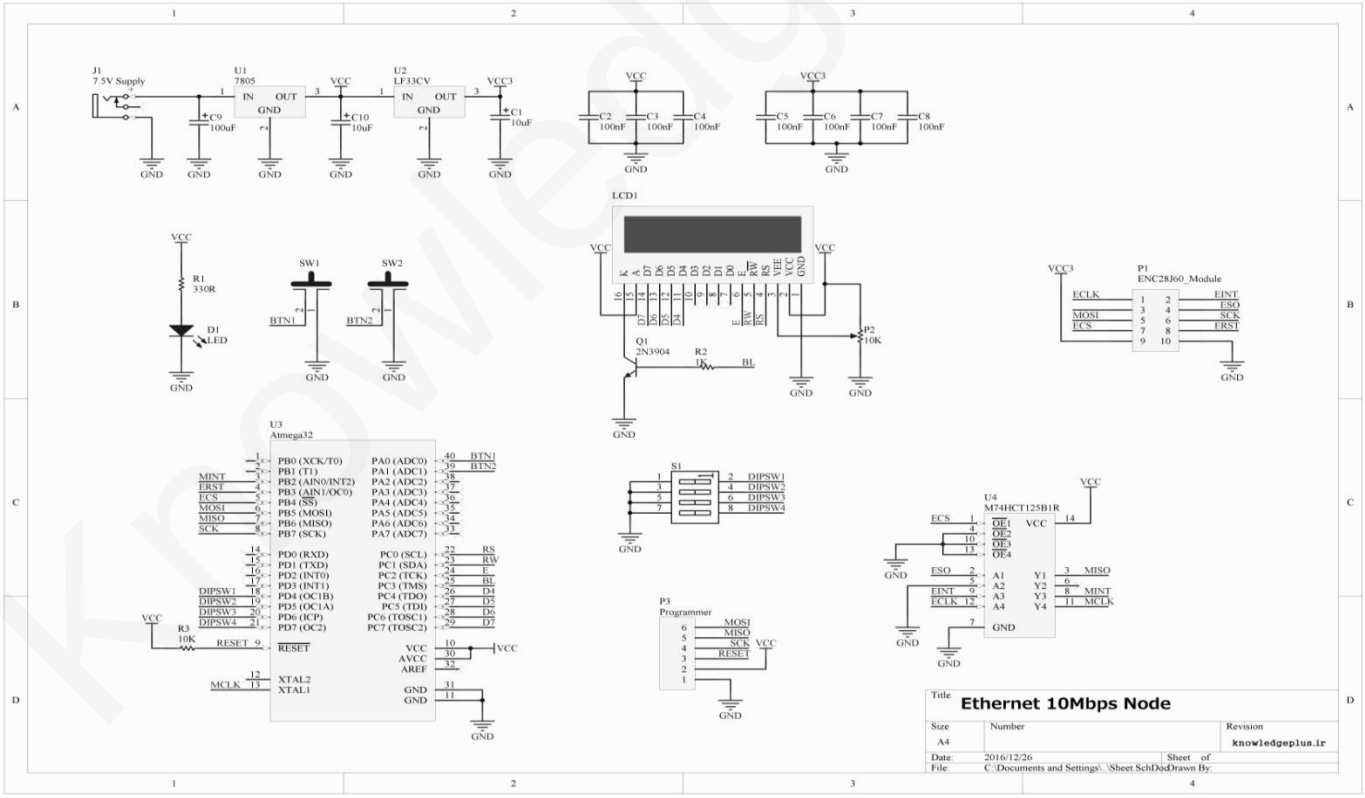
شکل ۸-۳

۸-۲-۲. روش دوم

در این روش از ماژول آماده تراشه ENC28J60 استفاده می شود. شکل زیر یک نمونه از این ماژول ها را نشان می دهد.

شکل ۸-۴

با استفاده از این ماژول مدار لازم برای اتصال به میکروکنترلر مطابق شکل زیر ساده می شود.



شکل ۸-۵

در این روش مطابق شکل صفحه قبل، می توان مدار مناسب برای ارتباط با ماژول را طراحی کرد یا از روش

هایی مانند برد بورد استفاده نمود.

۸-۲-۳. روش سوم

در این روش از برد آماده طراحی شده توسط شرکت ECA برای تراشه ENC28J60 که در شکل زیر نشان داده شده است می توان استفاده نمود.

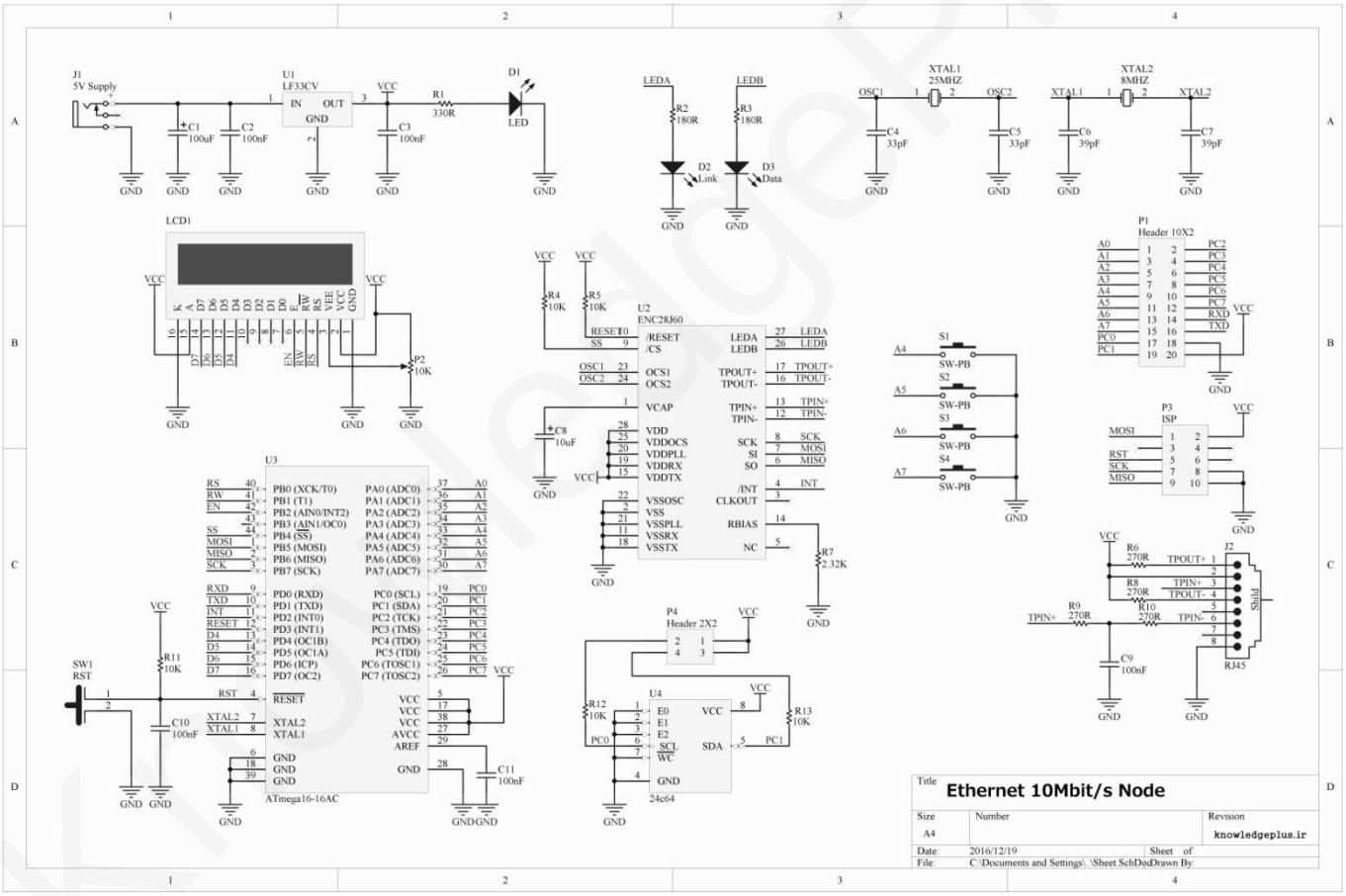
شکل ۸-۶

همچنین می توانید به جای خریداری این برد، از شماتیک و PCB طراحی شده که همراه مقاله ارائه شده است برای ساخت این برد استفاده کنید.

در این برد، میکروکنترلر و تراشه ENC28J60 با تغذیه 3.3v کار می کنند و لذا نیازی به استفاده از بافر مبدل سطح وجود ندارد. برای تکمیل مدار کلاک میکروکنترلر و تراشه ENC28J60 از دو کریستال جداگانه به ترتیب با فرکانس های 8MHz و 25MHz استفاده شده است.

از چهار کلید به عنوان ورودی و LCD کاراکتری به عنوان خروجی در این پروژه استفاده شده است. همچنین امکان اتصال تراشه EEPROM در این سخت افزار تعبیه شده است که به عنوان مثال در کاربردی که ما در این مقاله قصد پیاده سازی آن را داریم، می توانید از EEPROM برای ذخیره سازی صفحات وب یا آدرس های MAC و IP استفاده کنید. همچنین از دیگر ویژگی های این برد وجود پین هدر بر روی پایه های PORTA و PORTC و همچنین دو پایه TXD و RXD میکروکنترلر می باشد که امکان استفاده از این پایه ها را برای کاربردهای مختلف مهیا می سازد. (اتصال سنسورها، خروجی رله و...)

شکل زیر شماتیک طراحی شده برای این برد را نشان می دهد. (شماتیک و PCB به همراه مقاله ارائه شده است)



شکل ۸-۷

نکته : شماتیک و PCB این روش های پیشنهاد شده به همراه مقاله به صورت رایگان ارائه شده است که برای ساخت آنها می توانید تغییرات لازم را با توجه به قطعات موجود در بازار یا امکانات دلخواه انجام بدهید.

۸-۳. نرم افزار پروژه

در این بخش به بررسی سه پروژه به کمک سخت افزارهای طراحی شده در بخش قبل می پردازیم.

۸-۳-۱. پروژه اول

در این پروژه نیاز به دو سخت افزار شامل تراشه ENC28J60 داریم که میتواند به هر یک از روش های گفته شده در بالا ساخته شوند و توسط کابل CAT مناسب این دو سخت افزار به یکدیگر متصل شوند.

در این پروژه فقط قصد ارسال و دریافت بسته های داده را داریم، برای این منظور از توابع نوشته شده برای ارسال و دریافت داده در شبکه استفاده می کنیم و می توان داده های دریافت شده را بر روی LCD نمایش داد.

سه تابع اصلی که برای ارسال و دریافت بسته های داده در شبکه باید استفاده شوند عبارت اند از تابع enc28j60Init ، تابع enc28j60PacketSend و تابع enc28j60PacketReceive .

این توابع هم در فایل ethernet.h در پروژه دوم ارائه شده است و هم در فایل enc28j60 در پروژه سوم وجود دارد.

تابع enc28j60Init برای انجام عملیات پیکربندی اولیه تراشه (مطابق آنچه در فصل هفتم توضیح داده شد) مانند تعیین محل بافر گیرنده و بافر فرستنده، تنظیمات فیلترهای گیرنده، تنظیمات MAC و... استفاده می شود. ورودی این تابع آدرس MAC سخت افزار می باشد که باید تنظیم شود. پس از فراخوانی این تابع، می توانیم به ارسال و دریافت بسته های داده در شبکه بپردازیم، برای این منظور از دو تابع enc28j60PacketSend و enc28j60PacketReceive استفاده می کنیم.

از تابع enc28j60PacketReceive برای دریافت بسته های داده استفاده می کنیم. این تابع دارای دو ورودی است، ورودی اول متغیر maxlen می باشد که تعیین کننده ی حداکثر اندازه بسته دریافتی است و ورودی دوم یعنی متغیر اشاره گر packet آدرس محلی است که باید بسته ی دریافتی در آن ذخیره شود. خروجی تابع نیز طول بسته دریافتی برحسب بایت می باشد و در صورت عدم وجود بسته جدید برای دریافت، مقدار صفر بازگردانده می شود.

از تابع enc28j60PacketSend برای ارسال بسته های داده به شبکه استفاده می کنیم. این تابع

دارای دو ورودی است، ورودی اول متغیر len می باشد که تعیین کننده ی طول بسته ای است که باید ارسال شود و ورودی دوم یعنی متغیر اشاره گر packet که داده ای است که قصد ارسال آن را در شبکه داریم.

۸-۳-۲. پروژه دوم

در این پروژه قصد پیاده سازی یک وب سرور ساده را داریم. برای این منظور در بخش سخت افزار باید یکی از سخت افزار های پیشنهاد شده در بخش اول را توسط کابل CAT مناسب به کامپیوتر یا سوییچ شبکه متصل کنیم. (کدها مطابق سخت افزار پروژه شماره ۲ نوشته شده است اما میتوان با تغیر ثوابت و ماکرو های برنامه از آن در آرایش های سخت افزاری دیگر نیز استفاده نمود)

در بخش نرم افزار نیز باید پروتکل های لایه های بالاتر یعنی HTTP ، TCP ، IP ، ARP و UDP نیز پیاده سازی شوند. هر یک از این پروتکل ها وظایفی دارند که در فصل پنجم با این وظایف و نحوه استفاده از این پروتکل ها آشنا شدیم.

این پروژه دارای یک فایل اصلی main.c می باشد که ساختار اصلی برنامه در آن قرار گرفته است و بر روی میکروکنترلر پروگرام می شود. همان طور که در ابتدای این فصل اشاره شد، پیاده سازی توابع مربوط به تراشه ENC28J60 و پروتکل های ارتباطی (به غیر از پروتکل HTTP) در این پروژه در فایل ethernet.h صورت پذیرفته است و کافی است این فایل به برنامه با دستور #include اضافه شود و به راحتی می توان از توابع آن در برنامه استفاده نمود.

در این پروژه به غیر از توضیحاتی که در مورد پروتکل HTTP می دهیم، قصد ورود به جزئیات پیاده سازی پروتکل ها را نداریم. در پروژه سوم در مورد پیاده سازی این پروتکل ها توضیحات بیشتری ارائه می دهیم.

نکته : برای پیاده سازی صفحه وب و ارتباط با شبکه برای به روز کردن صفحه وب، باید پروتکل HTTP از لایه ی Application پیاده سازی شود. این پروتکل در کدهای این پروژه در حلقه while(1) برنامه اصلی (فایل main.c) پیاده سازی شده است اما در کدهای پروژه سوم برای پیاده سازی این پروتکل مانند دیگر پروتکل ها کتابخانه ای جداگانه ای ارائه شده است.

برای استفاده از فایل ethernet.h در کامپایلر CodeVisionAVR آن را در همان شاخه ای که پروژه خود را در آن ساخته اید کپی کنید.

نکته : معمولا کتابخانه های به کار رفته در زبان C دارای یک فایل h. و یک فایل lib. یا c. می باشند که باید در تنظیمات پروژه آنها را نیز به پروژه اضافه کنیم. به دلیل ساده سازی نحوه استفاده از کتابخانه Ethernet در این مقاله، پیاده سازی توابع در همان فایل ethernet.h انجام می شود و افزودن فایل دیگری به پروژه لازم نیست.

برای آشنایی بیشتر با این کتابخانه، ابتدا نحوه استفاده از آن به طور کلی بیان می شود و در ادامه پس از معرفی توابع مورد نیاز این کتابخانه، جزئیات استفاده از آنها در شرح برنامه ارائه می گردد.

مراحل استفاده از کتابخانه به طور خلاصه عبارت است از :

۱- مشخص کردن (Define) درگاه و پایه های میکروکنترلر که به تراشه متصل شده اند، براساس وضعیت سخت افزار سیستم

۲- اضافه کردن فایل ethernet.h

۳- فراخوانی تابع ethernet_init قبل از حلقه دائمی برنامه

۴- راه اول، سرکشی (Polling) : فراخوانی تابع ethernet_process درون حلقه دائم برنامه

راه دوم، وقفه (Interrupt) : فراخوانی تابع ethernet_process درون سرویس وقفه متناسب پایه وقفه تراشه

۵- تعریف توابع ethernet_tcp و ethernet_udp

در واقع پیاده سازی توابع ethernet_tcp و ethernet_udp تنها بخشی است که برنامه نویس با آن درگیر می شود.

نکته : در صورت استفاده از وقفه، پایه INT2 باید در حالت حساس به لبه پایین رونده تنظیم شود.

در ادامه برخی از توابع مهم موجود در کتابخانه ethernet.h را معرفی می کنیم.

تابع ethernet_init :

```
void ethernet_init(unsigned char *dev_mac, unsigned char *dev_ip, unsigned int port_tcp, unsigned int port_udp)
```

این تابع برای آماده سازی سخت افزار و نرم افزار تعریف شده است. ورودی های آن به ترتیب یک آرایه ۶ بایتی برای آدرس MAC دستگاه، یک آرایه ۴ بایتی برای آدرس IP دستگاه و شماره پورت TCP و شماره پورت UDP می باشند.

تابع ethernet_process :

این تابع برای پردازش بسته های دریافتی از شبکه تعریف شده است و ورودی نمی پذیرد و باید در یک حلقه دائم فراخوانی شود. با این کار، تمامی بسته ها در صورت نیاز پاسخ (Response) داده می شوند و بسته های TCP و UDP مورد نظر به توابع Ethernet_udp و Ethernet_tcp باز می گردند.

توابع concatstr و concatstrf :

```
unsigned int concatstr(char *buf, unsigned int pos, const char *s)
unsigned int concatstrf(char *buf, unsigned int pos, const flash char *s)
```

این توابع برای آسان سازی اضافه کردن یک رشته حرفی با انتها (Null Terminated ASCII String) به یک بافر تعریف شده اند. در اینجا buf* به ابتدای بافر خروجی اشاره می کند که رشته s* باید از اندیس pos به بعد در آن قرار گیرد. خروجی تابع نیز طول جدید بافر خروجی است و میتواند به عنوان اندیس برای اضافه کردن رشته های بعدی نیز به کار رود. تنها تفاوت این دو تابع، نوع ورودی رشته است که اولی برای رشته های تعریف شده در حافظه SRAM و دومی برای رشته های تعریف شده در حافظه Flash مورد استفاده قرار می گیرند.

تابع Ethernet_udp :

```
unsigned char ethernet_udp(unsigned char* data, unsigned char data_len)
```

این تابع باید توسط برنامه نویس برای پاسخگویی به درخواست های UDP تعریف شود و در صورت نیاز توسط تابع Ethernet_process و نه برنامه نویس فراخوانی گردد. توجه داشته باشید که این تابع حتی اگر بسته های UDP به پاسخگویی نیازی نداشته باشند نیز باید تعریف گردد و با فراخوانی آن، فقط

خروجی صفر بازگردانده می شود.

اولین ورودی تابع اشاره گر data* است که به ابتدای داده های دریافتی اشاره دارد. ورودی دوم، متغیر data_len است که طول داده دریافتی را مشخص می کند. در این تابع فرض می شود که پاسخ از طریق همان بافر درخواست یعنی data* داده می شود و طول پاسخ به عنوان خروجی تابع بازگردانده می شود.

نکته : بافر ورودی و خروجی برای کاهش مصرف حافظه به صورت اشتراکی تعریف شده است و در صورت نوشتن درون بافر، اطلاعات ورودی بازنویسی می شوند و معتبر نخواهند بود. به همین دلیل باید قبل از نوشتن خروجی ها در بافر، تمامی اطلاعات مورد نیاز از بافر استخراج شوند.

تابع Ethernet_tcp :

```
unsigned int ethernet_tcp(unsigned char* request, unsigned char* response);
```

این تابع باید توسط برنامه نویس برای پاسخگویی به درخواست های TCP تعریف گردد و در صورت نیاز توسط تابع Ethernet_process و نه برنامه نویس فراخوانی شود. توجه داشته باشید که این توابع حتی در صورتی که بسته های TCP به پاسخگویی نیازی نداشته باشند نیز باید تعریف گردد و با فراخوانی آن فقط خروجی صفر بازگردانده می شود.

ورودی های این تابع به صورت اشاره گرهایی به بافر ورودی و خروجی می باشند. در این تابع نیز خروجی طول بافر پاسخ است. طول درخواست به عنوان ورودی به این تابع وارد نمی شود زیرا درخواست از نوع رشته حرفی با انتهاست و دانستن طول آن لازم نیست.

توجه : مشابه تابع قبل، در تعریف بافرهای این تابع نیز از حافظه اشتراکی استفاده شده است که برنامه نویس را به برداشت اطلاعات مفید درخواست قبل از نوشتن در پاسخ محدود می کند.

۸-۳-۲-۱. شرح کلی برنامه

در ابتدای برنامه مانند هر پروژه دیگر، سرفایل متناسب با میکروکنترلر مورد استفاده اضافه می شود. سپس سایر کتابخانه هایی را که به توابع انها نیاز داریم، اضافه می کنیم.

در ادامه برای استفاده از LCD، درگاهی از میکروکنترلر را که LCD به آن متصل است مشخص می نماییم و سپس فایل lcd.h افزوده می شود. این کار در صورتی که برای ساخت پروژه از CodeWizard

استفاده شود، به طور خودکار انجام می گیرد.

قبل از اضافه کردن فایل ethernet.h ، درگاهی از میکروکنترلر که قابلیت ارتباط SPI دارد همراه شماره پایه های مربوط به آن درگاه به صورت تعدادی #define معرفی می گردند. در اینجا همان طور که در نقشه شماتیک مدار مشاهده مشخص می باشد پایه های MISO ، MOSI ، SCK و CS معادل پایه های ۶، ۵، ۷ و۴ پورت B می باشند و به همین دلیل تعاریف مطابق با آنها صورت می گیرند.

پس از آن، مقدار حافظه ای از میکروکنترلر که قرار است برای پردازش بسته های داده مورد استفاده قرار گیرد، توسط عبارت #define BUFFER_SIZE تعیین می شود که متناسب با طول بسته ها می باشد. برای پروژه هایی مشابه این پروژه، یک کیلوبایت مقدار معقول و مناسبی به نظر می رسد، اگرچه انجام آن برای برخی میکروکنترلرها مانند ATmega8 ممکن نیست و باید مقدار کمتری، مثلا ۵۱۲ بایت در نظر گرفته شود. توجه داشته باشید که در چنین شرایطی، میکروکنترلر قابلیت دریافت درخواست های طولانی (Long Requests) و پاسخ های طولانی (Long Responses) به شبکه را نخواهد داشت.

پس از تعیین پورت SPI و میزان حافظه تخصیصی، فایل ethernet.h افزوده می شود. معمولا برای راحتی کار و نیاز نداشتن به تغییر کل برنامه در صورت جا به جا شدن برخی از قطعات جانبی میکروکنترلر مانند کلیدهای فشاری، آنها را در ابتدای برنامه به صورت #define مشخص می کنند. در اینجا نیز، دو پایه در نظر گرفته شده برای کلید فشاری و پایه های متصل به دیپ سوییچ ۴ تایی به صورت ماکرو تعریف شده اند. توجه داشته باشید که کلید B1 هنگام به کار رفتن در متن برنامه به صورت یک بیت فعال با نام PUSHBUTTON1 دیده می شود، بدین معنا که در صورت فشار دادن کلید شماره ۱، پایه مربوطه که در حالت عادی در وضعیت یک منطقی است، به زمین متصل می شود و در نتیجه پایه صفر خوانده می شود و نهایتا با توجه به نحوه تعریف PUSHBUTTON1 که نقیض است، کل عبارت ارزش منطقی درست را خواهد داشت. به همین شکل، مقدار DIPSWITCH از شیفت یافتن پایه های ۴ تا ۷ پورت D که برای دیپ سوییچ در نظر گرفته شده اند، به دست می آید و عددی از صفر تا ۱۵ را در بر می گیرد.

همان طور که می دانیم، هر دستگاه برای قرارگیری در شبکه باید عددی به عنوان شماره اختصاصی خود (آدرس MAC) داشته باشد تا از سایر ادواتی که در شبکه قرار دارند، متمایز شود. بدین منظور هر سازنده، مجوز استفاده از محدوده ای از اعداد را کسب می کند و درستگاه های تولید خود را به شناسه ۴۸ بیتی آدرس MAC تجهیز می نماید.

در اینجا تولید انبوه دستگاه مد نظر نیست و فقط اتصال ادوات کوچک و جزئی به شبکه انجام می گیرد و بدون گرفتن مجوز، مقداری دلخواه برای MAC در نظر گرفته می شود و در صورتی که با یک دستگاه

دیگر تداخل پیدا کند، به راحتی و با تغییر مقدار دیپ سوییچ دستگاه بدون مزاحمت در شبکه قرار می گیرد.

آدرس MAC در نظر گرفته شده در متغیر mymac که آرایه ای ۶ بایتی است و مقدار IP دستگاه نیز در متغیر myip که آرایه ای ۴ بایتی است ذخیره شده اند.

فاصله زمانی دو درخواست (چه UDP و چه TCP) با استفاده از متغیرهای time_cs و tims_s که به ترتیب صدم ثانیه و ثانیه را اندازه می گیرند، محاسبه شده است. متغیرهای counter1 و counter2 نیز دفعات فشرده شدن کلیدهای فشاری B1 و B2 را در خود ذخیره می نمایند.

مشاهده می کنید که یک سرویس وقفه برای سر ریز (Overflow) تایمر ۱ در نظر گرفته شده است و با توجه به تنظیمات تایمر ۱، یک بار در هر 10ms فعال می گردد. در این سرویس وقفه، متغیرهای اندازه گیری زمان به روز می شوند و وضعیت کلیدهای فشاری نیز بررسی می گردد و در صورت فشار داده شدن، متغیرهای مربوطه افزایش می یابند. به دلیل بررسی دائم کلیدها در فواصل زمانی مشخص، احتمال شمرده نشدن آنها به حداقل می رسد. در ضمن در بخش سخت افزاری، هیچ گونه مداری برای حذف پرش کلید (Debounce) در نظر گرفته نشده است و در صورت لزوم می توان آن را به صورت نرم افزاری انجام داد.

در تابع اصلی برنامه، تایمر سگ نگهبان (Watchdog Timer) به عنوان اولین کار فعال شده است تا در صورت توقف ناگهانی برنامه (Hang)، میکروکنترلر را reset کند و مدار به وضعیت های نامعلوم دچار نشود. برای جلوگیری از reset ناخواسته، در نقاط مشخصی از برنامه که هنگام اجرا به صورت مداوم فراخوانده می شوند، مقدار تایمر سگ نگهبان دوباره صفر می گردد. در صورتی که سیستم در بخشی از برنامه متوقف شود، تایامر سگ نگهبان صفر می گردد و به reset میکروکنترلر منجر می شود.

در ادامه تمامی پایه های میکروکنترلر در وضعیت Pullup ضعیف قرار می گیرند. این کار بی خطرترین راه برای جلوگیری از معلق ماندن (float) ماندن پایه های بدون ورودی و سایر پایه هایی می باشد که قرار است در ادامه تعیین وضعیت (ورودی/خروجی) شوند.

تایمر ۱ در مد reset خودکار با کلاک سیستم به نحوی قرار می گیرد که در هر 10ms یک بار، یک وقفه سر ریز را فعال کند. بدین منظور با توجه به کلاک سیستم، مقداری که باید برای حداکثر شمارش تعیین شود از رابطه زیر به دست می آید که معادل F423H است و در رجیستر ICR1 نوشته می شود.

$$f_{CLK} = \frac{f_{XTAL}}{4} = \frac{25.0MHz}{4} = 6.5MHz = 6250000Hz$$

$$n = f_{CLK} \times T = 6250000Hz \times \frac{10}{1000}s = 62500$$
$$ICR1 = n - 1 = 62499 = 0xF423$$

در ابتدا مقایسه کننده آنالوگ داخلی میکروکنترلر نیز که در اینجا هیچ نقشی ندارد، برای کاهش مصرف خاموش می شود. سپس LCD راه اندازی و چراغ پس زمینه روشن می شود.

در این مرحله، مقادیر IP و MAC بر اساس پیش فرض و مقدار دیپ سوییچ تعیین می شوند و با استفاده از آنها تراشه شبکه راه اندازی می گردد. در اینجا مقدار پیش فرض برای IP برابر 192.168.1.10 است و با مقدار دیپ سوییچ که عددی بین صفر تا ۱۵ می باشد و توسط کاربر تعیین می شود، جمع می گردد. به این ترتیب محدوده IP از مقدار پیش فرض تا 192.168.1.25 تنظیم پذیر است. بایت چهارم مقدار MAC نیز که به صورت پیش فرض 0x112233445566 می باشد، با مقدار دیپ سوییچ جمع می گردد و MAC دستگاه را مشخص می کند.

نکته : تغییر دیپ سوییچ پس از راه اندازی دستگاه، هیچ تاثیری نخواهد داشت.

پس از راه اندازی دستگاه آدرس MAC و آدرس IP روی LCD نمایش داده می شوند و سیستم با فعال سازی وقفه ها، کار عادی خود را آغاز می کند.

در حلقه دائم اجرای برنامه، تایمر سگ نگهبان reset می شود و پس از آن متغیرهای زمان و تعداد فشار داده شدن کلیدهای فشاری، روی LCD نشان داده می شوند. مهم ترین کاری که در حلقه دائم برنامه انجام می شود، فراخوانی تابع Ethernet_process است که به منظور پردازش بسته های داده دریافتی از شبکه صورت می گیرد.

همان طور که توضیح داده شد، برنامه نویس باید دو تابع برای پردازش بسته های UDP و TCP بنویسد. تابع Ethernet_udp نوشته شده در اینجا، ابتدا اطلاعات دریافتی را روی LCD نمایش می دهد، سپس مقدار تایمر و دو شمارنده کلیدها را به شکل متنی در بافر می نویسد و در پایان نیز طول بافر نوشته شده را به عنوان خروجی تابع باز میگرداند.

کاری که در تابع Ethernet_tcp انجام می گیرد اندکی پیچیده تر از Ethernet_udp است. ابتدا اطلاعات دریافتی بررسی می شوند تا مشخص شود که دستور ورودی، دستور GET در HTTP است یا خیر. در صورتی که جواب منفی باشد، مقدار صفر بازگردانده می شود که به معنی نداشتن پاسخ برای درخواست است. در غیر این صورت، عبارت بعد از GET روی LCD نمایش داده می شود و سپس عبارتی متنی به شکل HTML بر اساس مقادیر زمان سنج و شمارنده تولید و درون بافر پاسخ نوشته می شود.

نحوه استفاده از تابع cocatstr و concatstrf برای انجام این کار به خوبی نمایان است. طول بافر نوشته شده نیز در انتها تابع بازگردانده می شود.

باید توجه داشت که می توان بخش های ثابت پیام متنی به قالب HTML را به صورت ثابت در حافظه Flash ذخیره کرده و یکباره آن را به کمک تابع concatstrf به بافر خروجی منتقل نمود.

در بخش بعدی، ابتدا به توضیح کدهای فایل main.c و سپس به توضیح کدهای کتابخانه ethernet.h می پردازیم.

۸-۳-۲-۲. فایل main.c

```
#include <mega32.h>
#include <stdio.h>
#include <string.h>
#include <delay.h>
```

اضافه کردن کتابخانه های میکروکنترلر Mega32 ، stdio (توابع استاندارد ورودی و خروجی) ، string (توابع رشته ای) و delay (توابع ایجاد تاخیر).

```
#asm                    // LCD Module
  .equ __lcd_port=0x15        // PORTC
#endasm
```

تعریف PORTC برای اتصال به LCD کاراکتری.

```
#include <lcd.h>
```

اضافه کردن کتابخانه lcd .

```
#define ENC28J60_SPI_PORT      PORTB
#define ENC28J60_SPI_DDR       DDRB
#define ENC28J60_SPI_MISO      6
#define ENC28J60_SPI_MOSI      5
#define ENC28J60_SPI_SCK       7
#define ENC28J60_SPI_CS        4
```

تعریف کردن ماکرو برای پایه هایی که به تراشه ENC28J60 متصل می شوند.

```
// Ethernet buffer size
#define BUFFER_SIZE         1024
```

تعیین کردن ظرفیت بافر مورد نیاز برای پردازش بسته های داده در حافظه میکروکنترلر که در اینجا ۱۰۲۴ بایت یا ۱ کیلوبایت تعریف شده است.

```
#include "ethernet.h"
```

اضافه کردن کتابخانه Ethernet برای استفاده از توابع آن.

```
#define LCD_BL_ON           PORTC.3 = 1
#define LCD_BL_OFF          PORTC.3 = 0
#define DIPSWITCH           (PIND>>4)
#define PUSHBUTTON1         (!PINA.0)
#define PUSHBUTTON2         (!PINA.1)
```

تعریف ماکرو برای پایه های روشن و خاموش کردن نور backlight مربوط به LCD (LCD_BL_ON و LCD_BL_OFF)، پایه ورودی متصل به دیپ سوییچ (DIPSWITCH) و پایه های متصل به کلیدهای تکی. (PUSHBUTTON1 و PUSHBUTTON2)

```
unsigned char   mymac[6] = {0x11,0x22,0x33,0x44,0x55,0x66};    // MAC Address
```

تعیین آدرس MAC دستگاه در یک آرایه ۶ بایتی.

```
unsigned char   myip[4] = {192,168,1,10};               // IP Address
```

تعریف آدرس IP دستگاه در یک آرایه ۴ بایتی.

```
unsigned char   time_cs = 0;        // 1/100 second counter
unsigned int    time_s = 0;         // seconds counter
```

متغیرهای شمارنده زمانی برای ایجاد فاصله بین دو درخواست متوالی (چه UDP و چه TCP)، متغیر time_cs برای شمارش صدم ثانیه و متغیر time_s برای شمارش ثانیه.

```
unsigned int    counter1 = 0;       // button 1 clicks
```

```
unsigned int    counter2 = 0;        // button 2 clicks
```

تعریف متغیرهای counter1 و counter2 برای ذخیره دفعات فشرده شدن کلیدهای فشاری B1 و B2 .

```
interrupt [TIM1_OVF] void timer1_ovf_isr(void)  // Timer 1 overflow ISR
```

شروع بخش کدهای subroutine یا زیر روال سرویس وقفه که با اتفاق افتادن وقفه سر ریز تایمر ۱ طبق تنظیمات انجام شده توسط رجیسترهای تایمر ۱، یک بار در هر 10ms این کدها اجرا می شوند.

```
static bit old_btn1 = 0, old_btn2 = 0;
```

تعریف متغیرهای نگه دارنده وضعیت فشرده شدن کلیدهای B1 و B2 از نوع static به منظور حفظ مقدار قبلی در هر بار فراخوانی تابع.

```
    if (++time_cs==100)          // one second elapsed
    {
        time_cs = 0;
        time_s ++;
    }
```

در صورتی که متغیر time_cs یا صدم ثانیه برابر عدد ۱۰۰ شود، با توجه به این که وقفه سر ریز تایمر هر ۱۰ میلی ثانیه اتفاق می افتد، به معنای گذشت زمان ۱ ثانیه است، به همین منظور متغیر time_cs برابر صفر شده (برای شمارش مجدد) و متغیر time_s به نشانه گذشت یک ثانیه، یک واحد افزایش پیدا می کند.

```
    if (PUSHBUTTON1)
    {
        if (!old_btn1)
            counter1 ++;
        old_btn1 = 1;
    }
    else
        old_btn1 = 0;
    if (PUSHBUTTON2)
```

```
{
    if (!old_btn2)
        counter2 ++;
    old_btn2 = 1;
}
else
    old_btn2 = 0;
```

بررسی فشرده شدن کلیدهای B1 و B2 و به روز رسانی متغیرهای مربوط به آنها در این بخش صورت می پذیرد. متغیرهای old_btn1 و old_btn2 نشان دهنده ی وضعیت قبلی فشرده شدن کلیدها هستند، بدین معنا که اگر پس از فراخوانی این بخش، اگر این دو متغیر یک باشند به این معنا می باشد که دست کاربر بر روی کلیدها باقی مانده است و نباید عمل شمارش صورت بپذیرد. در واقع با بررسی کردن این مسئله از شمارش متوالی زمانی که دست کاربر بر روی کلیدها می باشد جلوگیری به عمل می آید.

```
void main(void)
```

در این بخش کدهای بدنه اصلی برنامه قرار گرفته اند.

```
uint8_t str[20]="";    // LCD Buffer
```

تعریف بافر برای فرمت بندی و ارسال رشته ها به LCD از نوع unsigned char

نکته : در فایل ethernet.h با دستور typedef نام متغیرها برای سادگی تغییر داده شده است.

```
// Watchdog Timer Enabled with Prescaler: OSC/512k
WDTCR=0x0D;
```

فعال سازی تایمر سگ نگهبان برای جلوگیری از هنگ کردن برنامه

```
    // All I/O pins are pulled up
    PORTA=0xFF;
    DDRA=0x00;
    PORTB=0xFF;
    DDRB=0x00;
    PORTC=0x00;
```

```
DDRC=0x04;

PORTD=0xFF;

DDRD=0x00;
```

تعریف پورت های A و B و D به عنوان ورودی و فعال سازی مقاومت های pull-up داخلی آنها، تنظیم پورت C برای اتصال به LCD

```
// Timer 1 interrupts every 10 ms

TCCR1A=0x02;

TCCR1B=0x19;

ICR1H=0xF4;

ICR1L=0x23;
```

تنظیم رجیسترهای تایمر ۱ با توجه به فرکانس کلاک میکرو برای سرریز شدن در هر ۱۰ میلی ثانیه

```
// Timer(s) Interrupt(s) initialization

TIMSK=0x04;
```

فعال سازی وقفه تایمر ۱

```
// Analog Comparator: Off

ACSR=0x80;
```

غیرفعال سازی مقایسه کننده آنالوگ داخلی میکروکنترلر

```
// LCD module initialization

lcd_init(16);
```

تنظیم LCD کاراکتری با ۱۶ ستون

```
LCD_BL_ON;
```

روشن کردن چراغ backlight مربوط به LCD

```
#asm("wdr");
```

ریست کردن تایمر سگ نگهبان

```
// setting MAC & IP using dip-switch state
```

```
mymac[0] += DIPSWITCH;
myip[3] += DIPSWITCH;
```

تعیین آدرس MAC و آدرس IP بر اساس پیش فرض و مقدار دیپ سوییچ

بایت چهارم مقدار MAC نیز که به صورت پیش فرض 0x112233445566 می باشد (تنظیم شده در خطوط بالای برنامه)، با مقدار دیپ سوییچ جمع می گردد و آدرس MAC دستگاه را مشخص می کند. مقدار پیش فرض برای IP برابر 192.168.1.10 است (که در خطوط بالا تنظیم شد) و با مقدار دیپ سوییچ که عددی بین صفر تا ۱۵ می باشد و توسط کاربر تعیین می شود، جمع می گردد. به این ترتیب محدوده IP از مقدار پیش فرض تا 192.168.1.25 تنظیم پذیر است.

```
// Ethernet Initializations
ethernet_init(mymac, myip, 80, 1200);
```

فراخوانی تابع ethernet_init برای انجام تنظیمات اولیه تراشه ENC28J60 با متغیرهای mymac برای آدرس MAC ، myip برای آدرس IP ، شماره پورت TCP برابر ۸۰ و شماره پورت UDP برابر ۱۲۰۰

```
// start up display
lcd_gotoxy(0,0);
lcd_putsf("    MAC     ");
```

نوشتن عبارت MAC بر روی سطر اول LCD

```
lcd_gotoxy(0,1);
sprintf(str, " %02x%02x-%02x%02x-%02x%02x ",
    mymac[0], mymac[1], mymac[2], mymac[3], mymac[4], mymac[5]);
```

نوشتن آدرس MAC در متغیر mymac در بافر رشته ای str که در خطوط بالا تعریف شد به منظور تبدیل آدرس MCA به متغیر رشته ای برای ایجاد امکان نمایش آن بر روی LCD

```
lcd_puts(str);
```

نمایش آدرس MAC در سطر دوم و ستون اول LCD

```
delay_ms(2000);
```

تاخیر ۲ ثانیه برای نمایش اطلاعات LCD به کاربر

```
lcd_gotoxy(0,0);
lcd_putsf("     IP     ");
```

نوشتن عبارت IP بر روی سطر اول LCD

```
lcd_gotoxy(0,1);
sprintf(str, "%03u.%03u.%03u.%03u",
    myip[0], myip[1], myip[2], myip[3]);
```

نوشتن آدرس IP در متغیر myip در بافر رشته ای str که در خطوط بالا تعریف شد به منظور تبدیل آدرس IP به متغیر رشته ای برای ایجاد امکان نمایش آن بر روی LCD

```
lcd_puts(str);
```

نمایش آدرس IP در سطر دوم و ستون اول LCD

```
delay_ms(3000);
```

تاخیر ۳ ثانیه برای نمایش اطلاعات LCD به کاربر

```
lcd_clear();
```

پاک کردن LCD

```
// Global enable interrupts
#asm("sei")
```

فعال سازی وقفه همگانی

```
while (1)
```

کدهای در این بخش دائما اجرا می شوند.

```
#asm("wdr");
```

ریست کردن تایمر سگ نگهبان

```
sprintf(str, "T=%u,P=%u,%u ", time_s, counter1, counter2 );
```

نوشتن متغیرهای time_s ، counter1 و counter2 در بافر رشته ای str به منظور نمایش بر روی LCD

```
lcd_gotoxy(0,0);
lcd_puts(str);
```

نمایش متغیرها بر روی سطر اول و ستون اول LCD

```
// process ethernet transactions
ethernet_process();
```

اجرای تابع ethernet_process به منظور پردازش بسته های داده

در این بخش کدهای تابع ethernet_udp را توضیح می دهیم.

```
unsigned char ethernet_udp(unsigned char* data, unsigned char data_len)
```

تعریف تابع برای پردازش درخواست های UDP، با دو آرگومان برای داده های دریافتی (data) و طول بافر نوشته شده (data_len)

```
unsigned char str[20];
```

تعریف متغیر رشته ای

```
lcd_gotoxy(0,1);
lcd_putsf("UDP:        ");
```

نوشتن عبارت UDP بر روی سطر دوم و ستون اول LCD

```
strncpy(str, data, 11);
```

استفاده از تابع strncpy (در کتابخانه string) برای ریختن داده متنی موجود در متغیر data در متغیر str با طول ۱۱ کاراکتر

```
lcd_gotoxy(4,1);
lcd_puts(str);
```

نمایش رشته موجود در متغیر str بر روی سطر دوم و ستون پنجم LCD

```
sprintf(str, "T=%u,P=%u,%u\n", time_s, counter1, counter2 );
```

ذخیره سازی متغیرهای time_s ، counter1 و counter2 در متغیر رشته ای str برای نمایش بر روی LCD

```
data_len = concatstr(data, 0, str);
```

اتصال متغیر str به متغیر data از کاراکتر اول و به دست آوردن طول متغیر حاصل شده و ذخیره آن در متغیر data_len توسط تابع concatstr (در کتابخانه ethernet)

```
time_cs = 0;
time_s = 0;
```

صفر کردن شمارنده های زمان

```
return data_len;
```

فرستادن طول بافر داده به عنوان خروجی تابع

در این بخش کدهای تابع ethernet_tcp را توضیح می دهیم.

```
unsigned int ethernet_tcp(unsigned char* request, unsigned char* response)
```

تعریف تابع برای پردازش درخواست های TCP، با دو آرگومان ورودی برای درخواست و پاسخ. در این تابع نیز خروجی طول بافر پاسخ می باشد.

```
unsigned int len;
unsigned char str[20];
```

تعریف متغیر len و متغیر رشته ای str

```
if (strncmpf(request, "GET /", 5)!=0)
return 0;
```

توسط تابع strncmpf کتابخانه string ، محتویات ۵ کاراکتر از متغیر request با عبارت "GET /" مقایسه می شود و در صورتی که برابر نباشند (خروجی تابع strncmpf صفر نباشد)، تابع ethernet_tcp مقدار صفر را به عنوان پاسخ باز می گرداند.

```
lcd_gotoxy(0,1);
lcd_putsf("TCP:          ");
```

نوشتن عبارت TCP بر روی سطر دوم و ستون اول LCD

```
strncpy(str, request+5, 11);
```

کپی کردن ۱۱ کاراکتر از تابع request (که ۵ واحد افزایش پیدا کرده است) در متغیر str

```
lcd_gotoxy(4,1);
lcd_puts(str);
```

نوشتن محتویات متغیر str بر روی سطر دوم و ستون پنجم LCD

```
len = concatstrf(response, 0, "HTTP/1.0 200 OK\r\n");
len = concatstrf(response, len, "Content-Type: text/html\r\n");
len = concatstrf(response, len, "Pragma: no-cache\r\n\r\n");
len = concatstrf(response, len,
    "<HTML><TITLE>Ethernet Implementation </TITLE><BODY><CENTER>");
len = concatstrf(response, len,
    "<P>Transferring Data via Ethernet</P>\n<HR>");
len = concatstrf(response, len,
    "<P>Time = <FONT COLOR=#00FF00> ");
```

در این خطوط از تابع concatstrf در کتابخانه ethernet برای نوشتن پاسخ در متغیر response استفاده شده است.

دستورات نوشته شده مربوط به پیاده سازی پروتکل HTTP و زبان HTML برای طراحی صفحه وب برای نمایش بر روی صفحه کامپیوتر است. در واقع این دستورات در متغیر response نوشته شده و برای کامپیوتر ارسال می شود و حاصل آن در کامپیوتر صفحه وب می باشد که مشخصات آن توسط این دستورات مشخص شده است.

عبارت HTTP/1.0 200 OK\r\n که در ابتدای متغیر response نوشته می شود و طول عبارت نوشته شده در این متغیر در متغیر len ذخیره می شود.

سپس در خط بعدی عبارت Content-Type: text/html\r\n در ادامه ی متن نوشته شده در خط اول قرار می گیرد که محل این قرار گیری توسط متغیر len مشخص می شود.

این مراحل برای نوشتن کدهای دیگر HTML ادامه پیدا می کند.

```
sprintf(str, "%u", time_s);
```

نوشتن متغیر time_s در متغیر رشته ای str

```
len = concatstr(response, len, str);
```

نوشتن محتویات متغیر str (متغیر time_s) در متغیر response در ادامه ی رشته های قبلی

```
len = concatstrf(response, len, "</FONT></P>\n");
len = concatstrf(response, len,
    "<P>Counter_1 = <FONT COLOR=#0000FF> ");
```

نوشتن ادامه ی کدهای HTML در متغیر response

```
sprintf(str, "%u", counter1);
```

نوشتن متغیر counter1 در متغیر رشته ای str

```
len = concatstr(response, len, str);
```

نوشتن محتویات متغیر str (متغیر counter1) در متغیر response در ادامه ی رشته های قبلی

```
len = concatstrf(response, len, "</FONT></P>\n");
len = concatstrf(response, len,
    "<P>Counter_2 = <FONT COLOR=#FF0000> ");
```

نوشتن ادامه ی کدهای HTML در متغیر response

```
sprintf(str, "%u", counter2);
```

نوشتن متغیر counter2 در متغیر رشته ای str

```
len = concatstr(response, len, str);
```

نوشتن محتویات متغیر str (متغیر counter2) در متغیر response در ادامه ی رشته های قبلی

```
len = concatstrf(response, len, "</FONT></P>\n");
len = concatstrf(response, len,
    "<P><A HREF=\".\">refresh</A><P></CENTER><HR>\n");
```

نوشتن ادامه ی کدهای HTML در متغیر response

```
time_cs = 0;
time_s = 0;
```

صفر کردن مقدار شمارنده های ثانیه و صدم ثانیه

```
return len;
```

فرستادن طول بافر پاسخ به عنوان خروجی تابع ethernet_tcp

اگرچه کدهای HTML مربوط به برنامه نویسی تحت وب هست و مبحثی جدا از این مقاله می باشد، اما در مورد کدهای HTML که در خطوط بالا و پروژه بعدی آورده شده اند توضیح می دهیم. پیشنهاد می شود خوانندگانی که علاقه به پیاده سازی این نوع کاربرد ها مانند طراحی وب سرور مرکزی دارند، به مراجع و منابع موجود در زمینه زبان های برنامه نویسی HTML و PHP و... مراجعه فرمایند.

سه خط اول مربوط به کدهای ابتدایی پیاده سازی پروتکل HTTP نسخه ۱ (HTTP/1.0) می باشد که به ترتیب در بافر response برای ارسال به کامپیوتر نوشته می شوند :

```
HTTP/1.0 200 OK\r\n
Content-Type: text/html\r\n
Pragma: no-cache\r\n\r\n
```

نکته : HTTP/1.1 نسخه جدیدتر HTTP/1.0 می باشد. در HTTP/1.0 برای هر درخواست از سرور (در این پروژه میکروکنترلر)، یک اتصال (connection) جدید تشکیل می شود. در HTTP/1.1 توسط یک اتصال درخواست های متعددی از سرور صورت می پذیرد، به همین منظور سرعت HTTP/1.1 از HTTP/1.0 بیشتر است. (اگر پروتکل TCP استفاده می شود، به دلیل تاخیر TCP نسبت به UDP، پیاده سازی HTTP/1.1 برای کاهش تاخیر و افزایش سرعت پیشنهاد می شود)

ادامه ی کدها مربوط به زبان HTML برای طراحی صفحه مورد نظر برای نمایش در کامپیوتر می باشد.

یکی از مهم ترین بخش های در کدهای HTML تگ ها هستند که داخل علامت های <> قرار می گیرند. همچنین همه ی تگ ها دارای شروع و پایان هستند که پایان آنها با </> مشخص می شود.

کدهای HTML همیشه با تگ <HTML> شروع می شوند.

برای مشخص کردن عنوان صفحه وب از تگ <TITLE> استفاده می شود و عنوان صفحه بین این دو تگ قرار می گیرد، به عنوان مثال :

```
<TITLE>Knowledgeplus.ir</TITLE>
```

آنچه که بین تگ <BODY> قرار می گیرد روی صفحه نمایش داده می شود.

تگ <CENTER> برای قرار دادن محتویات صفحه یا نوشته ها در وسط استفاده می شود.

تگ <P> برای نوشتن یک پاراگراف استفاده می شود، به عنوان مثال :

```
<P>Transferring Data via Ethernet</P>
```

\n برای رفتن به خط بعدی استفاده می شود.

از تگ <font> برای تعیین مشخصات نوشته مانند اندازه، رنگ و ... استفاده می شود. به عنوان مثال برای تعیین رنگ نوشته :

```
<font color=#00FF00> Knowledgeplus.ir</font>
```

در اینجا کد رنگ یعنی #00FF00 نوشته شده است اما میتوان به جای آن نام خود رنگ را نوشت مثلا :

```
<font color=red> Knowledgeplus.ir</font>
```

از تگ <a> برای ایجاد لینک به صفحات مختلف استفاده می شود، به عنوان مثال :

```
<a href="http://www. Knowledgeplus.ir ">Visit Knowledgeplus.ir.com!</a>
```

با نوشتن کد بالا، عبارت Visit Knowledgeplus.ir.com! بر روی صفحه مرورگر ظاهر می شود که با کلیک بر روی آن سایت Knowledgeplus.ir باز خواهد شد.

در کد های نوشته شده از این تگ برای ایجاد لینک refresh کردن صفحه مرورگر استفاده شده است.

۸-۳-۲-۳. فایل ethernet.h

نکته : کد های نوشته شده در این فایل با کدهای استفاده شده در پروژه سوم تشابه زیادی دارد، لذا توضیحات تکمیلی در مورد بخش های مختلف این فایل در پروژه سوم داده خواهد شد.

در ابتدای این فایل توابع مختلف که در ادامه نوشته شده اند تعریف می شوند. به عنوان مثال توابع high level functions توابع سطح بالا می باشند که توابع دیگر نوشته شده، توسط این توابع فراخوانی می شوند یا توابع enc28j60 functions که مربوط به عملیات مختلف تراشه ENC28J60 مانند نوشتن در بافر، خواندن بافر و... می شود. از دیگر توابعی که تعریف شده است می توان به توابع پیاده سازی پروتکل های مختلف مانند Ethernet ، IP ، UDP ، TCP و ARP نام برد. همچنین در ابتدای برنامه نام متغیرهای زبان C توسط دستور typedef و در بخش integer type definition تغییر داده شده است.

از دیگر بخش های این فایل که بخش زیادی از ابتدای کدها را به خود اختصاص داده است، بخش تعریف شناسه ها یا ماکرو ها توسط دستور #define است که برای سادگی و ایجاد قابلیت انعطاف استفاده شده است. یکی از فواید دیگر استفاده از ماکرو ها ایجاد خوانایی بیشتر برای کسانی است که کدها را می خوانند، به عنوان مثال کسی که برنامه را می خواند با دیدن عدد 0x11 شاید نتواند علت نوشتن این عدد را متوجه بشود و باید به دیتاشیت تراشه مراجعه کند ولی با نوشتن عبارت PHSTAT2 به جای این عدد به راحتی می توان متوجه استفاده از این رجیستر در این قسمت از برنامه شد.

شناسه های هر بخش برای دسترسی آسان تر به صورت جداگانه نوشته شده است. در بخش اول شناسه هایی که در پیاده سازی توابع پروتکل ها استفاده شده اند نوشته شده اند. در بخش بعدی با عنوان enc28j60 definitions شناسه هایی که در توابع تراشه ENC28J60 استفاده شده اند تعریف شده است.

در ادامه این فایل، ابتدا توابع تراشه تعریف شده اند و سپس توابع پیاده سازی پروتکل ها.

بخش enc28j60 functions :

از این توابع برای انجام عملیات پایه مانند خواندن بافر تراشه، نوشتن در بافر، ارسال بسته و ... استفاده می شود. مراحل انجام این فرایندها در دیتاشیت توضیح داده شده است و در فصل هفتم مراحل این عملیات را توضیح دادیم. در این بخش به عنوان مثال کدهای سه تابع enc28j60ReadBuffer ، enc28j60PacketReceive و enc28j60Init را برای آشنایی با نحوه ی نوشتن این توابع مطابق توضیحات دیتاشیت توضیح می دهیم.

تابع enc28j60ReadBuffer :

```
void enc28j60ReadBuffer(uint16_t len, uint8_t* data)
{
    CSACTIVE;
    // issue read command
    SPDR = ENC28J60_READ_BUF_MEM;
    waitspi();
    while(len)
    {
        len--;
        // read data
        SPDR = 0x00;
        waitspi();
        *data = SPDR;
        data++;
    }
    *data='\0';
    CSPASSIVE;
}
```

این تابع دو ورودی دارد، اولی متغیر len که طول داده هایی است که می خواهیم از بافر بخوانیم و دومی متغیر اشاره گر data که در آن داده خوانده شده به صورت یک رشته نوشته خواهد شد.

در خط اول این تابع عبارت CSACTIVE نوشته شده است که به صورت شناسه در ابتدای برنامه به این شکل تعریف شده است :

```
#define CSACTIVE      ENC28J60_SPI_PORT&=~(1<<ENC28J60_SPI_CS)
```

توسط این عبارت پایه ای از میکروکنترلر که به پایه ی CS تراشه متصل شده است (که توسط شناسه ENC28J60_SPI_CS تعیین شده است) ، برابر صفر می شود. بدین ترتیب ارتباط SPI فعال می شود.

در خط بعدی رجیستر SPDR (رجیستر داده SPI در میکروکنترلر AVR) برابر عبارت ENC28J60_READ_BUF_MEM قرار گرفته است، این عبارت در ابتدای خطوط کد در زیر عنوان SPI operation codes برابر مقدار 0x3A قرار گرفته است، این مقدار طبق توضیحات ارائه شده در مورد دیتاشیت تراشه، از جدول 4-1 دیتاشیت در ستون های Opcode و Argument دستور Read Buffer Memory برداشته شده است. (برای اطلاعات بیشتر به فصل هفتم مراجعه شود)

سپس از عبارت waitspi() استفاده شده است. این عبارت در ابتدای برنامه اینگونه تعریف شده است :

```
#define waitspi()     while(!(SPSR&(1<<SPIF)))
```

توسط این عبارت برنامه منتظر می ماند تا پرچم SPIF در رجیستر SPSR برابر صفر شود. (که به معنای ارسال داده های موجود در شیفت رجیستر SPDR می باشد)

سپس در حلقه ی while(len) در هر بار خواندن حلقه، داده های موجود در آدرسی که توسط اشاره گر ERDPT تعیین می شود، از پایه So تراشه خارج شده و در رجیستر SPDR میکروکنترلر ذخیره می شود و ما آن را در اشاره گر data ذخیره می کنیم، سپس یک واحد به اشاره گر data اضافه می شود تا داده بعدی در آدرس بعدی حافظه میکروکنترلر ذخیره شود. در هر بار خواندن حلقه، یک واحد از متغیر len کاهش پیدا می کند و در ابتدای حلقه مورد بررسی قرار می گیرد، در نتیجه حلقه تا زمانی که متغیر len صفر نشده است تکرار می شود و با صفر شدن این متغیر (به معنای پایان طول داده ای که قرار است بخوانیم) برنامه از حلقه ی while خارج می شود.

سپس در متغیر data کاراکتر null ('\0') قرار می گیرد تا به عنوان رشته در نظر گرفته شود.

در نهایت توسط عبارت CSPASSIVE معادل :

```
ENC28J60_SPI_PORT|=(1<<ENC28J60_SPI_CS)
```

پایه CS متصل به تراشه یک شده و عملیات خواندن بافر به اتمام می رسد.

تابع enc28j60PacketReceive :

توسط این تابع در صورت وجود بسته های جدید دریافتی در بافر گیرنده، آنها را ذخیره سازی می کنیم.

همان طور که در فصل های قبلی توضیح داده شد، در ابتدای هر بسته، یک بخش header وجود دارد که هنگام تجزیه و تحلیل بسته باید این موضوع مد نظر قرار گیرد.

این تابع دارای دو ورودی است : ورودی اول متغیر maxlen است که تعیین کننده ی حداکثر طول بسته های دریافتی است و ورودی دوم یعنی متغیر اشاره گر packet آدرس محلی است که باید بسته در آن ذخیره شود.

خروجی تابع نیز طول بسته دریافتی برحسب بایت می باشد و در صورت عدم وجود بسته جدید برای دریافت مقدار صفر بازگردانده می شود.

مطابق توضیحات دیتاشیت تراشه در صورت وجود بسته جدید، رجیستر EPKTCNT که مربوط به شمارنده بسته های موجود است نباید صفر باشد. این مسئله در ابتدای این تابع توسط خواندن این رجیستر توسط تابع enc28j60Read بررسی می شود و در صورت صفر بودن این رجیستر، مقدار صفر بازگردانده می شود.

در خطوط بعدی توسط تابع enc28j60Write متغیر اشاره گر NextPacketPtr توسط نوشتن مقدار رجیسترهای ERDPTL و ERDPTH به آدرس ابتدای بسته های دریافتی جدید منتقل می شود و سپس آدرس بسته های جدید دریافتی در این متغیرها ذخیره سازی می شوند.

مطابق توضیحات صفحه ۴۳ دیتاشیت، مقدار طول بسته های دریافتی توسط تابع enc28j60ReadOp خوانده شده و در متغیر len ذخیره می شود. سپس با کم کردن ۴ واحد از این متغیر، کدهای CRC را حذف می کنیم.

با خواندن بخش بعدی بافر، بیت های مربوط به وضعیت بسته های دریافتی را در متغیر rxstat ذخیره سازی می کنیم.

در بخش بعدی، طول بسته دریافتی را با حداکثر طول تنظیم شده مقایسه کرده و در صورت لازم آن را تغییر می دهیم.

در قسمت بعدی مطابق توضیحات دیتاشیت در صفحه ۴۴ و جدول ۳-۷ ، کدهای CRC و خطاهای احتمالی را بررسی میکنیم و در صورت عدم وجود خطا، بسته جدید دریافتی را توسط تابع enc28j60ReadBuffer در متغیر ذخیره سازی میکنیم.

در بخش بعدی، متغیر اشاره گر مربوط به خواندن بسته ها را به آدرس بسته بعدی منتقل می کنیم.

در نهایت با اتمام عملیات خواندن بسته جدید، توسط تابع enc28j60WriteOp ، مقدار بیت PKTDEC موجود در رجیستر ECON2 را یک می کنیم، با این کار طبق توضیحات دیتاشیت مقدار رجیستر شمارنده بسته های دریافتی یعنی EPKTCNT یک واحد کاهش پیدا می کند.

در انتها مقدار متغیر len بازگردانده می شود.

تابع enc28j60Init :

یکی از توابع مهم بخش enc28j60 functions تابع enc28j60Init می باشد. از این تابع برای انجام

تنظیمات اولیه تراشه ENC28J60 استفاده می شود.

مشابه توضیحات تابع enc28j60ReadBuffer ، این تابع برای فرایند پیکربندی اولیه تراشه نوشته شده است و می توان کدهای آن را طبق comment های ارائه شده در برنامه و توضیحات دیتاشیت تجزیه و تحلیل کرد.

این تابع توسط تابع ethernet_init و توسط مقدار ورودی آدرس MAC فراخوانی می شود.

در بخش بعدی این فایل، کدهای eth/ip/arp/udp/tcp functions implementation توابع مرتبط با پیاده سازی این پروتکل ها آورده شده است.

در ابتدا، توابع پایه پیاده سازی پروتکل های مختلف نوشته شده است و سپس در انتها تابع ethernet_process (که در حلقه ی while(1) برنامه اصلی استفاده شده است) از این توابع پایه برای پیاده سازی ارتباط استفاده می کند.

همچنین دو توابع concatstrf و concatstr که در بخش های قبلی توضیح داده شد و برای اتصال رشته به بافر استفاده می شوند با توجه به کاربردشان در این بخش تعریف شده اند.

تابع ethernet_init :

این تابع باید قبل از همه ی توابع دیگر فراخوانی شود. در این تابع مقادیر آدرس MAC ، آدرس IP ، شماره پورت TCP و شماره پورت UDP در ورودی دریافت شده و در متغیرهای مناسب برای استفاده های بعدی ریخته می شوند. در انتهای این تابع آدرس MAC که توسط کاربر به این تابع ارسال شده است، به تابع enc28j60Init() برای تنظیم این آدرس در تراشه ENC28J60 ارسال می شود.

این تابع در انتهای بخش مربوط به پیاده سازی توابع پروتکل ها و قبل از تابع ethernet_process قرار گرفته است.

تابع ethernet_process :

این تابع در برنامه اصلی در حلقه ی دائمی برنامه ((while(1)) قرار گرفته است و وظیفه پیاده سازی بخش های مختلف ارتباط با شبکه را به عهده دارد.

پس از تعریف متغیرها در ابتدای تابع، بسته های جدید توسط تابع enc28j60PacketReceive دریافت و طبق توضیحاتی که در مورد این تابع ارائه شد، خروجی این تابع اندازه بسته دریافت شده را در

متغیر plen ذخیره می کند. سپس اگر این متغیر برابر با صفر نبود، به این معنا است که بسته جدید دریافت شده است و باید مورد بررسی قرار بگیرد.

در دستور if ، در بخش اول توسط تابع eth_type_is_arp_and_my_ip بسته دریافت شده بررسی می شود و اگر این بسته مربوط به پروتکل ARP باشد و IP آن برابر IP تنظیم شده در برنامه باشد، توسط تابع make_arp_answer_from_request به آن پاسخ داده می شود.

در صورتی که بسته از نوع ARP نباشد، توسط تابع eth_type_is_ip_and_my_ip بررسی می شود و اگر بسته دریافت شده مربوط به پروتکل IP باشد و IP آن برابر IP تنظیم شده در برنامه باشد، باید مورد پردازش قرار بگیرد، برنامه وارد حلقه شده و پروتکل های TCP و UDP توسط توابع تعریف شده در بخش های قبلی در این بخش پیاده سازی می شوند.

۸-۳-۲-۴. تست پروژه

پس از راه اندازی و آزمایش مدار، می توانید دستگاه را با استفاده از یک کابل شبکه به کامپیوتر خود متصل نمایید و ارتباط برقرار کنید. برای آزمایش دستگاه از یک نرم افزار مرورگر وب مثل Internet Explorer یا Mozilla Firefox استفاده کنید. ابتدا آدرس IP دستگاه (مثلا http://192.168.1.10) را در بخش آدرس مرورگر خود وارد نمایید و دکمه Enter را فشار دهید، البته قبل از این کار باید تنظیمات ارتباطی مرورگر را در مُد Automatic قرار دهید و آدرس IP کامپیوتر را به صورت دستی در محدوده دید دستگاه (مثلا 192.168.1.1) تنظیم کنید.

به این ترتیب، با انجام این کار، یک صفحه ساده HTML ظاهر می شود. این صفحه، حاوی اطلاعات ارسال شده از سوی دستگاه است که شامل مقدار شمارش شده در شمارنده و دفعات فشار داده شدن دکمه های B1 و B2 می باشد.

نکته: آدرس IP دستگاه همان طور که در بخش های قبلی نیز اشاره شد، با توجه به وضعیت دیپ سوییچ مشخص می شود و در بازه 192.168.1.10 تا 192.168.1.25 قرار می گیرد.

۸-۳-۳. پروژه سوم

این پروژه مشابه پروژه قبلی می باشد با این تفاوت که دارای امکانات بیشتری می باشد و کدها نیز به صورت پیشرفته تر و به صورت کتابخانه های مجزا نوشته شده اند. کدهای این پروژه مطابق سخت افزار پروژه شماره ۳ نوشته شده است اما میتوان با تغیر ثوابت و ماکرو های برنامه از آن در آرایش های سخت افزاری دیگر نیز استفاده نمود.

در پوشه اصلی پروژه سه فایل main.c برای برنامه اصلی، فایل global.c برای تعریف متغیرهای عمومی و برخی از ماکرو ها و فایل WINAVR_CAMP.h نیز که برخی دیگر از ثوابت در آن تعریف شده است قرار گرفته است.

کتابخانه های این پروژه برای کامپایلر CodevisionAVR در دو پوشه Driver و net قرار گرفته اند. در پوشه درایور فایل های مربوط به پیاده سازی تراشه ENC28J60 ، طراحی منوها، پیاده سازی پروتکل اترنت و راه اندازی بخش ADC میکروکنترلر قرار گرفته است.

در پوشه net فایل های مربوط به پیاده سازی پروتکل های لایه های بالاتر قرار گرفته است.

در بخش های بعدی به توضیح در مورد هریک از این کتابخانه ها و توابع موجود در آنها می پردازیم. برخی از کدهای نوشته شده مانند راه اندازی ADC و طراحی منوها مربوط به مباحث میکروکنترلر ها می باشد که مطالعه و بررسی آنها به خواننده واگذار می شود. همچنین بخش های مختلفی از کدهای نوشته شده با پروژه قبلی مشابهت دارد که از توضیح مجدد آنها نیز صرف نظر می کنیم.

پیش از شروع توضیح در مورد کدهای این پروژه، به چند نکته در مورد این کدها اشاره می کنیم.

نکته ۱ : همان طور که در ابتدای فایل های کتابخانه توضیح داده شده است، سورس های استفاده شده در این پروژه متعلق به Free Software Foundation یا بنیاد نرم افزار آزاد (موسسه آموزشی در شهر بوستون آمریکا) می باشد که به صورت رایگان و برای اهداف آموزشی در اختیار عموم قرار گرفته است که مطابق نیازهای پروژه ما تغییراتی در این کدها ایجاد کرده و از برخی از توابع این کتابخانه ها که در ادامه توضیح داده خواهد شد استفاده کرده ایم. برای دسترسی به کتابخانه ها و سورس های بیشتر می توانید به وب سایت این موسسه مراجعه فرمایید.

نکته ۲ : در مدل های شبکه مانند OSI یا مدل TCP/IP همواره باید این نکته را در نظر بگیریم که

لایه های پایین تر به لایه های بالاتر سرویس می دهند.

این مسئله در پیاده سازی لایه های مختلف در کتابخانه های ارائه شده قابل مشاهده است. به عنوان مثال در کتابخانه http برای پیاده سازی این پروتکل از توابع پروتکل TCP که در لایه پایین تر (لایه انتقال) می باشد استفاده شده است، در پیاده سازی پروتکل TCP از توابع پروتکل IP که در لایه پایین تر یعنی لایه شبکه قرار دارد استفاده شده است یا در پیاده سازی پروتکل ARP از توابع پروتکل Ethernet (به عنوان مثال تابع eth_generate_header) استفاده شده است.

نکته ۳ : همان طور که قبلا نیز اشاره شد، مطالعه و تجزیه و تحلیل این پروژه از پروژه قبلی پیچیده تر است و مستلزم تخصیص وقت بیشتر برای تحلیل آن می باشد. یکی از علت های این مسئله وابستگی کدهای نوشته شده در کتابخانه و فایل های مختلف است. به عنوان مثال در فایل main و در تابع server_process ، آرایه rxtx_buffer تعریف شده است.

ظرفیت اندازه این آرایه توسط ثابت MAX_RXTX_BUFFER تعیین شده است که این ثابت در فایل global تعیین شده است.

از آرایه rxtx_buffer در کتابخانه های مختلف و بخش های مختلف برنامه برای نوشتن و یا خواندن این حافظه استفاده شده است. به عنوان مثال در کتابخانه Ethernet تابع eth_generate_header که برای تولید فایل های سرآمد لایه اترنت نوشته است، در آدرس ETH_DST_MAC_P آرایه rxtx_buffer آدرس MAC مقصد و در آدرس ETH_SRC_MAC_P آدرس MAC میکروکنترلر نوشته می شود.

مقادیر ETH_DST_MAC_P و ETH_SRC_MAC_P در ابتدای فایل ethernet.h در پوشه Driver به عنوان ثابت به ترتیب با مقدار صفر و شش تعریف شده اند.

همان طور که مشاهده می شود مطالعه این پروژه و تجزیه و تحلیل آن به کمی دقت و وقت بیشتر نیاز دارد.

۸-۳-۳-۱. فایل main.c

در ابتدای برنامه کتابخانه های مورد نیاز توسط دستور #include به برنامه اضافه شده اند، سپس دو تابع server_process و client_process تعریف شده اند که در ادامه در مورد آنها توضیح می دهیم.

در ابتدا، آدرس MAC برد در avr_mac.byte تعیین شده است.

سپس آدرس IP میکروکنترلر و سرور از تراشه EEPROM خوانده شده و در avr_ip.byte مربوط به آدرس IP میکروکنترلر AVR و server_ip.byte مربوط به آدرس IP سرور ریخته می شود. (ذخیره سازی آدرس های IP در حافظه EEPROM در فایل global صورت گرفته است)

در بخش بعدی تنظیمات فعال سازی مقاومت های pull-up انجام شده است.

در بخش setup clock for timer1 تنظیمات رجیسترهای تایمر شماره ۱ میکروکنترلر انجام شده است.

سه تابع adc_init ، lcd_init و menu_init به ترتیب وظیفه تنظیم بخش ADC میکروکنترلر، تنظیم LCD کاراکتری با ۱۶ ستون و تنظیم اولیه منوهای برنامه که بر روی LCD کاراکتری نشان داده می شود را به عهده دارند.

در بخش بعدی تنظیمات مربوط به پایه های LED انجام شده است.

تابع enc28j60_init() برای تنظیمات و پیکربندی اولیه تراشه ENC28J60 فراخوانی می شود.

در حلقه while(1) یا حلقه دائمی برنامه :

برای زمان بندی سیستم، از سر ریز تایمر ۱ در هر ۴ میلی ثانیه استفاده می شود.

توسط تابع adc_read_temp مقدار دمای محیط توسط سنسور متصل شده به ADC میکروکنترلر اندازه گیری می شود.

توسط تابع server_process به درخواست های پروتکل های ARP ، ICMP و HTTP پاسخ داده می شود.

توسط تابع client_process (که به صورت پیشفرض غیر فعال شده است)، دمای اندازه گیری شده توسط واحد ADC که توسط تابع adc_read_temp خوانده شده است، از طریق پروتکل HTTP به سمت وب سرور ارسال می شود.

تابع menu_process وظیفه تنظیم منوهای LCD مطابق دستورات کاربر و همچنین وظیفه تنظیم

آدرس IP و شمارش تایمر را به عهده دارد.

تابع standby_display وظیفه نمایش دما، آدرس IP میکروکنترلر و سرور و شمارش تایمر را به عهده دارند.

تابع client_process :

همان طور که اشاره شد، این تابع برای ارسال دمای اندازه گیری شده به سمت وب سرور مورد استفاده قرار می گیرد و به صورت پیش فرض در برنامه استفاده نشده است و غیر فعال است.

برای استفاده از این تابع، باید وب سرور و کدهای آن را قبل از استفاده از این تابع پیاده سازی کنید. (به عنوان مثال می توانید از وب سرور Apache و کدهای PHP استفاده کنید)

برای توضیحات بیشتر در مورد پیاده سازی Apache و استفاده از کدهای PHP می توانید به آدرس http://www.avrportal.com یا مراجع موجود در این زمینه مراجعه کنید.

تابع Server_process :

همان طور که اشاره شد از این تابع برای پیاده سازی پروتکل های مورد استفاده در برنامه استفاده می شود.

در بخش اول این تابع، متغیر client_mac از نوع ساختار MAC_ADDR (تعریف شده در فایل struct به صورت شش بایتی)، متغیر client_ip از نوع ساختار IP_ADDR (تعریف شده در فایل struct به صورت چهار بایتی)، آرایه ۸ بایتی rxtx_buffer با ظرفیت MAX_RXTX_BUFFER (تعریف شده در فایل global معادل ۱۵۱۸ بایت) به عنوان حافظه بافر فرستنده و گیرنده و متغیر plen با اندازه دو بایت تعریف شده اند.

در بخشی get new packet بسته داده جدید را دریافت و در متغیر plen ذخیره می کنیم.

توسط تابع memcpy آدرس MAC موجود در بافر تراشه در متغیر client_mac برای استفاده های بعدی ذخیره می شود.

در قسمت بعدی توسط تابع arp_packet_is_arp بسته های مربوط به پروتکل ARP بررسی می شوند و در صورتی که آدرس IP بسته ها با آدرس IP تنظیم شده در میکروکنترلر برابر باشد، توسط تابع arp_send_reply به آنها پاسخ داده می شود.

توسط تابع memcpy آدرس IP کلاینت دریافت و برای ذخیره سازی مورد بررسی قرار می گیرد.

در بخش بعدی توسط تابع icmp_send_reply داده های دریافتی مورد بررسی قرار می گیرند و اگر بسته ها مربوط به پروتکل ICMP باشند، به آنها پاسخ داده خواهد شد.

در بخش بعدی بسته های دریافتی برای پروتکل UDP توسط تابع udp_receive مورد بررسی قرار می گیرند.

در انتهای این تابع توسط پروتکل TCP و پورت شماره ۸۰ ، توسط تابع http_webserver_process تعریف شده در فایل http.c ، پروتکل HTTP برای ایجاد صفحه وب پیاده سازی می شود.

۸-۳-۳-۲. فایل global.c

MAC_ADDR به شکل زیر به عنوان یک ساختار (structure) در فایل struct در پوشه Driver تحت یک آرایه ۶ بایتی تعریف شده است :

```
// mac address structure
typedef struct _MAC_ADDR
{
  uint8_t byte[6];
}MAC_ADDR;
```

به شکل مشابه IP_ADDR به شکل زیر به عنوان یک ساختار در فایل struct تحت یک آرایه ۴ بایتی تعریف شده است :

```
// IP address structure
typedef struct _IP_ADDR
{
      uint8_t byte[4];
}IP_ADDR;
```

در ابتدای این فایل، متغیرهایی برای ذخیره سازی آدرس IP و آدرس MAC میکروکنترلر AVR

(avr_ip و avr_mac) و متغیرهایی برای ذخیره سازی آدرس IP و آدرس MAC گیرنده (server_ip و server_mac) تعریف شده اند.

سپس آدرس IP میکروکنترلر و سمت گیرنده مقداردهی شده اند و در حافظه EEPROM میکروکنترلر ذخیره سازی شده اند.

همان طور که در فصل هفتم در توضیحات دیتاشیت تراشه ENC28J60 توضیح داده شد، باید توسط میکروکنترلر حداکثر حافظه بافر تعیین شود. توسط ثابت MAX_RXTX_BUFFER حداکثر ظرفیت حافظه بافر ۱۵۱۸ بایت تعریف شده است.

در انتهای این فایل نیز پایه های مربوط به LED1 و LDE2 مشخص شده اند.

۸-۳-۳-۳. فایل WinAVR_CAMP.h

در این فایل برخی ثوابت مورد استفاده در برنامه تعریف شده است. به عنوان مثال نوشتن عبارت :

```
delay_ms(ms)
```

که تابع ایجاد تاخیر برحسب میلی ثانیه در کتابخانه delay می باشد معادل عبارت :

```
_delay_ms(ms)
```

می باشد یا عبارت :

```
LOW(x)
```

معادل عبارت :

```
(x&0xFF)
```

می باشد، به عنوان مثال اگر data یک متغیر باشد عبارت :

```
LOW(data)
```

معادل :

```
(data&0xFF)
```

می باشد.

در بخش بعدی در مورد دو فایل Ethernet و struct در پوشه Driver توضیح می دهیم.

۸-۳-۳-۴. فایل Ethernet.c

تابع eth_generate_header :

توسط این تابع سرآمد یا header مربوط به پروتکل اترنت (شامل آدرس MAC مبدا و مقصد) تولید می شود.

ورودی های این تابع عبارت اند از آرایه rxtx_buffer به عنوان حافظه بافر فرستنده و گیرنده، متغیر دو بایتی type و اشاره گر dest_mac .

توسط حلقه for نوشته شده در تابع که تعداد تکرار آن به تعداد بایت های متغیر MAC_ADDR (که در فایل struct به صورت پیشفرض ۶ بایت تنظیم شده است) بستگی دارد، آدرس MAC مقصد در آدرس ETH_DST_MAC_P و آدرس MAC میکروکنترلر در آدرس ETH_SRC_MAC_P آرایه rxtx_buffer نوشته می شوند.

مقدار ETH_DST_MAC_P و ETH_SRC_MAC_P در ابتدای فایل Ethernet.h در پوشه Driver به عنوان ثابت به ترتیب با مقدار صفر و شش تعریف شده اند.

در انتهای تابع، مقدار متغیر type توسط کد HIGH(type) در آدرس ETH_TYPE_H_P (معادل عدد ۱۲) آرایه rxtx_buffer نوشته شده و سپس توسط کد LOW(type) هشت واحد به راست شیفت پیدا کرده و در آدرس ETH_TYPE_L_P (معادل عدد ۱۳) آرایه rxtx_buffer ذخیره سازی می شود.

از این تابع در پیاده سازی پروتکل های ARP ، ICMP ، TCP و UDP استفاده شده است.

تابع software_checksum :

توسط این تابع الگوریتم خطایابی checksum بر روی ۲۰ بایت اول header پروتکل IPv4 انجام می شود تا از صحت اطلاعات دریافتی (توسط مقایسه با checksum انجام شده در فرستنده) اطمینان حاصل شود.

در دو لینک زیر اطلاعات بیشتری در مورد محاسبه این الگوریتم آروده شده است :

http://www.netfor2.com/checksum.html

http://www.avrportal.com/?page=avrnet-checksum

همچنین در لینک زیر مربوط به مقالات RFC (انجمن انتشار استانداردهای اینترنت)، این الگوریتم به زبان C پیاده سازی شده است.

http://www.faqs.org/rfcs/rfc1071.html

از این تابع در پیاده سازی پروتکل های ICMP ، IP ، TCP و UDP استفاده شده است.

۸-۳-۳-۵. فایل struct.h

در بخش اول این فایل، نام نوع متغیرهای تغییر پیدا کرده است :

```
typedef unsigned char     uint8_t;     // 8-bit
typedef unsigned int   uint16_t;       // 16-bit
typedef unsigned long     uint32_t;    // 32-bit
```

به عنوان مثال به جای unsigned char از uint8_t استفاده می شود.

در دیگر بخش های این فایل، ساختارهای مختلف مورد استفاده در بخش های دیگر این پروژه تعریف شده است. به عنوان مثال MAC_ADDR برای آدرس MAC و IP_ADDR برای آدرس IP در فایل global.c استفاده شده است.

در بخش های بعدی ساختار بسته های های پروتکل های مختلف و سرآیند آنها با توجه به توضیحات ارائه شده در فصل پنجم به عنوان ساختار تعریف شده اند.

در بخش های بعدی در مورد کتابخانه های پوشه Network توضیح می دهیم.

نکته : ثوابت استفاده شده در کتابخانه هایی که در ادامه توضیحات داده می شود در فایل سرآیند آنها تعریف شده است، به عنوان مثال در فایل arp.c از ثابت ARP_HARDWARE_TYPE_H_P استفاده شده است که این ثابت در فایل arp.h تعریف شده است.

۸-۳-۳-۶. کتابخانه ARP

تابع arp_generate_packet :

از این تابع برای تولید بسته های ARP استفاده می شود. ورودی های تابع عبارت اند از حافظه بافر، آدرس MAC مقصد و آدرس IP مقصد. در مورد ساختار بسته های ARP در فصل پنجم توضیح داده شد. همچنین در بالای این فایل نیز به شکل خلاصه ساختار بسته های ARP رسم شده است.

در ابتدای این تابع، نوع سخت افزار برای پروتکل اترنت عدد ۱ قرار می گیرد.

در بخش بعدی نوع پروتکل تعیین می شود که مطابق توضیحات فصل پنجم برای IPv4 برابر 0x0800 قرار می گیرد.

در خطوط بعدی به ترتیب طول آدرس MAC برابر ۶ بایت و طول آدرس IP برابر ۴ بایت تنظیم می شوند.

در حلقه ی for اول آدرس MAC فرستنده و گیرنده و در حلقه for دوم آدرس IP فرستنده و گیرنده تنظیم می شوند.

تابع arp_send_request :

همان طور که در فصل پنجم اشاره شد، پروتکل ARP بر مبنای درخواست (request) و پاسخ (reply) بنا شده است.

از این تابع برای ارسال درخواست ARP (مورد نیاز در فرایند Who-is برای یافتن آدرس MAC مقصد) استفاده می شود. تابع arp_send_reply نیز که در ادامه توضیح می دهیم برای پاسخ دادن نوشته شده است.

از این تابع در تابع arp_who_is برای یافتن آدرس MAC مقصد استفاده می شود.

ورودی های این تابع حافظه بافر و آدرس IP گیرنده می باشد.

در این تابع برای ارسال درخواست ARP ، ابتدا برای تعیین نوع پیام از نوع Broadcast ، آدرس MAC مقصد (dest_mac.byte) را برابر FF:FF:FF:FF:FF:FF تنظیم می کنیم.

سپس با استفاده از تابع eth_generate_header (در فایل Ethernet)، بخش هدر اترنت (مورد نیاز برای بسته های ارسالی در شبکه اترنت) برای بسته ARP را می سازیم. سپس توسط تابع arp_generate_packet که توضیح داده شد، بسته ی درخواست ARP را تشکیل می دهیم و در نهایت توسط تابع enc28j60_packet_send (در فایل enc28j60)، بسته ARP تشکیل شده را در شبکه ارسال می کنیم.

نکته : مطابق توضیحات فصل پنجم، بسته های درخواست دارای opcode برابر عدد ۱ هستند که در بخش set arp opcode is request تنظیم می شود.

تابع arp_packet_is_arp :

از این تابع برای تشخیص بسته های از نوع ARP و همچنین تعیین این مسئله که آیا IP این بسته برابر IP میکروکنترلر می باشد یا خیر استفاده می شود.

این تابع در تابع arp_packet_is_arp استفاده می شود.

ورودی های این تابع حافظه بافر و opcode می باشند. (برای بسته های درخواست برابر عدد ۱ و برای بسته های پاسخ برابر عدد ۲ می باشد)

در صورتی که بسته دریافتی از نوع ARP باشد و IP آن برابر IP میکروکنترلر باشد، تابع مقدار ۱ را بر میگرداند و در غیر اینصورت مقدار صفر را برمیگرداند.

تابع arp_send_reply :

این تابع مشابه تابع arp_send_request برای ایجاد پاسخ به بسته های ARP مورد استفاده قرار می گیرد. ورودی های این تابع حافظه بافر و آدرس MAC مقصد است که میخواهیم پاسخ ARP را برای آن ارسال کنیم.

نکته : مطابق توضیحات فصول قبلی، برای بسته های پاسخ opcode برابر عدد ۲ می باشد.

تابع arp_who_is :

همان طور که در فصول پنجم توضیح داده شد، اگر قصد ارسال بسته ای را به گره ای از شبکه داشته باشیم و آدرس MAC آن را ندانیم، طبق قوائد پروتکل ARP باید فرایند Who-is اجرا شود. این تابع برای این منظور نوشته شده است. لذا اگر آدرس MAC گره ای را نمی دانیم، قبل از ارسال بسته به آن، باید این تابع را اجرا کنیم.

ورودی های این تابع عبارت اند از حافظه بافر، متغیر برای ذخیره سازی آدرس MAC مقصد و آدرس IP مقصدی که میخواهیم آدرس MAC آن را به دست آوریم.

ابتدا توسط تابع arp_send_request بسته ی درخواست ARP از نوع Broadcast به همراه آدرس IP گره مورد نظر ارسال می کنیم.

سپس در حلقه for با زمان بندی مشخصی، بسته پاسخ را توسط تابع enc28j60_packet_receive دریافت کرده و سپس تابع arp_packet_is_arp نوع بسته و IP آن را بررسی کرده و در نهایت توسط تابع memcpy آدرس MAC دریافت شده را در متغیر dest_mac ذخیره سازی می کنیم.

در صورتی که توانستیم آدرس MAC گره مورد نظر را پیدا کنیم، تابع مقدار ۱ را برمیگرداند و در غیر اینصورت مقدار صفر برگشت داده می شود.

۸-۳-۳-۷. کتابخانه IP

تابع ip_generate_header :

از این تابع برای تولید بخش سرآیند بسته های IP استفاده می شود.

ورودی های این تابع عبارت اند از حافظه بافر، بخش های ظرفیت بسته های IP، پروتکل مورد استفاده در لایه انتقال و آدرس IP گیرنده که مربوط به سرآیند پروتکل IP می شوند. کدهای نوشته شده به ترتیب بخش های مختلف سرآیند بسته های IP را تکمیل می کنند.

نکته : برای محاسبات checksum باید ابتدا بیت های آن را صفر کنیم و سپس محاسبات را انجام دهیم.

تابع ip_packet_is_ip :

این تابع که تنها ورودی آن اشاره گر حافظه بافر می باشد، بسته های دریافتی را بررسی کرده و در صورتی که بسته از نوع IP باشد و آدرس IP گیرنده آن (در بخش سرآیند IP بسته) برابر IP تنظیم شده در میکروکنترلر باشد، مقدار یک به عنوان خروجی تابع بازگردانده می شود، در غیر اینصورت مقدار صفر بازگردانده می شود.

۸-۳-۳-۸. کتابخانه ICMP

تابع icmp_generate_packet :

از این تابع برای تولید بسته های پروتکل ICMP استفاده می شود. تنها ورودی این تابع حافظه بافر می باشد و خروجی ندارد.

از این تابع در تابع icmp_send_request برای ارسال درخواست های پروتکل ICMP در شبکه استفاده می شود.

تابع icmp_send_request :

از این تابع برای ارسال درخواست پروتکل ICMP در شبکه استفاده می شود. ورودی های این تابع حافظه بافر، آدرس MAC و آدرس IP گیرنده هستند.

توسط تابع eth_generate_header (در فایل Ethernet)، هدر پروتکل اترنت ساخته می شود و توسط تابع ip_generate_header (در فایل IP) هدر پروتکل IP ساخته می شود. در بخش بعدی قسمت های مختلف هدر تنظیم می شود. از جمله این بخش ها برای پیام های درخواست، تنظیم مقدار type برابر عدد ۸ (توسط ثابت ICMP_TYPE_ECHOREQUEST_V که درفایل ICMP.h تعریف شده است) و تنظیم بخش code برابر مقدار صفر می باشد.

همچنین مقدار identifier و sequence number توسط دو متغیر icmp_id و icmp_seq که در ابتدای برنامه تعریف شده اند برابر مقدار یک تنظیم شده اند.

توسط تابع icmp_generate_packet بسته های پروتکل ICMP ساخته و همچنین عملیات checksum صورت می پذیرد و در نهایت توسط تابع enc28j60_packet_send بسته درخواست ICMP در شبکه ارسال می شود.

تابع icmp_send_reply :

توسط این تابع به درخواست های ICMP که توسط تابع icmp_send_request در شبکه ایجاد می شوند پاسخ داده می شود، به همین دلیل این دو تابع شباهت دارند.

ورودی های این تابع مانند تابع icmp_send_request می باشد. در ابتدای تابع نوع بسته بررسی می شود که از نوع ICMP است یا خیر، سپس نوع درخواست بسته ICMP برای تشخیص این که از نوع echo است یا خیر بررسی می شود. در صورت مثبت بودن این دو مسئله بالا (بسته از نوع ICMP و درخواست از نوع echo)، مانند تابع icmp_send_request مراحل مختلف برای ساخت بسته ی پروتکل ICMP انجام می شود (با این تفاوت که مقدار type برای پیام پاسخ باید برابر صفر تنظیم شود که توسط ثابت ICMP_TYPE_ECHOREPLY_V این مقدار تنظیم می شود) و در نهایت توسط تابع enc28j60_packet_send بسته پاسخ ICMP در شبکه ارسال می شود.

همچنین مقدار برگشتی تابع در صورت مثبت بودن پاسخ دو سوال اول (که توسط دستور if بررسی می شوند) یک و در غیر اینصورت صفر خواهد بود.

تابع icmp_ping :

از این تابع برای انجام فرایند ping در شبکه استفاده می شود.

در عملیات ping ، یک پیام درخواست فرستاده می شود و مدت زمانی منتظر پاسخ می مانیم (در این تابع به اندازه 100ms)، در صورتی که پیام پاسخ را دریافت کردیم، مقدار یک به عنوان خروجی تابع بازگردانده می شود و در غیر اینصورت مقدار صفر بازگشت داده می شود.

۸-۳-۳-۹. کتابخانه TCP

تابع tcp_get_dlength :

از این تابع برای محاسبه ظرفیت بخش داده بسته های TCP دریافت شده استفاده می شود. ورودی این تابع حافظه بافر می باشد و مقدار برگشتی آن اندازه بخش داده بسته های TCP می باشد.

تابع tcp_get_hlength :

از این تابع برای محاسبه ظرفیت بخش سرآیند بسته های TCP دریافت شده استفاده می شود. ورودی این تابع حافظه بافر می باشد و مقدار برگشتی آن اندازه بخش سرآیند بسته های TCP می باشد.

تابع tcp_puts_data_p :

از این تابع برای قرار دادن داده های حافظه flash درون حافظه فرستنده استفاده می شود. ورودی های این تابع به ترتیب عبارت اند از حافظه بافر، داده موجود در حافظه flash و آفست آدرس مورد نیاز . مقدار برگشتی تابع نیز مقدار آفست تغییر یافته می باشد.

تابع tcp_puts_data :

مشابه تابع tcp_puts_data_p برای قرار دادن داده های حافظه SRAM در بافر فرستنده

تابع tcp_send_packet :

از این تابع برای ارسال بسته های tcp در شبکه استفاده می شود.

ورودی های تابع شامل موارد مختلفی مورد نیاز برای تنظیم سرآیند پروتکل TCP می باشد مانند پورت

فرستنده، پورت گیرنده، پرچم ها، آدرس MAC و آدرس IP مقصد و موارد مورد نیاز دیگر .

در ابتدا توسط تابع eth_generate_header (در فایل Ethernet) سرآیند پروتکل اترنت مورد نیاز برای بسته های شبکه ساخته می شود.

سپس تنظیمات مربوط به شماره ترتیب (sequence number) صورت می گیرد. حداکثر ظرفیت بسته های TCP (segment) و ظرفیت سرآیند TCP (برحسب کلمات ۳۲ بیتی) تنظیم می شوند.

در بخش بعدی توسط تابع ip_generate_header (فایل ip) سرآیند پروتکل ip برای بسته های داده ساخته می شود.

در بخش بعدی تنظیمات مربوط به پرچم ها و بخش های دیگر سرآیند TCP مانند آدرس پورت فرستنده و آدرس پورت گیرنده و دیگر بخش ها صورت می پذیرد.

در انتها توسط تابع software_checksum (در فایل Ethernet) عملیات checksum صورت پذیرفته و توسط تابع enc28j60_packet_send (در فایل ENC28J60) بسته TCP تشکیل شده برای ارسال به شبکه به تراشه ENC28J60 تحویل داده می شود.

۸-۳-۳-۱۰. کتابخانه UDP

تابع udp_generate_header :

از این تابع برای تولید بخش سرآیند بسته های UDP استفاده می شود.

ورودی های این تابع عبارت اند از حافظه بافر، شماره پورت مقصد و اندازه بسته های UDP . تابع به عنوان خروجی مقداری را برنمی گرداند.

کدهای نوشته شده به ترتیب بخش های مختلف سرآیند بسته های UDP را تکمیل می کنند.

از این تابع در تابع udp_receive استفاده می شود.

تابع udp_puts_data و udp_puts_data_p :

این دو تابع مشابه دو تابع tcp_puts_data و tcp_puts_data_p که در فایل TCP برای پیاده سازی پروتکل TCP نوشته شدند عمل می کنند و برای نوشتن بخش داده بسته های UDP مورد استفاده قرار می گیرند. تابع udp_puts_data_p برای نوشتن داده های موجود در حافظه flash میکروکنترلر استفاده می شود و تابع udp_puts_data برای نوشتن داده موجود در حافظه SRAM میکروکنترلر مورد

استفاده قرار می گیرد.

مقدار ورودی این توابع عبارت است از حافظه بافر، داده ای که میخواهیم در بسته UDP بنویسیم و مقدار آفست آدرس حافظه بافر برای بخش داده بسته های UDP و خروجی این توابع مقدار جدید آفست می باشد.

از این دو تابع در تابع udp_receive استفاده می شود.

تابع udp_receive :

از این تابع برای دریافت بسته های UDP و انجام پردازش بر روی آنها استفاده می شود.

ورودی های تابع عبارت اند از حافظه بافر، آدرس MAC مقصد و آدرس IP مقصد.

در صورتی که بسته دریافت شده از نوع UDP باشد و آدرس پورت آن با آدرس پورت میکروکنترلر یکسان باشد، مقدار یک برگردانده می شود و در غیر اینصورت مقدار صفر برگردانده می شود.

۸-۳-۳-۱۱. کتابخانه HTTP

همان طور که در توضیحات تابع server_process توضیح داده شد، این فایل حاوی توابع مورد نیاز برای پیاده سازی پروتکل HTTP و طراحی صفحه وب توسط کدهای به زبان HTML می باشد.

نکته : در مورد کدهای HTTP و HTML توضیحاتی در پروژه قبلی ارائه شد. در این بخش نیز برخی توضیحات دیگر در مورد کدهای استفاده شده در این پروژه ارائه خواهد شد.

در ابتدای این فایل چند ثابت در حافظه flash تعریف شده است که از آنها در تابع http_home استفاده خواهد شد. توسط رشته web_title عنوان صفحه مرورگر تعیین شده است، در رشته tag_hr خصوصیاتی مانند اندازه و سایز فونت نوشته ای که در آن از این تگ استفاده می کند مشخص می شود.

tag_br برای رفتن به خط بعدی در کد نویسی HTML مورد استفاده قرار می گیرد.

tag_form برای گرفتن اطلاعات از کاربر می باشد. (در مورد این تگ در تابع http_home توضیح می دهیم)

تابع http_get_ip :

از این تابع برای گرفتن آدرس IP از حافظه بافر (پس از فراخوانی تابع http_get_variable) که به صورت کد اسکی می باشد و تبدیل آن به معادل باینری و ذخیره سازی آن در متغیر dest استفاده می شود.

تابع http_get_variable :

از این تابع برای گرفتن متغیر در روش ارسال GET استفاده می شود.

از این تابع در تابع http_webserver_process برای دریافت متن های نوشته شده در جعبه های متنی و نمایش بر روی LCD استفاده می شود.

تابع hex2int :

از این تابع برای تبدیل یک کاراکتر hex به معادل عدد صحیح آن استفاده می شود. از این تابع در تابع urldecode استفاده می شود.

تابع http_put_request :

از این تابع برای نوشتن درخواست های http در بافر فرستنده استفاده می شود.

تابع urldecode :

با زدن دکمه Write LCD در صفحه وب طراحی شده، متن نوشته شده در کادرهای متنی به صورت کد URL به سمت میکروکنترلر ارسال می شوند. به عنوان مثال با نوشتن عبارات زیر در کادر متنی :

به دلیل استفاده از روش GET در HTTP (در تابع http_home توضیح داده می شود)، رشته ارسالی به سمت میکروکنترلر در نوار آدرس مروگر ظاهر می شود :

همان طور که مشاهده می کنید به دلیل استفاده از کد URL، فاصله (space) بین دو کلمه Ethernet و Project تبدیل به علامت + شد.

برای نمایش صحیح متن ارسالی بر روی LCD ، توسط این تابع کد URL رمزگشایی می شود و به عنوان مثال عبارت Ethernet+Project تبدیل به Project Ethernet می شود.

از این تابع در تابع http_webserver_process برای نمایش متن های نوشته شده در صفحه وب بر روی LCD استفاده می شود.

تابع http_webserver_process :

بررسی درخواست های HTTP در تابع http_webserver_process انجام می شود.

ورودی های تابع عبارت اند از حافظه بافر، آدرس MAC مقصد (dest_mac) و آدرس IP مقصد (dest_ip)

در بخش اول توسط دستور if شماره پورت TCP بررسی می شود.

به عنوان مثال برای کنترل LED1 از کدهای زیر در تابع استفاده شده است :

```
// get LED1 on/of command
if ( http_get_variable ( rxtx_buffer, dlength, "l1", generic_buf ) )
                {
                        if ( generic_buf[0] == '0' )
                                LED_PORT |= _BV ( LED_PIN1 );
                        else
                                LED_PORT &= ~_BV ( LED_PIN1 );
                }
```

همان طور که مشاهده می کنید با استفاده از شرط if وجود کلید I1 در درخواست http بررسی می شود و اگر درخواست مربوط به کلید I1 باشد، عملکرد مورد نظر روی پایه ی میکروکنترلر متصل به LED اعمال می شود.

_BV در فایل WINAVR_CAMP تعریف شده است که معادل کد زیر می باشد:

```
(1<<LED_PIN1 );
```

و همچنین عبارت :

~_BV (LED_PIN1)

معادل کد :

~(1<<LED_PIN1);

می باشد.

مشابه این کد برای LED2 نوشته شده است.

در تابع http_home برای ایجاد یک جعبه متن و گرفتن یک رشته از کد زیر استفاده می کنیم :

```
// Write LCD form

dlen = tcp_puts_data_p ( rxtx_buffer,  tag_form, dlen );

dlen = tcp_puts_data_p ( rxtx_buffer,  ( "<input name=\"lcd1\" type=\"text\" size=\"16\" maxlength=\"16\"> LCD Line 1<br><br>" ), dlen );

dlen = tcp_puts_data_p ( rxtx_buffer,  ( "<input name=\"lcd2\" type=\"text\" size=\"16\" maxlength=\"16\"> LCD Line 2<br><br>" ), dlen );

dlen = tcp_puts_data_p ( rxtx_buffer,  ( "<input type=\"submit\" value=\"Write LCD\"></form>" ), dlen );
```

کلید submit برای ارسال محتویات متن به سرور می باشد. کدهای مورد نیاز برای پاسخ دادن به این درخواست و نمایش رشته گرفته شده روی LCD به صورت زیر می باشد :

```
// get LCD string and show on first line

if ( http_get_variable ( rxtx_buffer, dlength, ( "lcd1" ), generic_buf ) )

        {

                urldecode ( generic_buf );

                lcd_putc ( '\f' );

                lcd_puts ( generic_buf );

                flag1.bits.lcd_busy = 1; }
```

توسط شرط if و تابع http_get_variable ، وجود رشته "lcd1" که در بخش name تگ input در ساخت جعبه متنی در تابع http_home استفاده کردیم بررسی میشود.

در این تابع ابتدا با استفاده از تابع urldecode کدهای URL موجود رمزگشایی می شوند.

سپس توسط تابع lcd_putc کاراکتر \f به سمت LCD ارسال می شود و سپس تابع lcd_puts محتویات موجود در generic_buf که معادل متن نوشته شده در جعبه متنی اول در صفحه وب می باشد بر روی سطر اول LCD نوشته می شود.

در نهایت lcd_busy که در فایل struct تعریف شده است یک می شود.

مشابه این کد برای سطر دوم LCD نوشته شده است.

تابع http_home :

تابع http_home جهت ایجاد صفحه وب نوشته شده است.

با برنامه ای که در این تابع نوشته شده است امکان کنترل دو LED که به پایه های دو و سه PPORTA میکروکنترلر وصل شده است وجود دارد. همچنین دو ورودی متن نیز در صفحه HTML ایجاد می شود که متن LCD کاراکتری متصل به میکرو را با استفاده از آنها می توان تغییر داد.

کدهای HTML به صورت سطر به سطر و توسط تابع tcp_puts_data_p به درخواست کننده (در این پروژه کامپیوتر) ارسال می شوند. (از طریق نوشتن در بافر rxtx_buffer)

در ابتدای تابع، توسط کد زیر پیاده سازی پروتکل HTTP/1.0 (نسخه 1.0) انجام می شود.

```
HTTP/1.0 200 OK\r\nContent-Type: text/html\r\n\r\n
```

توسط تگ <title> عنوان صفحه برابر ثابت web_title (که در ابتدای صفحه تعریف شده است) تنظیم می شود.

در ادامه توسط تگ <a> و مشخصه href یک لینک به سایت www.knowledgeplus.ir توسط عبارت knowledgeplus AVR-Ethernet Board ایجاد می شود.

از ثابت tag_hr که در ابتدای فایل تعریف شده است برای تعیین فرمت عبارت Test Project استفاده شده است.

در ادامه، کدهای مربوط به طراحی دکمه های کنترل LED و جعبه های متن نوشته شده است.

به عنوان مثال در صورتی که بخواهیم یک صفحه ساده HTML که تنها شامل یک لینک می باشد ایجاد کنیم، تابع http_home باید به صورت زیر تغییر کند :

```
uint16_t http_home( uint8_t *rxtx_buffer )

{
```

```
Uint16_t dlen;

dlen = tcp_puts_data_p ( rxtx_buffer ,  PSTR ("HTTP/1.0 200 OK\r\nContent-
Type:text/html\r\n\r\n" ), 0 );

dlen = tcp_puts_data_p ( rxtx_buffer ,  PSTR ("<title>"), dlen );

dlen = tcp_puts_data_p ( rxtx_buffer ,  (PGM_P)web_title, dlen);

dlen = tcp_puts_data_p ( rxtx_buffer ,  PSTR ("<title>"), dlen );

dlen = tcp_puts_data_p ( rxtx_buffer ,  PSTR ("<a href=\http://www.
knowledgeplus.ir/\target=\"_blank\"><b><font color=\"#000099\" size=\"+1\">" ), dlen );


return(dlen);

}
```

اگر بخواهیم لینک هایی ایجاد کنیم که عملکرد میکروکنترلر را کنترل کنند (مثلا روشن و خاموش کردن LED)، باید از حالت زیر در تعریف لینک ها استفاده کنیم :

```
dlen = tcp_puts_data_p ( rxtx_buffer ,  ("<a href=\"./?I1="), dlen );
```

کلید I1 که در این لینک تعریف شده است برای مشخص کردن یک عملکرد خاص می باشد و بعدا برای تشخیص عملکردهای مختلف که توسط لینک ها ایجاد می شوند استفاده خواهد شد. بعد از کاراکتر مساوی، عدد یا رشته ای قرار می گیرد که نشان خواهد داد مثلا آن LED باید روشن یا خاموش شود. مثلا برای روشن کردن LED کد بالا به صورت زیر تغییر می کند :

```
dlen = tcp_puts_data_p ( rxtx_buffer ,  ("<a href=\"./?I1=1"), dlen );
```

در بخش انتهای تابع که با توضیح Write LCD form مشخص شده است،

با استفاده از تگ form دو کادر طراحی کردیم که متن نوشته شده در کادر اول مربوط به سطر اول LCD و متن نوشته شده در کادر دوم مربوط به سطر دوم LCD است. با زدن دکمه Write LCD این دو متن در سطر اول و دوم LCD نوشته می شوند.

برای استفاده از این تگ باید خصوصیات (element) این تگ را تنظیم کنیم.

یکی از مهم ترین خصوصیات این تگ، input می باشد. توسط این خصوصیت می توان نوع دریافت اطلاعات را از کاربر مشخص کرد. به عنوان مثال :

```
<input type="text">
```

برای ایجاد یک کادر متنی،

```
<input type="radio">
```

برای ایجاد گزینه های مختلف یا :

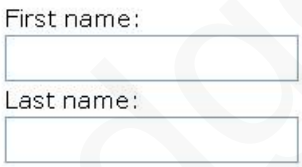
```
<input type="submit">
```

برای ساختن دکمه submit یا تایید اطلاعات (دکمه write lcd در این پروژه)

به عنوان مثال با نوشتن کد :

```
<form>
 First name:<br>
 <input type="text" name="firstname"><br>
 Last name:<br>
 <input type="text" name="lastname">
</form>
```

این نتیجه حاصل می شود :



درابتدا ثابت tag_form نوشته می شود که شروع کننده ی این تگ می باشد.

در خط بعدی کادر اول ساخته می شود :

```
<input name=\"lcd1\" type=\"text\" size=\"16\" maxlength=\"16\"> LCD Line 1<br><br>
```

نکته مهم : مشخصه name حتما باید نوشته شود، در غیر اینصورت اطلاعات نوشته شده در کادر متنی ارسال نمی شود.

Size اندازه فونت ها را مشخص می کند، maxlength حداکثر تعداد کاراکتر قابل نوشتن در کادر را مشخص می کند (که باتوجه به استفاده از LCD ۱۶ کاراکتری، این عدد برابر ۱۶ تنظیم شده است) و سپس عنوان کادر به نام LCD Line 1 در کنار کادر نوشته می شود.

در خطر بعد نیز به همین شکل کادر لازم برای سطر دوم LCD ایجاد می شود.

توسط تعریف type برابر submit ، دکمه لازم برای تایید و ارسال متن ها به سوی میکروکنترلر ساخته می شود. در بخش value نام این دکمه Write LCD قرار داده می شود و توسط </form> تگ form

به پایان می رسد.

در بخش بعدی نیز یک لینک با نام Refresh ساخته می شود و عبارت www.knowledgeplus.ir نیز در پایین آن نوشته می شود.

در انتها متغیر dlen به عنوان خروجی تابع بازگردانده می شود.

در تعریف ثابت tag_form که در ابتدای برنامه در حافظه flash تعریف شده است و در این بخش از آن استفاده شد، مشخصه action مطابق کد زیر تنظیم شده است :

```
<form action=\"./?\" method=\"get\
```

action تعیین کننده عملیاتی است که پس از زدن دکمه تایید روی صفحه باید صورت بپذیرد. به عنوان مثال معمولا در صفحات وب با زدن تایید، اطلاعات تنظیم شده توسط کاربر به سمت یک صفحه وب در سمت سرور ارسال می شود. به عنوام مثال با نوشتن کد :

```
<form action="action_page.php">
```

اطلاعات به سمت صفحه action_page.php که توسط سرور طراحی شده است ارسال می شود.

در کد نوشته شده، با توجه به این که صفحه ای در سمت سرور (میکروکنترلر) تعریف نشده است، آدرس صفحه ای نوشته نشده است.

در بخش method روش ارسال اطلاعات در پروتکل HTTP (معمولا GET یا POST) تعیین می شود. در اینجا روش GET انتخاب شده است.

نشانه ی استفاده از این روش این است که پس از زدن دکمه تایید اطلاعات، اطلاعات نوشته شده در نوار آدرس مرورگر نوشته می شود. از این روش زمانی باید استفاده کرد که حجم اطلاعات ارسالی کم می باشد. (شاید در هنگام تست این پروژه گاهی اوقات با نوشتن چند باره در کادر متنی و زدن دکمه تایید، متن نوشته شده روی LCD نمایش داده نشود که علت این مسئله همین موضوع است)

در مقابل در روش POST ، پس از زدن دکمه تایید اطلاعات، اطلاعات نوشته شده در نوار آدرس مرورگر نوشته نمی شود. در این روش محدودیتی برای ارسال اطلاعات وجود ندارد.

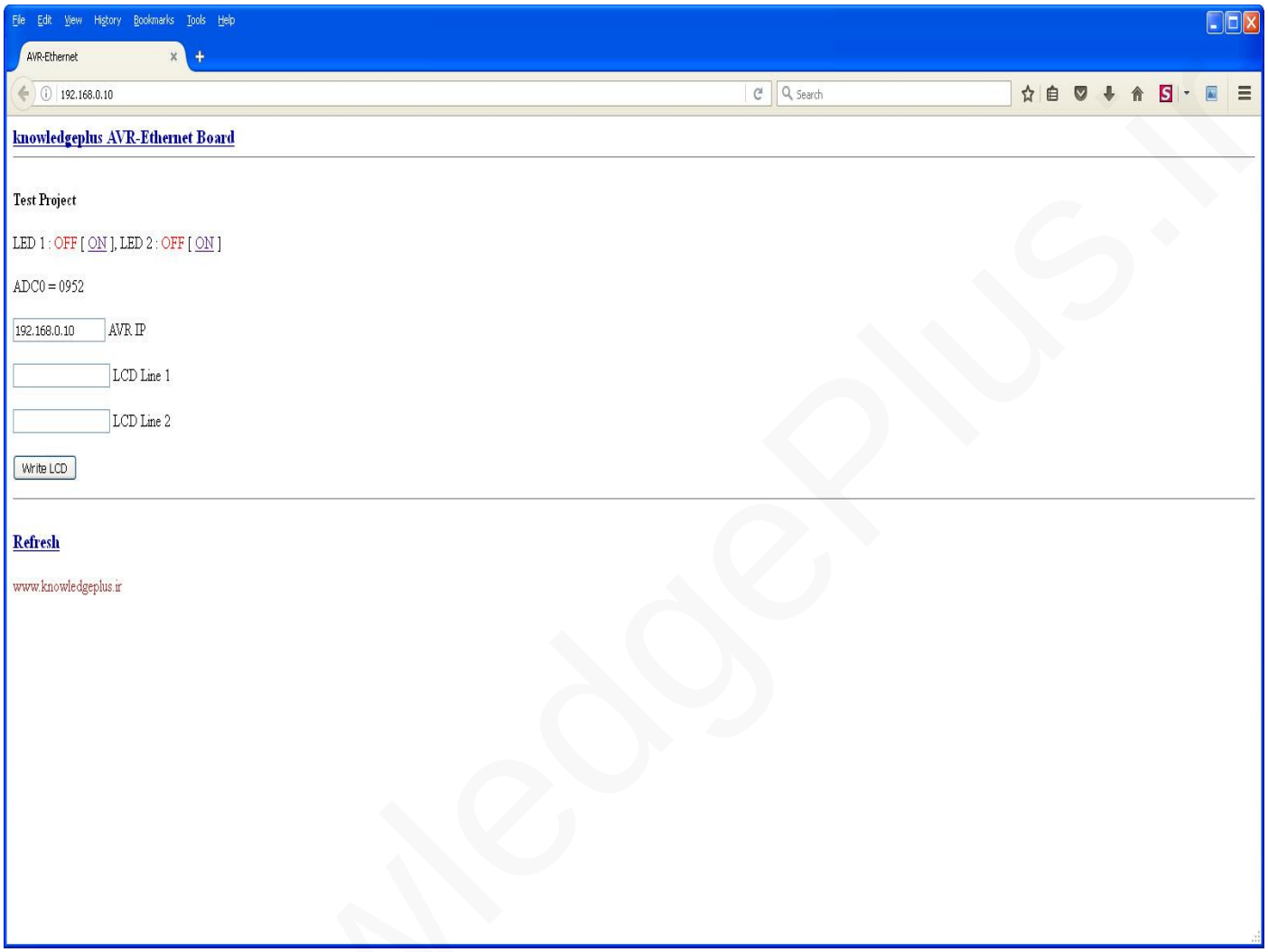
قبل از این بخش که با توضیح AVR IP address مشخص شده است، IP میکروکنترلر باید نوشته شود.

ابتدا توسط tag_form و مشخصه input مطابق توضیحات ارائه شده، کادر مناسب برای وارد کردن IP طراحی می شود.

سپس توسط تابع print_ip که در فایل menu تعریف شده است، متغیر avr_ip در این کادر نوشته

می شود.

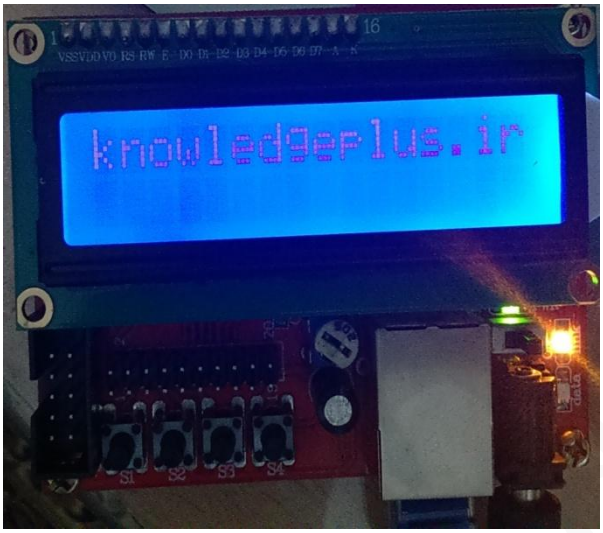
نتیجه کدهای نوشته شده در این کتابخانه مطابق شکل زیر می شود :



شکل ۸-۸

۸-۳-۳-۱۲. تست عملی پروژه

برای تست مدار، ابتدا سورس برنامه را کامپایل نموده و توسط پروگرامر روی میکروکنترلر پروگرام نمایید.
تنظیمات فیوز بیت را روی کریستال 8MHZ خارجی قرار دهید.
مدار را توسط کابل شبکه به پورت LAN کامپیوتر وصل کرده و روشن کنید. با روشن کردن دستگاه، صفحه زیر ظاهر می شود :

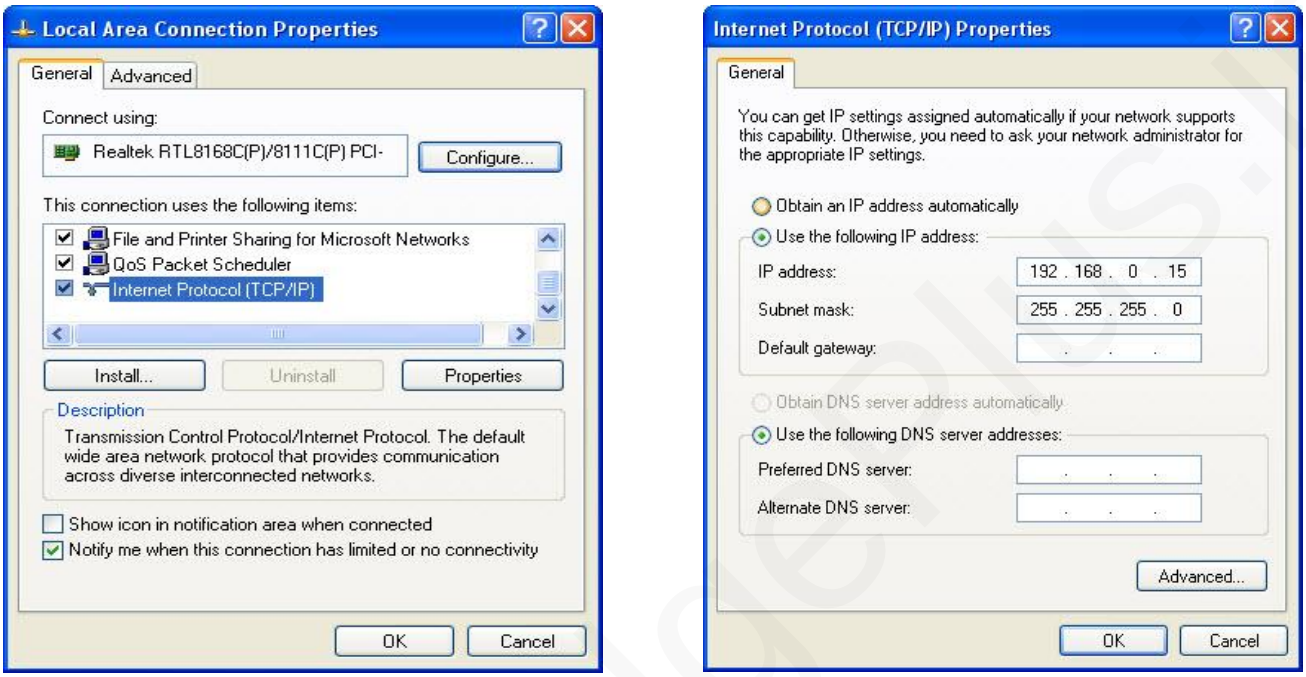


شکل ۸-۹

به صورت پیشفرض IP برد روی 255.255.255.255 تنظیم شده است. با استفاده از کلیدهای جهت نما وارد منوی تنظیمات و بخش AVR IP Config شوید و IP را طبق رنج IP شبکه خود تنظیم کنید.
برای مثال اگر رنج IP شبکه 192.168.0.xx باشد، IP را روی 192.168.0.10 تنظیم کنید.

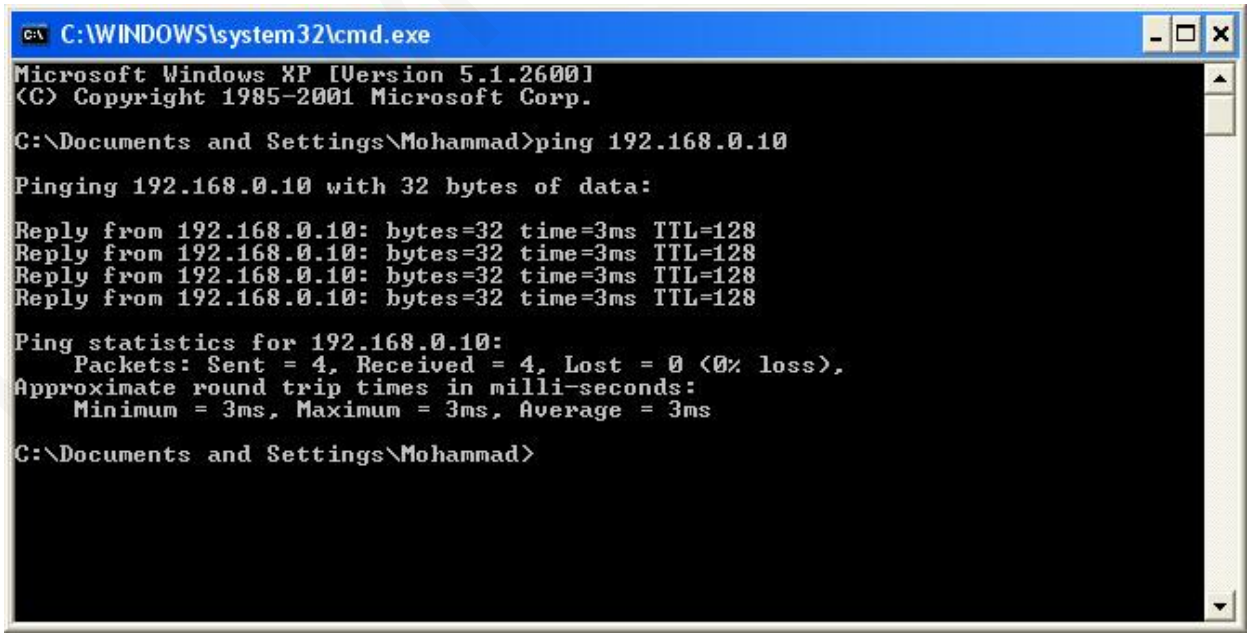
شکل ۸-۱۰

نکته : مطابق شکل های زیر در بخش تنظیمات TCP/IP کارت شبکه، باید رنج IP شبکه را به شکل مناسبی تنظیم کنیم. به عنوان مثال برای تطبیق با IP برابر 192.168.0.10 ، می توان این آدرس را روی 192.168.0.15 تنظیم کرد.



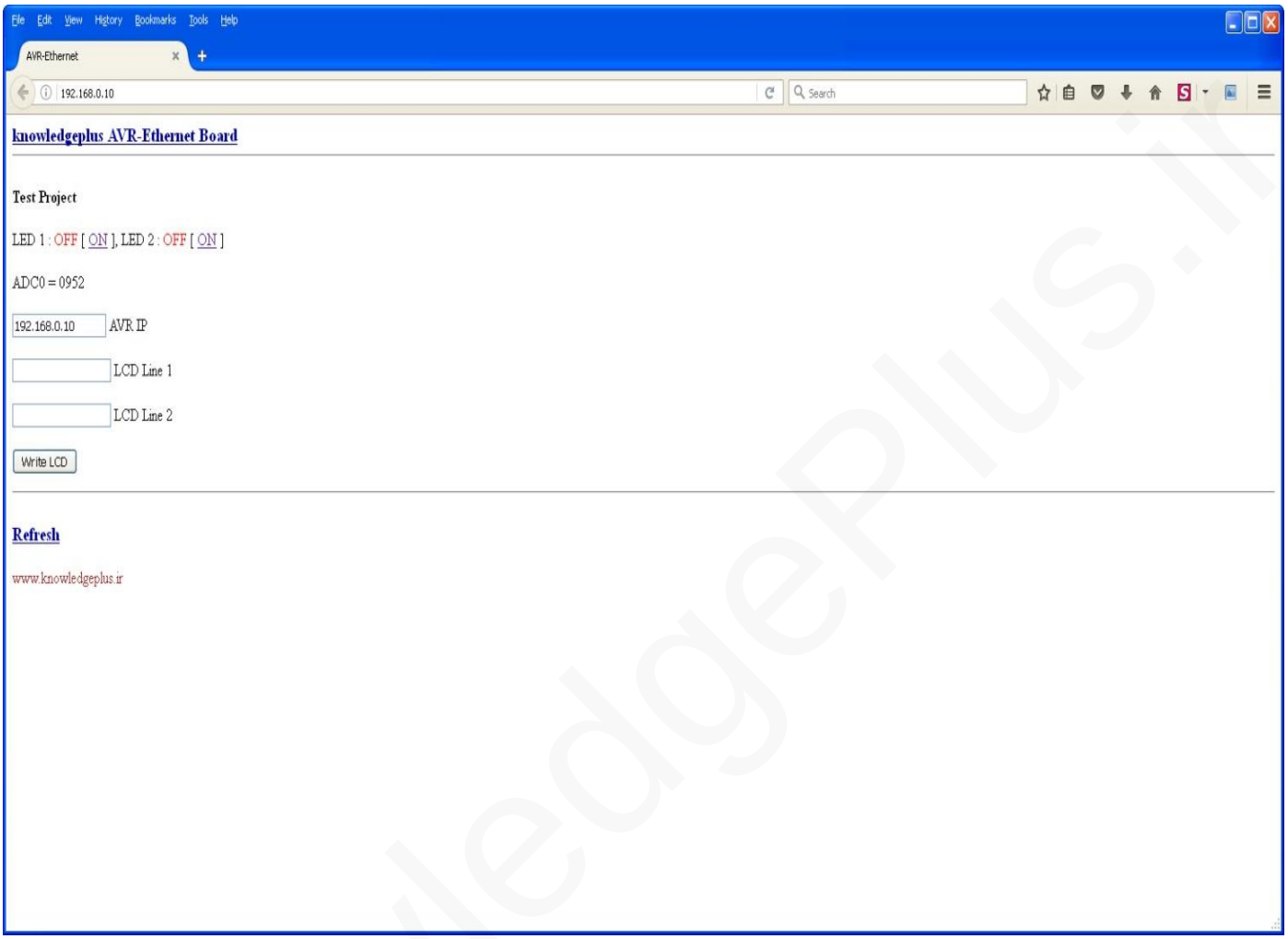
شکل ۸-۱۱

شکل زیر نتیجه دستور ping را برای تست ارتباط نشان می دهد.



شکل ۸-۱۲

مرورگر اینترنتی خود را باز کنید و IP برد را وارد کنید. حال باید صفحه ای شبیه تصویر زیر را روی مرورگر مشاهده کنید. (فیلم تست عملی سخت افزار)



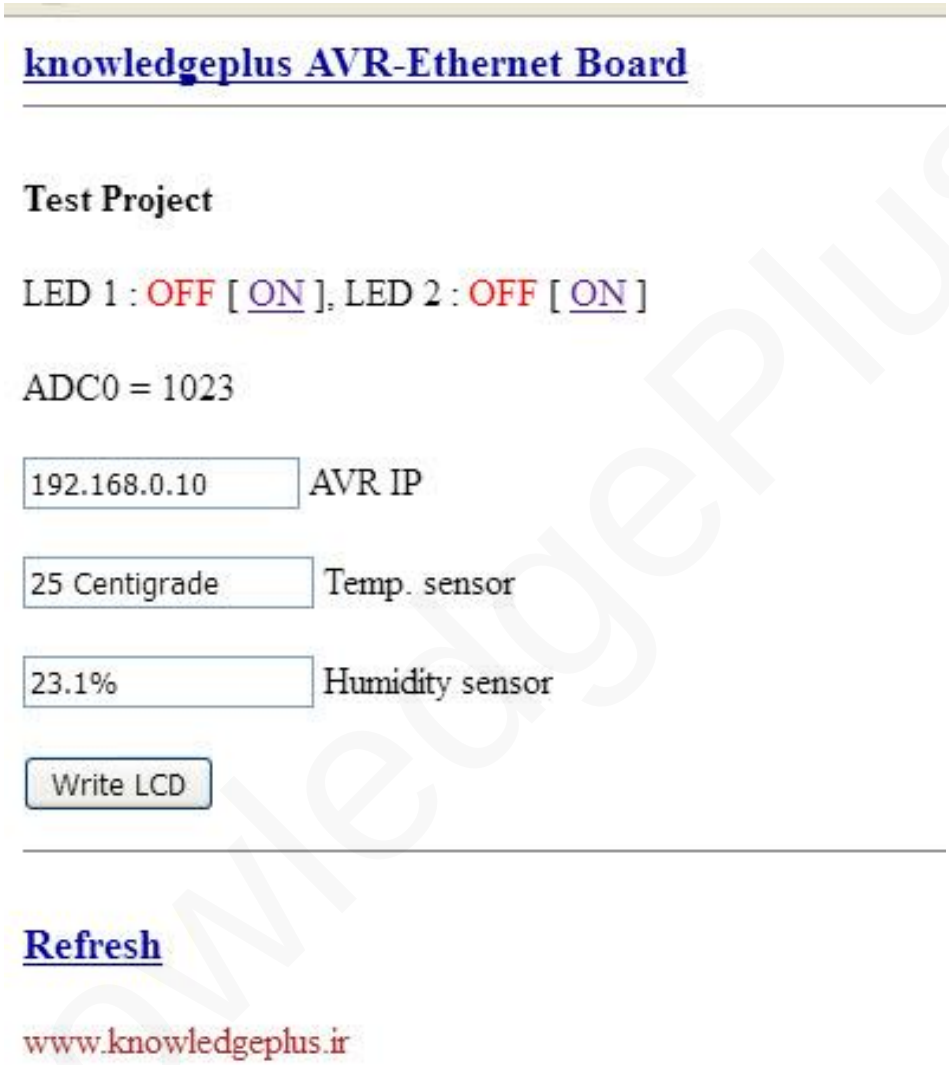
شکل ۸-۱۳

با نوشتن در کادر LCD Line 1 و LCD Line 2 و زدن دکمه Write LCD مطابق کدهایی که در بخش های قبل توضیح داده شد، متن نوشته شده در این دو جعبه متنی توسط دستور GET در پروتکل HTTP به سمت میکروکنترلر ارسال می شود و بر روی سطر اول و سطر دوم LCD متصل شده به میکروکنترلر نمایش داده می شود.

شکل ۸-۱۴

شکل ۸-۱۵

برای گسترش این پروژه می توانید به پایه های PORTA میکروکنترلر برد طراحی شده که توسط pin header در دسترس می باشد، سنسورهای مختلف متصل نموده و مقادیر خوانده شده از سنسورها را در صفحه وب طراحی شده نمایش دهید. (فیلم تست عملی)



شکل ۸-۱۶

با اتصال برد طراحی شده به یک switch شبکه و تنظیم صحیح آدرس های IP ، امکان دسترسی گره های مختلف شبکه به وب سرور محلی طراحی شده وجود خواهد داشت.

همچنین توسط اتصال switch به یک router که به عنوان access point عمل می کند (یا استفاده مستقیم از یک router) و انجام تنظیمات مربوطه، امکان دسترسی به وب سرور محلی به صورت wireless نیز وجود خواهد داشت. (فیلم تست عملی)

شکل ۸-۱۷

نکته اول :

هدف از پروژه ارائه شده در این فصل، پیاده سازی وب سرور در شبکه های محلی مبتنی بر اترنت می باشد. برای اتصال شبکه محلی یا سخت افزار طراحی شده به شبکه های بزرگتر مانند اینترنت، نیاز است با توجه به کاربرد مورد نظر برای اتصال به اینترنت (ارسال و دریافت ایمیل، ارسال فایل و...) پروتکل های مورد نیاز در لایه کاربرد پیاده سازی شود.

نکته دوم :

مطابق آنچه توضیح داده شد، برای تست سخت افزار و کدهای نوشته شده کافی است سخت افزار را با کابل CAT به یک کامپیوتر متصل کرده و آدرس IP کارت شبکه را برای شناسایی سخت افزار تنظیم نمایید.

در کاربرد های عملی معمولا ارتباطات در قالب یک شبکه صورت می پذیرد، این شبکه یا از قبل طراحی شده است و یا باید با توجه به نیازهای تعریف شده طراحی می شود. در این حالت باید تمامی ملاحظات مربوط به پیاده سازی شبکه های کامپیوتری مانند تنظیمات صحیح کارت شبکه و تجهیزات شبکه رعایت گردد.

۸-٤. پروژه های پیشنهادی

۱- یک رله به برد اترنت متصل کرده و با تغییر نرم افزار پروژه، این رله را از طریق شبکه کنترل کنید.
۲- چند سنسور مختلف مانند سنسور دما، رطوبت و فشار به برد اترنت متصل کرده و با تغییر نرم افزار پروژه، اطلاعات دریافت شده از سنسورها را در شبکه ارسال کنید. (یک مثال ساده برای کاربرد گلخانه های هوشمند)
۳- با تغییر ساختار پروژه، یک مبدل اترنت به سریال (RS-232) طراحی کنید.
۴- با پیاده سازی پروتکل های دیگر لایه Application مانند FTP ، Telnet و ... کاربردهای دیگر قابل پیاده سازی در بستر اترنت را اجرا کنید.

در پایان ضمن تقدیر از مهندس اوژن کی نژاد جهت حمایت از این پروژه آموزشی، باعث افتخار بنده است اگر سوالات، نظرات یا پیشنهادات خود را از طریق راه های ارتباطی زیر با ما در میان بگذارید.

09361955350

Mohammadghamsari@ieee.org

mohammadghamsari18@yahoo.com

پیوست ها

پیوست الف) ساختن کابل های Straight-Through و Cross-Over

کابل کشی شبکه یکی از مراحل مهم در زمان پیاده سازی یک شبکه کامپیوتری است که می بایست با دقت، ظرافت خاص و پایبندی به اصول کابل کشی صورت پذیرد.

اگرچه کابل هایی که در فصل چهارم به آنها اشاره شد به راحتی در بازار پیدا می شوند و قیمت نسبتا مناسبی دارند، با این حال در این بخش به طور خلاصه مراحل ساخت دو کابل Straight-Through و Cross-Over را که در فصل چهارم معرفی شدند را مرور می کنیم.

همان طور که در فصل چهارم اشاره شد، به منظور اتصال کانکتورهای RJ-45 به کابل های CAT از دو استاندارد رایج T-568A و T-568B استفاده می گردد. نحوه عملکرد دو استاندارد فوق یکسان بوده و تنها تفاوت موجود مربوط به ترتیب رنگ سیم هایی است که به کانکتور متصل می شود. در فصل چهارم به طور کامل در مورد نحوه ی اتصال کانکتور به کابل و استانداردها توضیح داده شد، در این بخش نیز اشاره ای به این دو استاندارد می کنیم.

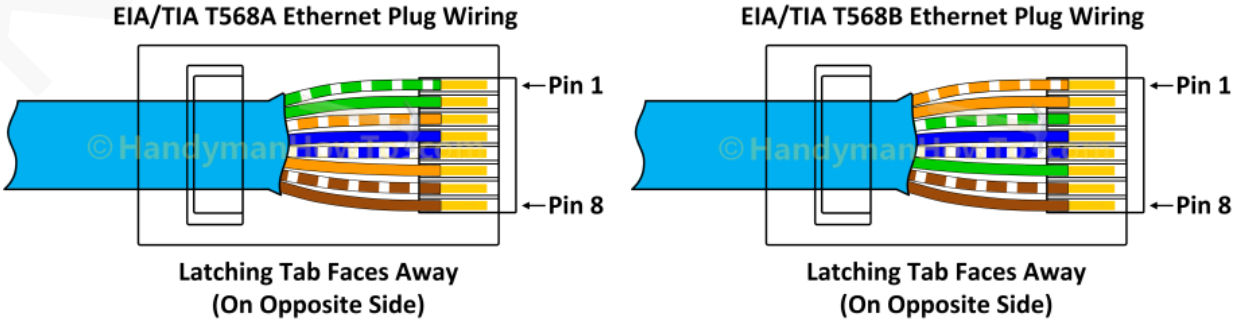
جدول زیر شماره پایه های استاندارد T-568B را نشان می دهد. همان طور که مشاهده می شود شماره پایه های فرد همواره سفید بوده که با یک نوار رنگی پوشش داده می شوند.

کاربرد	زوج	رنگ	شماره پایه
TxData+	دوم	سفید/ نارنجی	یک
TxData-	دوم	نارنجی	دو
RecvData+	سوم	سفید / سبز	سه
	یک	آبی	چهار
	یک	سفید /آبی	پنج
RecvData-	سوم	سبز	شش
	چهارم	سفید/قهوه ای	هفت
	چهارم	قهوه ای	هشت

جدول زیر شماره پایه های استاندارد T-568A را نشان می دهد. در استاندارد T-568A اتصالات سبز و نارنجی برعکس شده است، بنابراین زوج های یک و دو بر روی چهار پایه وسط قرار می گیرند.

کاربرد	زوج	رنگ	شماره پایه
RecvData+	سوم	سفید/ سبز	یک
RecvData-	سوم	سبز	دو
TxData+	دوم	سفید / نارنجی	سه
	یک	آبی	چهار
	یک	سفید /آبی	پنج
TxData-	دوم	نارنجی	شش
	چهارم	سفید/قهوه ای	هفت
	چهارم	قهوه ای	هشت

شکل های زیر ترتیب پایه ها را طبق دو استاندارد T-568A و T-568B را نشان می دهند.



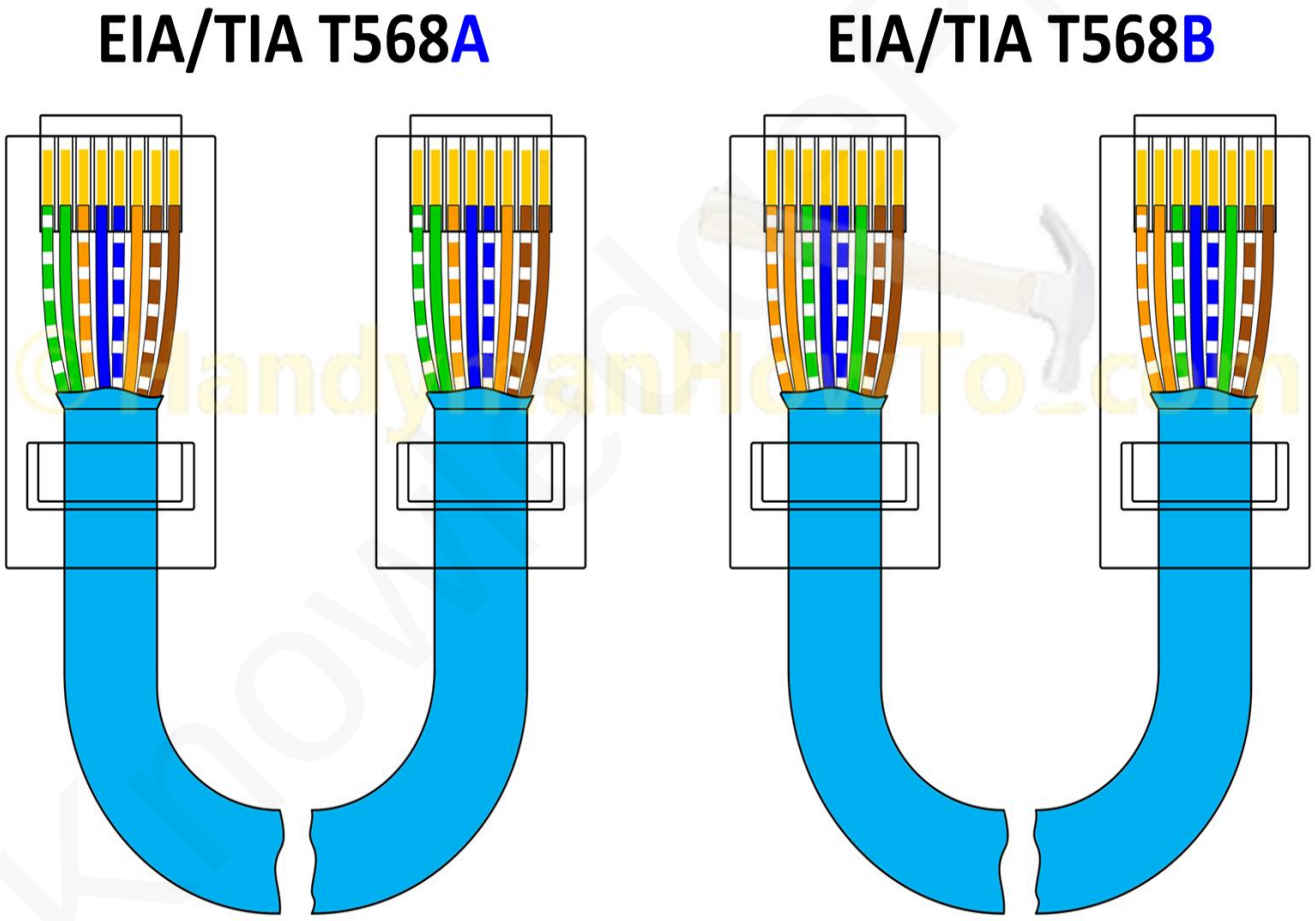
سوال : از کدام استاندارد استفاده کنیم؟

استاندارد T-568B استاندارد جدیدتری است و در بیشتر تجهیزات و شبکه های امروزی استفاده می شود. در کشور آمریکا بیشتر سیم های جدید بر مبنای این استاندارد هستند.

اگر شبکه ای که در حال سیم بندی هستید جدید است و خودتان دارید طراحی می کنید، می توانید از هر کدام از استانداردها استفاده کنید، ولی اگر شبکه ای در حال حاضر وجود دارد، بهتر است از استاندارد سیم های متصل شده به شبکه پیروی کنید.

نکته : اگر در دو سر کابل از دو استاندارد متفاوت استفاده کنیم، کابل Cross-over ساخته می شود.

شکل استفاده زیر از این دو استاندارد را برای اتصال کابل CAT به کانکتور RJ-45 نشان می دهد.



در استاندارد های سرعت های 10Mbit/s و 100Mbit/s از زوج های دو و سه (رنگ نارنجی و سبز) استفاده می شود و زوج های یک و چهار رزو شده می باشند اما پیشنهاد می شود که تمام سیم ها در هنگام طراحی کابل اتصال پیدا کنند و با این کار قابلیت استفاده از کابل در شبکه های دیگر نیز وجود دارد. همچنین از دو زوج دیگر (رنگ قهوه ای و آبی) می توان به منظور یک خط اترنت دوم و یا اتصالات تلفن استفاده نمود.

در استاندارد های اترنت گیگابیت، از تمامی چهار زوج استفاده می گردد. به عنوان مثال در 10/100Base-T تنها سیم های 1,2,3 و 6 استفاده می شوند. در 100Base-T4 یا شبکه های پرسرعت تر مانند 1000Base-T از تمام چهار زوج سیم استفاده می شود.

کابل CAT5 متداولترین نوع کابل UTP بوده که دارای انعطاف مناسب بوده و نصب آن به سادگی انجام می شود.

نکته : در کابل CAT5e ، e مخفف enhanced یا بهبود یافته و به معنای کاهش هم شنوایی یا crosstalk بین سیم های کنار یکدیگر می باشد.

- تجهیزات لازم

- کابل CAT
- کانکتور RJ-45
- آچار پرس RJ-45 (Cripper)
- سیم لخت کن

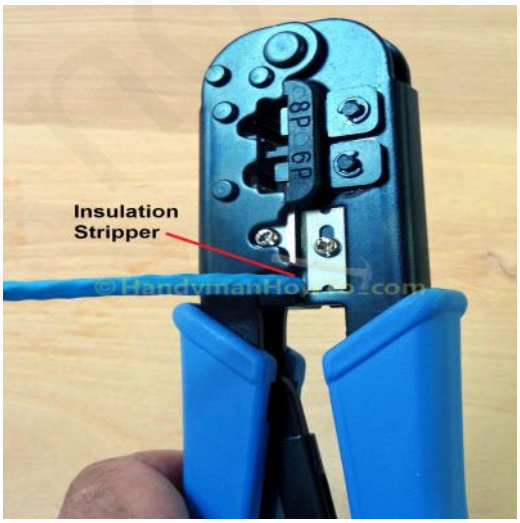
شکل زیر یک نمونه آچار پرس برای کابل های RJ-45 و کانکتورهای مشابه را نشان می دهد.

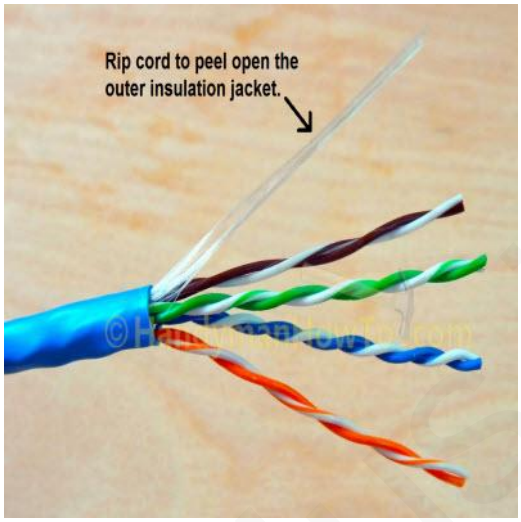
- مراحل ساخت کابل

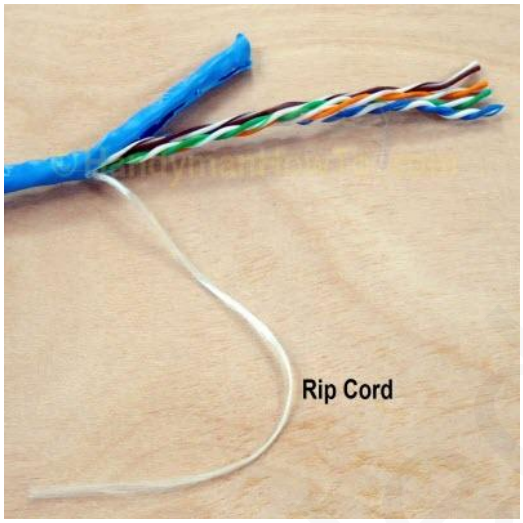
۱. ابتدا 1.5 تا 2 اینچ از عایق بیرونی سیم را جدا می کنیم.
۲. سیم ها را طبق استاندارد مورد نظر مرتب می کنیم.
۳. سرهای سیم ها را برای ورود به کانکتور به اندازه مناسب می بریم.
۴. سیم ها را به آهستگی و بدون به هم خوردن ترتیب آنها وارد کانکتور می کنیم تا به بخش هادی کانکتور متصل شوند.
۵. توسط ابزار مناسب، کانکتور و کابل را به یکدیگر متصل می کنیم.

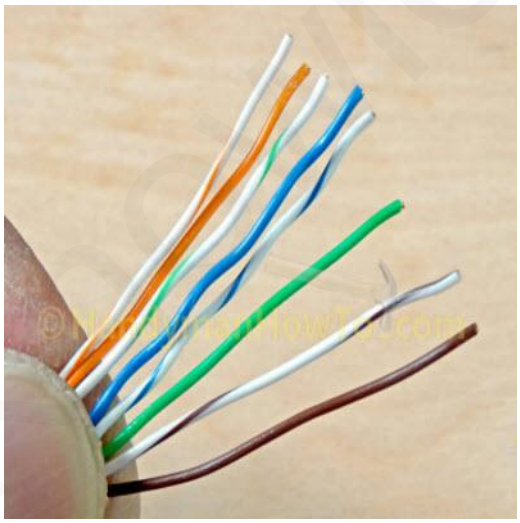
نکته : به منظور تسهیل در امر نگهداری، می بایست به اندازه ضروری سیم های بهم تابیده را از حالت پیچش خارج نمود. (مثلا حدود یک سانتیمتر)

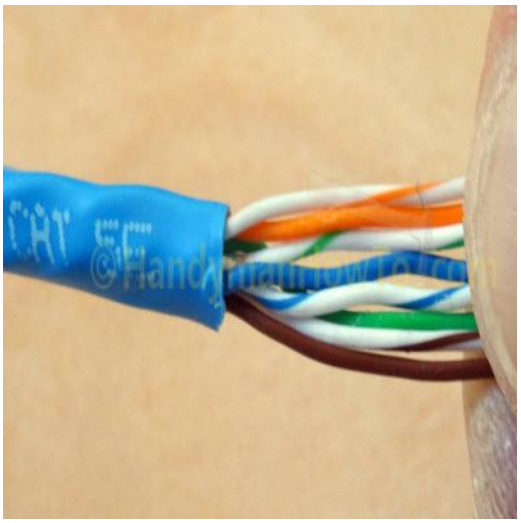
شکل های زیر مراحل توضیح داده شده را نشان می دهند.

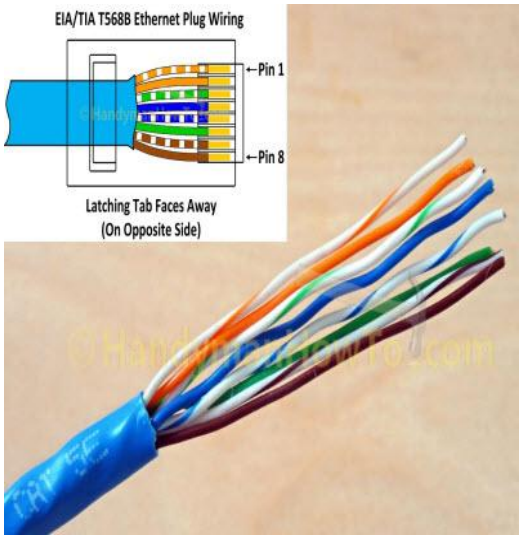


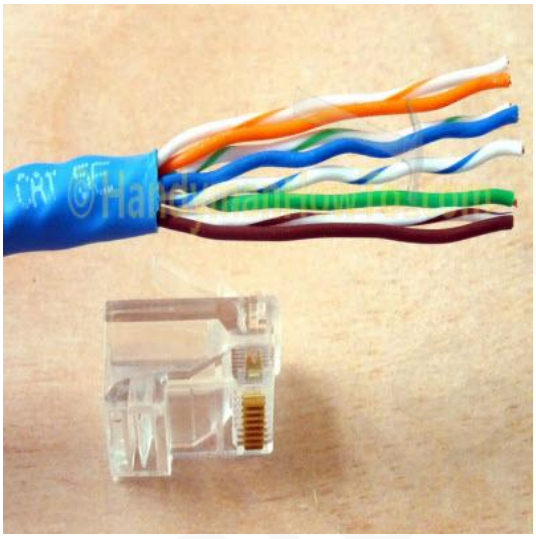

Rip cord to peel open the
outer insulation jacket.

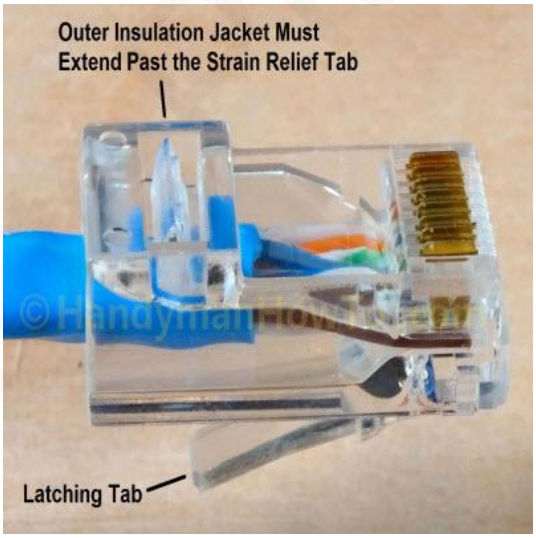

Rip Cord

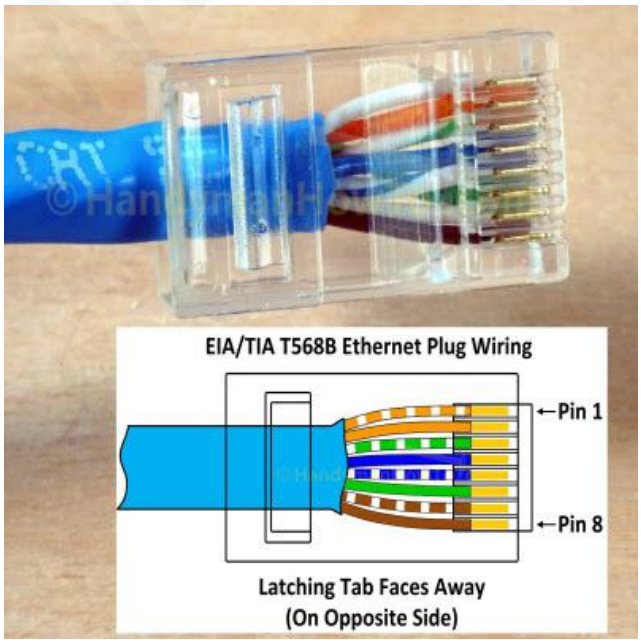




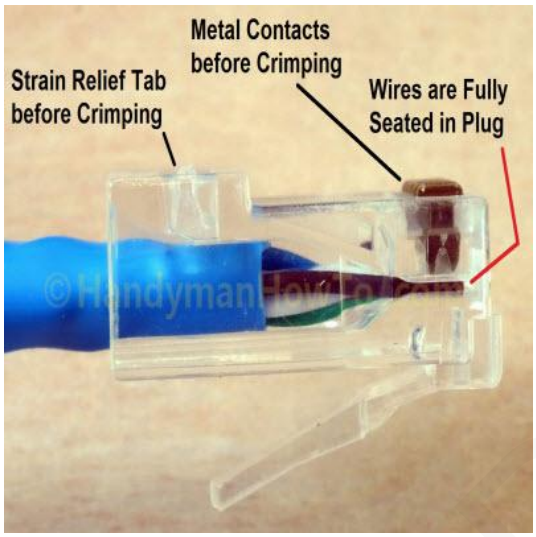

EIA/TIA T568B Ethernet Plug Wiring
←Pin 1
←Pin 8
Latching Tab Faces Away
(On Opposite Side)



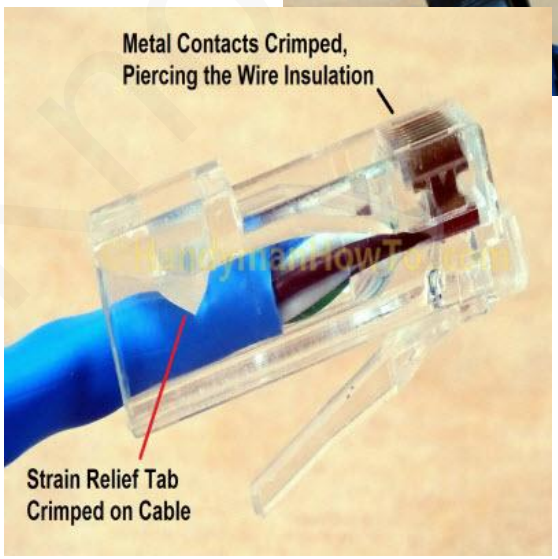

Outer Insulation Jacket Must
Extend Past the Strain Relief Tab
Latching Tab

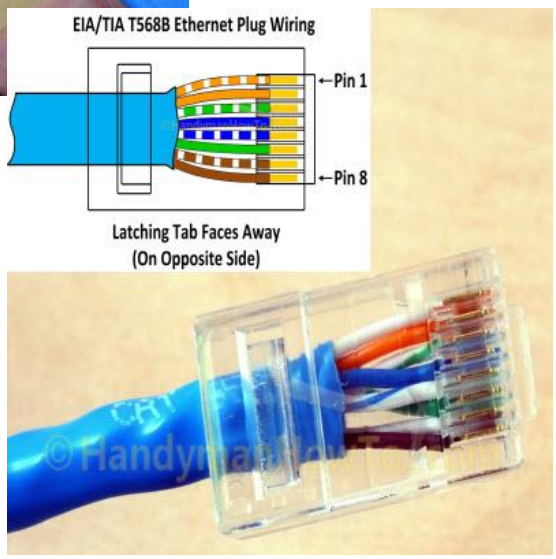

EIA/TIA T568B Ethernet Plug Wiring
←Pin 1
←Pin 8
Latching Tab Faces Away
(On Opposite Side)

شکل زیر وضعیت کانکتور و سیم را قبل از اتصال نشان می دهد.



شکل های زیر وضعیت کانکتور و سیم را بعد از اتصال نشان می دهند.

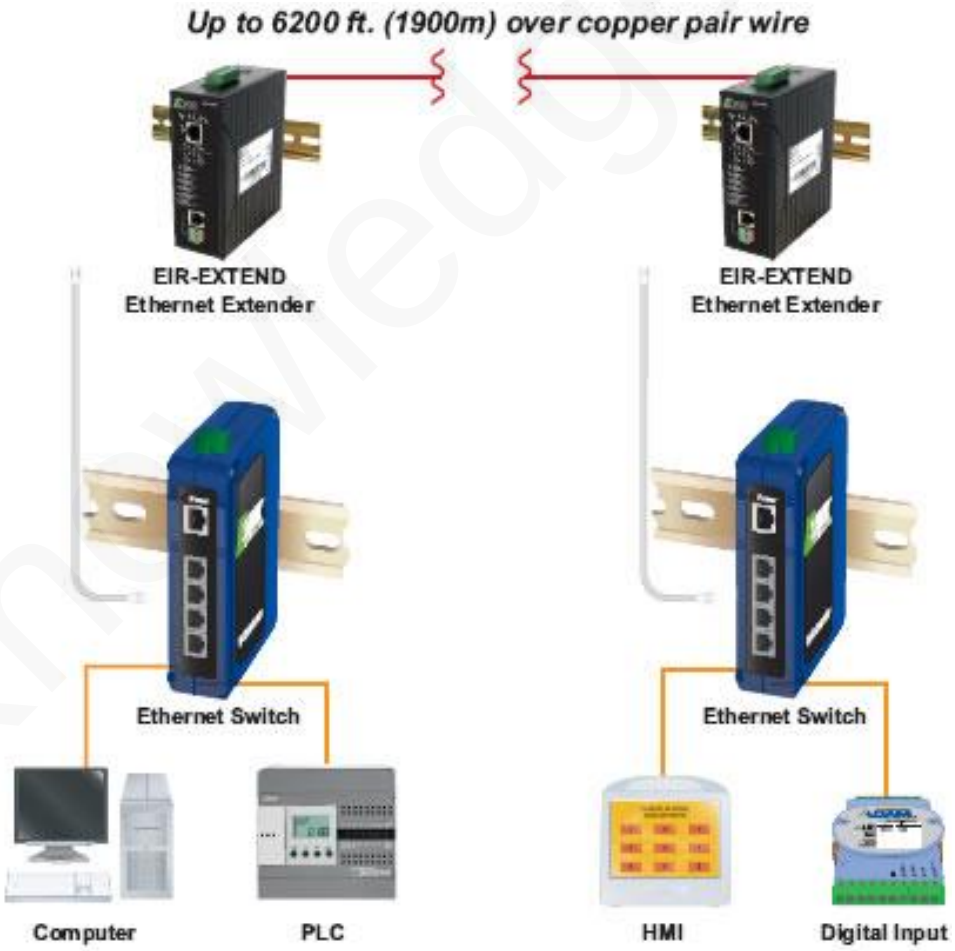




نکته : برای تست کابل، یک سر آن را به پورت شبکه کامپیوتر و سر دیگر آن را به تجهیزات شبکه مانند سوییچ یا یک گره اترنت دیگر وصل کنید. اگر عملیات سخت افزاری درست انجام شده باشد، باید چراغ های وضعیت اتصال روی کانکتور RJ-45 یا تجهیزات شبکه روشن شود. همچنین می توانید برای تست کابل از تجهیزات تست کابل (cable tester) استفاده نمایید.

پیوست ب) توسعه دهنده اترنت (Ethernet Extender)

توسعه دهنده اترنت یا Ethernet Extender (network extender یا LAN extender)، به تجهیزاتی گفته می شود که امکان افزایش مسافت شبکه را فراتر از آنچه در استانداردهای اترنت تعیین شده است (معمولا حدود ۱۰۰ متر) می دهند.



پیوست ج) ایزوله کننده شبکه (Network Isolator)

ایزوله کننده شبکه یا Network isolator، تجهیزاتی غیرفعال یا passive (به معنای عدم نیاز به منبع تغذیه) هستند که اغلب در شبکه های اترنت مبتنی بر کابل های مسی (مانند کابل های CAT) به عنوان ایزوله کننده الکتریکی استفاده می شوند.

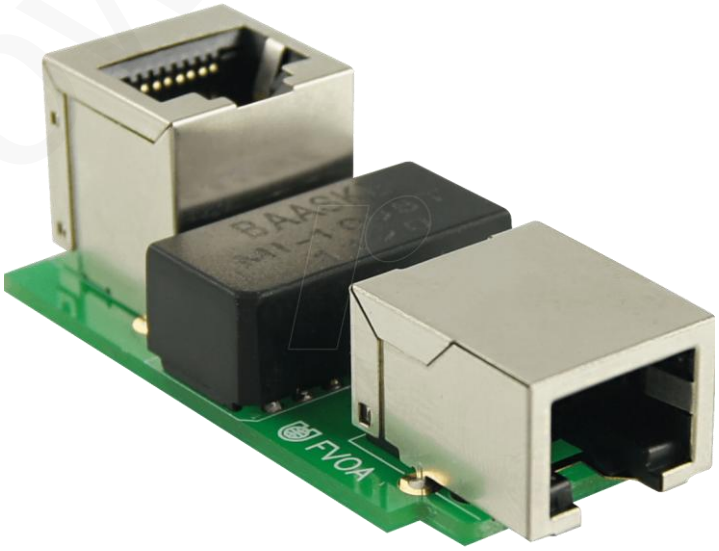
این تجهیزات با ایجاد ایزولاسیون الکتریکی، مانع از ایجاد اختلال در تجهیزات متصل شده به شبکه در اثر نوسانات در شبکه یا در اثر اختلاف پتانسیل بین گره های شبکه می شوند.

یکی از مهم ترین کاربرد های ایزوله کننده شبکه، در تجهیزات پزشکی می باشد که ایجاد اختلاف پتانسیل در زمین دستگاه متصل شده به شبکه ممکن است باعث ایجاد جریان های نشتی و آسیب رسیدن به مصرف کننده دستگاه بشود.

از دیگر کاربردهای این تجهیزات عبارت است از :

- در تجهیزات تست و نظارت شبکه
- سیستم های دارای چندین سرور (redundant server systems)
- در شبکه های خصوصی یا تجاری که امکان به وجود آمدن اختلاف پتانسیل در زمین شبکه وجود دارد و استفاده از فیبر نوری از لحاظ اقتصادی امکان پذیر نیست

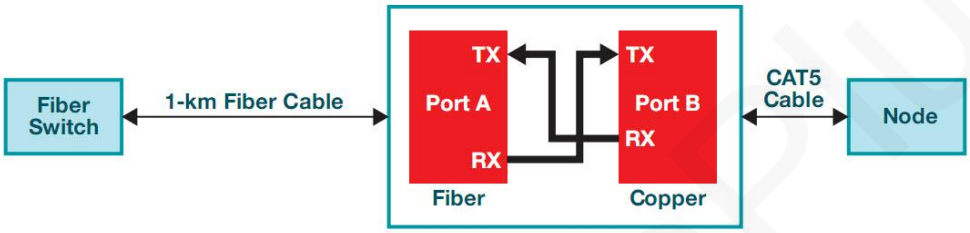
از جمله مشخصات این تجهیزات حداکثر ولتاژ (برحسب کیلو ولت) و حداکثر سرعت انتقال مورد پشتیبانی (مثلا 1000Base-T) می باشد.



پیوست د) تبدیل کننده فیبر نوری (Fiber Media Converter)

تبدیل کننده فیبر نوری یا Fiber media converter، برای تبدیل کابل های فیبر نوری به کابل های مسی مانند CAT در شبکه هایی که از فیبر نوری پشتیبانی نمی کنند استفاده می شود.

در فصل ششم نیز به عنوان یکی از کاربردهای تراشه DP83849IF نیز به همین دستگاه اشاره گردید که در شکل زیر نشان داده شده است.



شکل زیر یک نمونه از این دستگاه ها را برای استفاده در تکنولوژی Fast Ethernet برای تبدیل استاندارد 100Base-FX به 100Base-TX نشان می دهد.

پیوست ه) مقایسه اینترنت، اترنت، اینترانت و اکسترانت

تمامی این اصطلاحات در کلمات خود دارای شباهت و اشتراک می باشند. دانستن شباهت ها و تفاوت های آنها می تواند در استفاده صحیح و مناسب آنها مفید باشد.

از جمله شباهت های اینترنت، اترنت، اینترانت و اکسترانت می توان به این موضوع اشاره کرد که همه ی آنها بر اساس ضوابط و معیارهایی مشخصی مبتنی بر مدل TCP/IP بوده و از برنامه های مشترکی استفاده می کنند. همچنین با استفاده از IP می توانند ارتباط و اتصال بین دستگاه های مختلف را ایجاد نمایند.

برای درک بهتر تفاوت ها بهتر است تعریف کلی آن ها را بدانیم. همان طور که در ابتدای این مقاله اشاره شد، اینترنت عمومی ترین شبکه متشکل از میلیون ها کامپیوتر در سرتاسر دنیا می باشد که به هم متصل شده اند. اترنت بخش کوچک تری از اینترنت و شبکه ای محدود به یک مکان جغرافیایی است که دستگاه های موجود در آن مکان را به هم وصل می کند. اینترانت در اصل یک شبکه خصوصی برای یک مکان است که ارتباط آن با خارج قطع بوده و فقط اطلاعاتی که در آن قرار داده شده برای کسانی که از آن استفاده می کنند قابل دسترسی می باشند و اکسترانت نوع بزرگ تری از اینترانت است که برای گسترش آن این امکان وجود دارد تا بعضی از کاربران به صورت محدود و ایمن به اینترنت متصل شوند.

مراجع (References)

[1]. www.en.wikipedia.org

[2]. www.Microchip.com

[3]. www.Ti.com

[4]. www.tech-faq.com

[5]. www.vceit.com

[6]. www.webkaran.com

[7]. www.handymanhowto.com

[8]. www.pluto.ksi.edu

[9]. www.erg.abdn.ac.uk

[10]. www.parsdata.com

[11]. www.finisar.com

[12]. www.automationz.ir

[13]. www.ECA.ir

[14]. www.tebyan-zn.ir

[15]. www.srco.ir

[16]. www.mbaradaran.ir

[17]. www.mehradkit.ir

[18]. www.PinoutsGuide.com

[19]. Designing Embedded Internet Devices -Newnes (2002)

[20]. Practical Ethernet Implementation, N. Ganesan

[21]. Ethernet Implementation on Microcontroller , Yasmeen.S , Nagabhushan Katte and Anitha , Ballari Institute of Technology & Management , Department of Electronics and Communication Engineering

[22]. Ethernet Theory of Operation , AN1120 Application note , Microchip

[23]. GUIDELINES FOR INDUSTRIAL ETHERNET INFRASTRUCTURE

IMPLEMENTATION : A CONTROL ENGINEER’S GUIDE , Carlos Rojas , Peter Morell

[24]. Implementing Fast Ethernet , Adaptec.com

[25]. Transformerless Applications of Microchip’s Ethernet Devices , AN2190 Application note , Microchip

[26]. DP83848 PHYTER Transformerless Ethernet Operation , AN-1519 Application note , TI

[27]. Microchip ENC28J60 Ethernet Controller datasheet

[28]. 11 projects with AVR , سید مهدی حسینی

[29]. Ethernet , دانشگاه آزاد اسلامی واحد شبستر ، سهیلا کاویان