Designing in better connectivity

What’s involved in adding Ethernet functionality to your embedded system design. By Alex Walsh.

Despite the popularity of Bluetooth and Wi-Fi, a wired connection is useful, either as an alternative or, in some cases, as the main form of connectivity in embedded systems. Ethernet is the most popular way to implement a wired connection; the system components are widely available and often integrated into modern microcontrollers.

When considering Ethernet, designers need to make a number of decisions about the software and hardware components. Ethernet comes in many ‘flavours’, differentiated by maximum bit rate, the physical media and the duplex mode. Designers also need to decide how to partition the different components of the Ethernet subsystem: for example, the physical connectors and the required magnetic isolation can be in one package or split into two.

The physical layer device (PHY) and the media access controller (MAC) can be discrete devices or internal to an MCU. Designers will have to make trade offs with each of these options and remember that a higher layer protocol, typically TCP/IP may also be used. Other factors to be borne in mind include data rate and duplex type and there are several ways to implement these functions.

Most embedded MCU designs run at 100Mbit/s. Whilst links of 1Gbit/s or faster are not normally supported, there are sometimes advantages to using a lower data rate – 10Mbit/s, for example.

Figure 1 shows the blocks common to an Ethernet subsystem. The PHY, the lowest layer of the OSI seven layer protocol stack, undertakes auto negotiation of the maximum data rate, whether the link will be full or half duplex and if flow control will be used. It will implement the correct voltage levels for the defined rate and use the correct encoding mechanism. Although Ethernet PHYs are relatively complex mixed signal devices, they tend to be small and low cost; for example, devices in the Microchip LAN874x family come in 4 x 4mm packages at less than $1 in low quantities.

The media independent interface (MII) is a standard interface (IEEE802.3u) between the PHY and the MAC. In theory, you should be able to connect any media to it, including copper and fibre. It is not vendor specific and any MAC with an MII should communicate with any PHY with an MII.

The MII interface is a 4bit bus clocked at 25MHz. However, other versions – such as RMII, SGMI and TBI – run more quickly or use fewer bits. For embedded MCU designs, the most common will be MII or RMII; the latter being a 2bit interface on both edges of a 25MHz clock. There is also an Ethernet management interface called SMI (sometimes called MDC/MDIO, as those are the two signals) which allows set up and configuration of Ethernet devices.

The MAC, the digital function of the interface, constructs the Ethernet frames for despatch and deconstructs the frames when received. Found in Layer 2 – the Data Link Layer – of the OSI seven layer model, the MAC is responsible for Ethernet addressing and for CRC error checking. Many MCUs have an integral MAC, as well as an MII and/or an RMII. An example is

---

Fig 1: A typical MAC and PHY configuration

![Fig 1: A typical MAC and PHY configuration](image1)

Fig 2: Adding another Ethernet port

![Fig 2: Adding another Ethernet port](image2)
Ethernet standards require a transformer on the Ethernet port to provide isolation and remove any DC bias on the signal. It is possible – and sometimes desirable – to connect two PHYs directly without magnetic coupling. Whilst this could be useful if both PHYs are on the same board, this is not recommended. How these system blocks are partitioned across the devices used in a design varies, depending on the trade-offs that designers need to make between size, cost and performance.

Some MCUs will have integral MACs and PHYs, but it is more common for the purely digital MAC block to be integrated with an external PHY. It is also possible to have both MAC and PHY in a single device called an Ethernet controller, such as the Microchip ENC624J600.

The transformer and RJ45 connector can also be in one device called a Mag-Jack, which gives a size advantage, or in two separate packages, which can be cheaper.

There are different functions that need to be achieved in the hardware and these can be split into the MAC function, the PHY function and isolation.

An Ethernet switch – not to be confused with a router – routes Ethernet frames between its ports. It does this by checking the source address of each frame it receives and storing it in a table. In many embedded designs, this is usually the device at the other end of the cable. However, using a switch means you can increase the number of ports while still using the same single MAC processor. The Microchip LAN89303 is a three port switch with two internal PHYs and a port with an MII interface. By connecting the MII port to the processor and the two PHY ports to MagJacks, you can add an extra port to your design. The MII port can also run at 200Mbit/s in Turbo-MII mode. This removes traffic congestion when both external ports are sending data to the MCU.

Most designs that implement Ethernet will be running a higher level protocol – usually TCP/IP, although other protocols are used.

The Microchip TCP/IP stack is a suite of programs that provides services to standard TCP/IP based applications, including an HTTP server or a mail client. The stack, implemented in a higher modular fashion, provides ‘hooks’ for the developer and it is not necessary to know how the stack works.

Each layer of the protocol stack is responsible for a particular function; the PHY, for example, handles electrical transmission of bits across a medium. Each higher layer in the model uses the underlying layers in a somewhat independent fashion. This layered approach is implemented through the use of encapsulation (see fig 3). Starting at the application layer, the web browser generates an HTTP request using an application specific command. This request is passed to the TCP layer, which constructs a TCP packet comprising header and data. The TCP header contains information particular to the TCP protocol, such as packet sequencing information, checksum and source and destination ports.

At the IP protocol level, an IP datagram – comprising an IP header and IP data – holds the TCP packet. The IP header contains information such as the type of service, checksum, protocol type and the source and destination IP addresses. The data field contains the complete TCP packet.

At the data link/physical layer, the IP datagram is transported across the network using IEEE 802.3. A MAC (IEEE802.3) frame consists of a header and a payload. The MAC header contains information about the frame, such as the source and destination address and frame length. The payload field contains the complete IP datagram. Note the various addresses encapsulated within each protocol are different, and, typically, have no fixed relationships.

Ethernet’s performance and features have continued to evolve, due largely to its widespread use in commercial IT. The standard is easily implemented in an embedded system and offers many opportunities to support the system design goals.

**Author profile:**
Alex Walsh is a field applications engineer with Anglia Components.

---

**Fig 3: A data encapsulation example**

Microchip’s 32bit PIC32MX6xx/7xx ranges.

Ethernet’s performance and features have continued to evolve, due largely to its widespread use in commercial IT.