電腦網路 ch2.1

2.1 General Issues

2.1一般問題

Sandwiched between the physical link and the network layer, the link layer

夾在物理鏈路和網絡層之間，即鏈路層

provides control on physical communications and services to upper network layer.

提供對上層網絡層的物理通信和服務的控制。

This layer includes the following major functions.

該層包括以下主要功能。

Framing Control information, in the header, comes along with the data, in the payload, in the bit stream to specify the destination station, indicate the upper-layer protocol, check possible errors and so on. Data are sent and

標頭中的幀控制信息與數據，有效負載中的數據，位流中的數據一起指定目標站，指示上層協議，檢查可能的錯誤等。數據已發送並

processed in units of frames which are “packets or packet data units (PDUs) at the link layer” and usually contains two main parts: control information and the data.

以幀為單位進行處理，這些幀是“鏈路層上的分組或分組數據單元（PDU）”，通常包含兩個主要部分：控制信息和數據。

The link protocols refer to control information during frame processing.

鏈接協議在幀處理期間引用控制信息。

The data part is from the upper layer and encapsulated with the control information into the frame. The link layer service should delimit the bit stream into frames and turn frames into bit stream. The two terms, packets and frames, are usually used interchangeably.

數據部分來自上層，並與控制信息一起封裝到幀中。鏈路層服務應將位流定界為幀，然後將幀轉換為位流。包和幀這兩個術語通常可互換使用。

We refer to the packet data unit in the link layer as frames to be specific. Similarly for payload/data, header/control information, and node/station, data, control information, and station are called more often in the link layer.

我們將鏈路層中的數據包數據單元稱為特定幀。類似地，對於有效負載/數據，報頭/控制信息和節點/站，數據，控制信息和站在鏈路層中被更頻繁地調用。

Addressing We need an address when writing a letter to our friends, and also need a phone number when dialing up to them. Addressing is needed for the same reason in the link layer. The identity of a station is indicated by an address,

尋址在給我們的朋友寫信時，我們需要一個地址，在與他們的朋友撥號時也需要一個電話號碼。出於相同原因，在鏈接層中需要尋址。站的身份由地址指示，

often presented in a numeric form of a certain length.

通常以一定長度的數字形式顯示。

Error control Frames transmitted over physical media are subject to errors,

錯誤控制通過物理介質傳輸的幀會出現錯誤，

which must be detected by the receiver. The receiver may simply drop the frame,

接收者必須檢測到。接收者可以簡單地丟幀，

or inform the transmitter that errors occur for the transmitter to retransmit the data.

或通知發送器發生錯誤，使發送器重新發送數據。

Flow control The transmitter may send at a rate faster than the receiver can afford. In this case, the receiver has to discard the frames, making the transmitter retransmits the dropped frames, but this is inefficient. Flow control provides a method to let the receiver slow down the transmitter.

流量控制發送器的發送速度可能比接收器所能承受的速度快。在這種情況下，接收器必須丟棄幀，從而使發送器重新發送丟失的幀，但這效率很低。流量控制提供了一種讓接收器降低發送器速度的方法。

Medium Access control There must be an arbitration mechanism when multiple nodes want to transmit data over shared media. A good arbitration mechanism must offer fair access to a shared medium and keep the utilization of the shared

介質訪問控制當多個節點要通過共享介質傳輸數據時，必須有一種仲裁機制。良好的仲裁機制必須提供對共享介質的公平訪問權，並保持對共享介質的利用

medium high when many nodes have backlog, i.e. queued data to transmit.

當許多節點積壓時，即要傳輸的排隊數據時，為中等高。

2.1.1 Framing

2.1.1取景

Frame Delimiting Because data are transmitted in raw bit stream in the physical layer,

幀定界由於數據是在物理層的原始位流中傳輸的，

the link layer must tell the beginning and the end of a frame. It must also turn frames into raw bit stream for physical transmission.

鏈接層必須指出幀的開始和結束。它還必須將幀轉換為原始位流以進行物理傳輸。

This function is called framing. Many methods can delimit the frames.

此功能稱為成幀。許多方法可以界定幀。

Depending on the basic unit of a frame,

根據框架的基本單位，

which can be byte (or octet) or bit,

可以是字節（或八位位組）或位，

called a byte-oriented or bit-oriented frame,

稱為面向字節或面向位的幀，

special sentinel characters or bit patterns can mark the frame boundary.

特殊的前哨字符或位模式可以標記幀邊界。

We introduce the framing examples of bit-oriented HDLC frames.

我們介紹了面向比特的HDLC幀的成幀示例。

There are still other ways to delimit frames.

還有其他分隔幀的方法。

For examples,

舉些例子，

some Ethernet systems use special physical encoding to mark frame boundary,

一些以太網系統使用特殊的物理編碼來標記幀邊界，

while others identify the boundary simply by the presence or absence of signal 1 .

其他人僅通過信號1的存在或不存在來識別邊界。

The former method is used since fast Ethernet (i.e.,

由於快速以太網（即，

100 Mb/s) because it can detect the physical link status.

100 Mb / s），因為它可以檢測物理鏈路狀態。

The latter is unable to to do so because it cannot tell whether the physical link is broken or no frames are being transmitted (no signal is on the link in both cases).

後者無法這樣做，因為它無法判斷物理鏈路是否斷開或是否沒有幀正在傳輸（兩種情況下鏈路上都沒有信號）。

It was used in 10 Mb/s Ethernet,

它用於10 Mb / s以太網，

and no longer used in later Ethernet technology.

不再在以後的以太網技術中使用。

A bit-oriented frame can specify a special bit pattern,

面向位的幀可以指定特殊的位模式，

say 01111110 in HDLC,

在HDLC中說01111110，

while a byte-oriented frame can specify special characters,

面向字節的幀可以指定特殊字符，

say SOH (start of header) and STX (start of text) to mark the beginning of frame header and data.

說SOH（標題的開始）和STX（文本的開始）來標記幀標題和數據的開始。

An ambiguity may exist when normal data characters or bits are the same as the special characters or pattern.

當常規數據字符或位與特殊字符或模式相同時，可能會存在歧義。

A technique called byte- or bit-stuffing is used to solve the ambiguity,

一種稱為字節填充或位填充的技術用於解決歧義，

as illustrated in Figure 2.1. A special escape character,

如圖2.1所示。一個特殊的轉義字符，

namely DLE (data link escape),

即DLE（數據鏈接轉義），

precedes a special character to indicate the next character is normal data in a byte-oriented frame. Because DLE itself is also a special character,

在特殊字符之前，表示下一個字符是面向字節的幀中的普通數據。由於DLE本身也是一個特殊字符，

two consecutive DLEs represent a normal DLE character.
兩個連續的DLE代表正常的DLE字符。
For 01111110 used in HDLC,
對於HDLC中使用的01111110，
when five consecutive 1’s are in the normal data bits,
當正常數據位中有五個連續的1時，
a 0 is stuffed,
填充0
so that the pattern 01111110 never appears in normal data.
因此模式01111110永遠不會出現在普通數據中。
Both the transmitter and the receiver follow the same rule to solve the ambiguity.
發送器和接收器都遵循相同的規則來解決歧義。
A different approach is in the Ethernet.
以太網是另一種方法。
For example,
例如，
100BASE-X uses special encoding to mark the boundary because after 4B/5B encoding,
100BASE-X使用特殊的編碼來標記邊界，因為在4B / 5B編碼之後，
32 (= 2的5次方) possible codes can be transmitted over physical media while only 16 out of them come from actual data.
可以在物理媒體上傳輸32個（= 2的5次方）可能的代碼，而其中只有16個來自實際數據。
Other codes can serve as control codes.
其他代碼可以用作控制代碼。
These control codes are uniquely recognizable by the receiver and thus used to delimit a frame out of a sequence of bit stream.
這些控制碼可由接收器唯一識別，因此可用於從比特流序列中定界幀。
Another Ethernet system,
另一個以太網系統，
10BASE-T,
10BASE-T，
recognizes the frame boundary simply according to the presence or absence of a signal.
僅根據信號的存在或不存在來識別幀邊界。

Frame Format
影格格式
A frame is divided into header fields that include various kinds of control information and the data from the network layer,
幀分為多個標頭字段，這些標頭字段包括各種控制信息和來自網絡層的數據，
which again contains control information of higher layers and the actual data.
它再次包含更高層的控制信息和實際數據。
The control information of higher layers is treated as normal data in the link layer.
高層的控制信息在鏈路層中被視為普通數據。
Typical header fields of control information other than the data field are listed below.
下面列出了除數據字段以外的控制信息的典型頭字段。
Address:
地址：
It usually indicates the source or the destination address.
它通常指示源或目標地址。
The receiver knows the frame is for it if the destination address matches its own.
如果目的地址與其自己的地址匹配，則接收方知道該幀適用於該幀。
It also can respond to the source by filling in the destination address of the outgoing frame with the source address of the incoming frame.
它還可以通過用輸入幀的源地址填充輸出幀的目標地址來響應源。

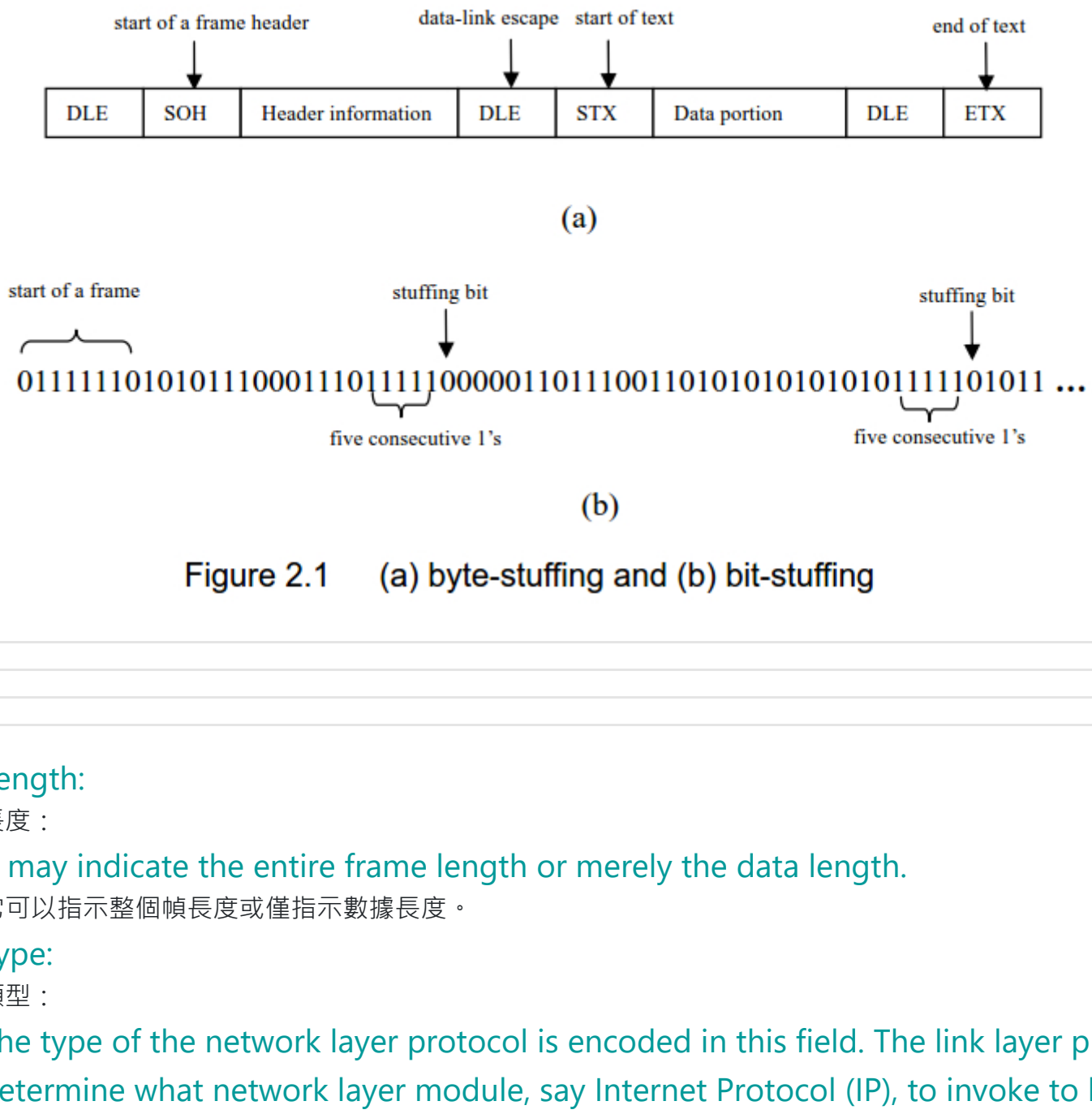

Figure 2.1 (a) byte-stuffing and (b) bit-stuffing

Length:
長度：
It may indicate the entire frame length or merely the data length.
它可以指示整個幀長度或僅指示數據長度。
Type:
類型：
The type of the network layer protocol is encoded in this field. The link layer protocol can read the code to determine what network layer module, say Internet Protocol (IP), to invoke to handle the data field further.
網絡層協議的類型在此字段中編碼。鏈路層協議可以讀取該代碼，以確定要調用哪個網絡層模塊（例如Internet協議（IP））來進一步處理數據字段。
Error detection code:
錯誤檢測代碼：
It is a mathematical function of the content in a frame. The transmitter computes the function and embeds the value in the frame. Upon receiving the frame, the receiver computes in the same way to see if both results match. If they do not match, it implies the content is changed somewhere during transmission.
它是框架中內容的數學函數。發送器計算功能並將值嵌入幀中。接收到幀後，接收器以相同的方式進行計算以查看兩個結果是否匹配。如果它們不匹配，則表示內容在傳輸過程中發生了更改。

2.1.2 Addressing

2.1.2尋址
Global or Local Address
全球或本地地址
An address is an identifier to identify a station from others in communications.
地址是一個標識符，用於在通信中與其他站點進行標識。
Although a name is easier to remember, it is compact to use a numerical address in low-layer protocols. We leave the concept of name as an identifier to Chapter 5 (See Domain Name System). An address can be globally unique or locally unique.
儘管名稱更容易記住，但在低層協議中使用數字地址卻很緊湊。我們將名稱作為第5章的標識符（請參閱域名系統）。地址可以是全局唯一或本地唯一。
A globally-unique address is unique worldwide, while a locally-unique address is only unique in a local site. In general, a locally-unique address consumes fewer bits and requires the administrator's efforts to ensure the local uniqueness. Since the bit overheads of the address are trivial, globally-unique addresses are preferred nowadays. The administrator simply adds a station at will, and does not need to worry about the conflict over addresses.
全球唯一地址在全球範圍內是唯一的，而本地唯一地址僅在本地站點中是唯一的。通常，本地唯一的地址消耗的位更少，並且需要管理員努力確保本地唯一性。由於地址的位開銷很小，因此當今首選全局唯一的地址。管理員只需添加一個工作站即可，而不必擔心地址衝突。
Address Length
地址長度

How long should an address be? A long address takes more bits to be transmitted, and is harder to refer to or remember, but a short address may not be enough for global uniqueness. For a locally-unique address, 8 or 16 bits should be enough. For a globally unique address, much more bits are necessary. A very popular addressing format in IEEE 802 has 48 bits long. We leave it as an exercise for the readers to discuss whether the length is sufficient.

地址應保留多長時間？長地址需要傳輸更多的位，並且更難以引用或記住，但是短地址可能不足以實現全局唯一性。對於本地唯一地址，8或16位應足夠。對於全局唯一地址，需要更多的位。 IEEE 802中一種非常流行的尋址格式具有48位長。我們將其作為練習供讀者討論長度是否足夠。

IEEE 802 MAC Address

IEEE 802 MAC地址

The popular link address specified in the IEEE 802 Standards is an excellent example because the addressing is widely adopted in many link protocols, including Ethernet, Fiber Distribution Data Interface (FDDI), Token Ring, wireless LAN, etc. While the IEEE 802 specifies the use of either 2-byte of 6-byte addresses, most implementations adopt 6-byte (or 48-bit) addresses.

IEEE 802標準中指定的流行鏈接地址就是一個很好的例子，因為尋址已在許多鏈接協議中廣泛採用，包括以太網，光纖分配數據接口（FDDI），令牌環，無線LAN等。IEEE802指定了使用2字節或6字節地址，大多數實現採用6字節（或48位）地址。

To ensure the address is globally unique, the address is partitioned into two main parts:

為確保地址在全局上唯一，該地址分為兩個主要部分：

Organization-Unique Identifier (OUI) and Organization-Assigned Portion, each occupying three bytes.

組織唯一標識符（OUI）和組織分配的部分，每個佔據三個字節。

The IEEE administrates the former. The manufacturers

IEEE管理前者。廠商

can contact the IEEE to apply for an OUI2, and they are in charge of the uniqueness of the Organization-Assigned Portion. In theory, totally 248 (around 1015) addresses can be assigned.

可以聯繫IEEE申請OUI2，並且由他們負責組織分配部分的唯一性。理論上，總共可以分配248（大約1015）個地址。

This number is large enough for global uniqueness. The address is often written in hexadecimal form separated by

此數字足夠大，可以實現全局唯一性。地址通常以十六進制形式寫，中間用

dashes or colons, e.g. 00-32-4f-cc-30-58. Figure 2.2 illustrates the address format.

破折號或冒號，例如00-32-4f-cc-30-58。圖2.2說明了地址格式。

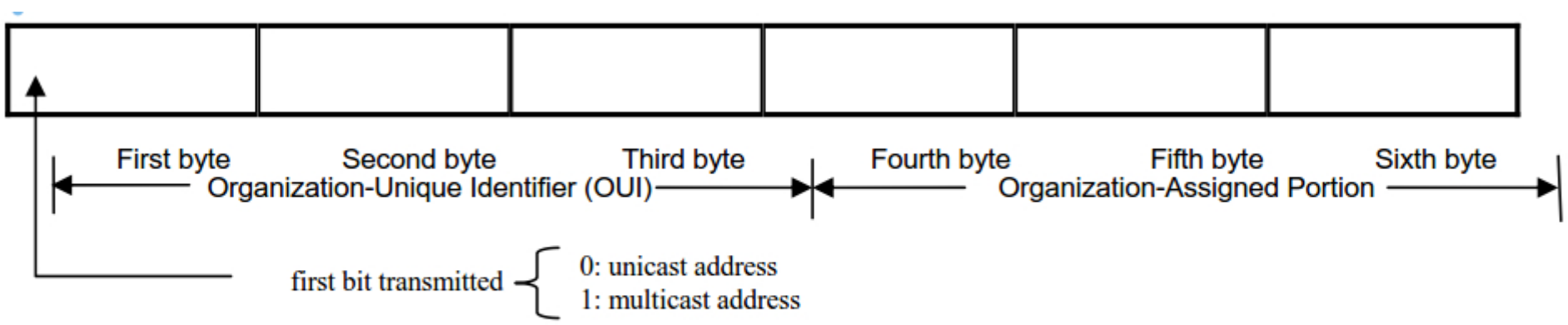

Figure 2.2 IEEE 802 address format

The first bit in transmission order is reserved to indicate whether the address is unicast or multicast3. A unicast address is destined for a single station,

保留傳輸順序的第一位，以指示該地址是單播還是多播3。單播地址以單個電台為目的地，

while a multicast address is destined for a group of stations.

而多播地址則指定給一組站。

A special case of multicast is broadcast,

廣播是一種特殊情況，

where all bits in the address are 1's.

地址中的所有位均為1。

It is destined for all stations as far as a frame can reach in the link layer.

只要幀可以在鏈路層中到達，它就被指定給所有站。

Another interesting thing is that the transmission order of bits in each byte in the address may be different from the order stored in memory.

另一個有趣的事情是地址中每個字節中位的傳輸順序可能與存儲在存儲器中的順序不同。

In Ethernet,

在以太網中，

the transmission order is least significant bit (LSB) first in each byte,

每個字節的傳輸順序是最低有效位（LSB），

called little-endian.

叫做little-endian。

In a byte b7b6...b0,

在字節b7b6...b0中，

for example,

例如，

b0 is transmitted first,

b0首先發送，

then b1,

然後是b1

and so on.

等等。

In other protocols,

在其他協議中

such as FDDI and Token Ring,

例如FDDI和令牌環，

the transmission order is most significant bit (MSB) first in each byte,

傳輸順序是每個字節中的最高有效位（MSB）首先，

called big-endian.

叫做大端。

2.1.3 Error Control

2.1.3錯誤控制

Frames are subject to errors during transmission.

幀在傳輸過程中會出錯。

The errors should be detected.

應該檢測到錯誤。

As mentioned in Subsection 2.1.1,

如2.1.1小節所述，

error detection code is a function of the content of a frame,

檢錯碼是幀內容的函數，

computed to fill into a field by the transmitter,

計算為由發射機填充到一個字段中，

and re-computed for checking at the receiver. We now illustrate two common functions in error detection: checksum and cyclic redundancy check (CRC).

並重新計算以在接收器處進行檢查。現在，我們說明錯誤檢測中的兩個常用功能：校驗和和循環冗餘校驗（CRC）。

Error Detection Code

錯誤檢測碼

The checksum computation simply divides the frame content into blocks of m bits and takes the sum of these blocks. In Open Source Implementation 2.1,

校驗和計算僅將幀內容劃分為m位的塊，然後取這些塊的總和。在開放源代碼實施2.1中，

we will introduce a piece of code that implements the checksum computation.

我們將介紹一段實現校驗和計算的代碼。

Another powerful technique is cyclic redundancy check,

另一項強大的技術是循環冗餘校驗，

which is slightly complicated than checksum,

比校驗和稍微複雜一點

but it is easy to implement in hardware.

但是很容易在硬件中實現。

Suppose m bits are in the frame content.

假設幀內容中有m位。

The transmitter can generate a sequence of k bits as the frame check sequence (FCS) such that the total frame, having m+k bits, can be divided by a predetermined bit pattern, called generator.

發送器可以生成k位序列作為幀檢查序列（FCS），從而可以將具有m + k位的總幀除以預定的位模式（稱為生成器）。

The receiver divides in the same way and sees if the remainder is zero.

接收器以相同的方式除法，看看餘數是否為零。

If the remainder is non-zero,

如果餘數不為零，

there are errors during transmission.

傳輸過程中有錯誤。

The following example shows a trivial CRC procedure to generate the FCS.

以下示例顯示了用於生成FCS的普通CRC過程。

frame content F= 11010001110 (11 bits)
generator B= 101011 (6 bits)
FCS = (5 bits)

The procedure goes like the following steps:

Step 1 Shift F by 2^5, and append it with 5 0's, yielding 1101000111000000.

Step 2 The resulting pattern in Step 1 is divided by B. The process is as follows:
(The computation is all module-2 arithmetic)

```
                11100000111
          ┌─────────────────
101011    ) 1101000111000000
            101011
            ------
             111110
             101011
             ------
              101011
              101011
              ------
                 110000
                 101011
                 ------
                  110110
                  101011
                  ------
                   111010
                   101011
                   ------
                    10001  ←—— the remainder
```

Step 3

第三步

The remainder in the above computation is appended in the original frame content,

上面計算中的其餘部分附加在原始幀內容中，

yielding 1101000111010001. The resulting frame is then transmitted.

產生1101000111010001。然後發送結果幀。

The receiver divides the incoming frame by the same generator to verify the frame.

接收器將傳入幀除以同一生成器以驗證幀。

We leave the verification on the receiver side as an exercise.

作為練習，我們將驗證留在接收方。

Although the preceding description is trivial, the reasoning behind the practical CRC computation is mathematically complex. With careful design of the generator,

儘管前面的描述很簡單，但是實際CRC計算背後的原因在數學上很複雜。經過精心設計的發電機，

the CRC is proved to be able to detect many kinds of errors, including

CRC被證明能夠檢測多種錯誤，包括

1. single-bit error

1.單位錯誤

2. double-bit error

2.雙位錯誤

3. Any burst errors whose length is less than that of the FCS.

3.任何長度小於FCS的突發錯誤。

The CRC computation can be easily implemented in hardware with *exclusive-OR* gates and *shift registers*. Suppose we represent the generator in the form $a_n a_{n-1} a_{n-2} \ldots a_1 a_0$. The bits a_n and a_0 must be 1. We plot a general circuit architecture that implements the CRC computation in Figure 2.3. The frame content is shifted into this circuit bit by bit, and the final bit pattern in the shift registers is the FCS, i.e. $C_{n-1} C_{n-2} \ldots C_1 C_0$. The initial values of $C_{n-1} C_{n-2} \ldots C_1 C_0$ are insignificant because they will be finally shifted out after computation. For very high-speed links, circuits of parallel CRC computation is employed instead to meet the high-speed requirement.

The CRC computation can be easily implemented in hardware with exclusive-OR gates and shift registers.

具有異或門和移位寄存器的硬件可以輕鬆實現CRC計算。

Suppose we represent the generator in the form anan-1an-2…a1a0.

假設我們以anan-1an-2…a1a0的形式表示生成器。

The bits an and a0 must be 1. We plot a general circuit architecture that implements the CRC computation in Figure 2.3. The frame content is shifted into this circuit bit by bit,

an和a0位必須為1。我們在圖2.3中繪製了實現CRC計算的通用電路架構。幀內容一點一點地移入該電路，

and the final bit pattern in the shift registers is the FCS,

移位寄存器中的最後一個位模式是FCS，

i.e. Cn-1Cn-2…C1C0.

即Cn-1Cn-2…C1C0。

The initial values of Cn-1Cn-2…C1C0 are insignificant because they will be finally shifted out after computation.

Cn-1Cn-2…C1C0的初始值無關緊要，因為它們將在計算後最終移出。

For very high-speed links,

對於超高速鏈接，

circuits of parallel CRC computation is employed instead to meet the high-speed requirement.

取而代之的是使用並行CRC計算電路來滿足高速需求。

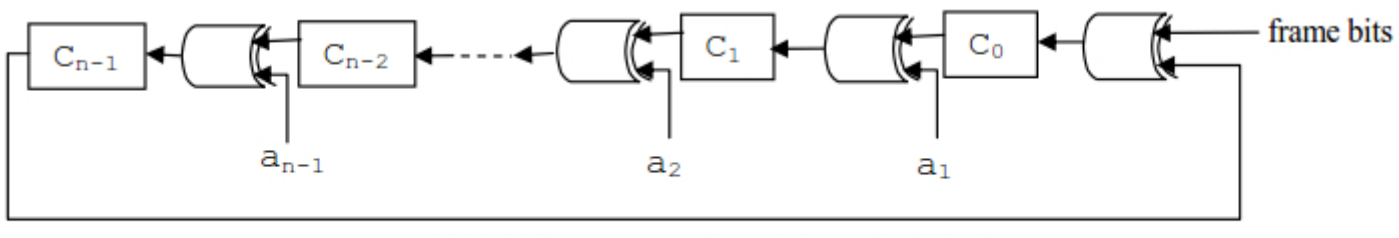

Figure 2.3 CRC circuit diagram

But how does the receiver respond to an error frame? The receiver can respond in the following ways:

但是接收器如何響應錯誤幀？接收方可以通過以下方式進行響應：

1. Silently discard when the incoming frame is incorrect

1.當輸入幀不正確時，靜默丟棄

2. Positive acknowledgement when the incoming frame is correct

2.傳入幀正確時的肯定確認

3. Negative acknowledgement when the incoming frame is incorrect

3.傳入幀不正確時的否定確認

The transmitter may retransmit the frame that is incorrectly received or simply ignore the errors. In the latter case, higher layer protocols, say TCP, could handle the retransmission.

發送器可以重新發送錯誤接收的幀，或者簡單地忽略錯誤。在後一種情況下，高層協議（例如TCP）可以處理重傳。

Open Source Implementation 2.1: Checksum

開源實施2.1：校驗和

Checksum computation is common in Internet protocols, such as IP, UDP and TCP. The adjacent octets are first paired to form 16-bit integers, and then the 1's complete sum of these pairs is computed.

校驗和計算在Internet協議（例如IP，UDP和TCP）中很常見。首先將相鄰的八位字節配對以形成16位整數，然後計算這些對的1的完整和。

If there is a byte left in the pairs, it is added into the checksum. Finally, the 1's complement of the result is filled into the checksum field. The receiver follows the same procedure to compute over the same octets and adds the checksum field.

如果對中還剩下一個字節，則將其添加到校驗和中。最後，將結果的1的補碼填充到校驗和字段中。接收器遵循相同的過程來計算相同的八位位組，並添加校驗和字段。

If the result is all 1's, the check succeeds. Because the Linux implementation of checksum is usually written in

如果結果全為1，則檢查成功。因為Linux的校驗和實現通常是寫在

assembly languages for efficiency, so we present the C code in RFC 1071 for better readability. Open Source Implementation 3.2 in Chapter 3 explains the assembly version in the Linux kernel for IP checksum.

彙編語言以提高效率，因此我們在RFC 1071中提供C代碼以提高可讀性。第3章中的開源實現3.2解釋了Linux內核中用於IP校驗和的程序集版本。

```
 /* Compute Internet Checksum for "count" bytes
  *         beginning at location "addr".
  */
register long sum = 0;

 while( count > 1 )  {
        sum += * (unsigned short) addr++;
        count -= 2;
}
/*  Add left-over byte, if any */
if( count > 0 )
        sum += * (unsigned char *) addr;
/*  Fold 32-bit sum to 16 bits */
while (sum>>16)
    sum = (sum & 0xffff) + (sum >> 16);
checksum = ~sum;
```

Open Source Implementation 2.2: CRC32

開源實施2.2：CRC32

CRC-32 is a common for many MAC protocols, such as Ethernet and 802.11 Wireless LAN.

CRC-32在許多MAC協議（例如以太網和802.11無線LAN）中很常見。

An open-source implementation can be found in the Ethernet MAC project on the OpenCores Web site (http://www.opencores.org). (See the Verilog implementation eth_crc.v in the CVS repository of the project.) In this

可以在OpenCores網站（http://www.opencores.org）的以太網MAC項目中找到一種開源實現。（請參閱項目CVS存儲庫中的Verilog實現eth_crc.v。）

implementation, 4 bits in the data comes into the CRC module in batch sequentially.

實施中，數據中的4位按順序批量進入CRC模塊。

The CRC value is arbitrarily assigned all 1's in the beginning.

CRC值在開始時被任意分配為全1。

Each bit of the CRC value in the next stage comes from xor'ing the selected bits in the incoming 4 bits and those of the CRC value in the last stage.

下一階段的CRC值的每一位來自於對輸入的4位中的所選位與最後一階段的CRC值的那些位進行異或。

Because of their verbosity, we leave the details in the equations of each bit to eth_crc.v. After the data bits finish, the final CRC value is derived at the same time.

由於它們的冗長性，我們將細節保留在eth_crc.v的每一位方程式中。數據位完成後，將同時導出最終的CRC值。

The receiver follows the same procedure to derive the CRC value and check the correctness of the incoming frames.

接收器遵循相同的過程來得出CRC值並檢查輸入幀的正確性。

Sidebar – Principle in Action: CRC or Checksum?

補充工具欄–實施原則：CRC還是校驗和？

Checksum is used in higher-layer protocols such as TCP, UDP and IP, while CRC is found in Ethernet and Wireless LAN. The distinction has two reasons.

校驗和用於更高層的協議中，例如TCP，UDP和IP，而CRC在以太網和無線LAN中發現。區別有兩個原因。

First, CRC is easily implemented in hardware, but not in software.

首先，CRC易於在硬件中實現，而不能在軟件中實現。

Because higher-layer protocols are almost implemented in software, using checksum for them is a natural choice. (Implementing these protocols in ASIC was still costly when they were born.) Second, CRC is mathematically proven to be robust to a number of errors in physical transmission.

由於高層協議幾乎是在軟件中實現的，因此對它們使用校驗和是很自然的選擇。（當這些協議誕生時，在ASIC中實施這些協議仍然很昂貴。）其次，CRC在數學上已被證明對物理傳輸中的許多錯誤具有魯棒性。

Since it has filtered out most transmission errors, using checksum to double-check unusual errors (e.g., those happen within a network device) should be sufficient in practice.

由於它已濾除了大多數傳輸錯誤，因此在實踐中使用校驗和仔細檢查異常錯誤（例如，發生在網絡設備中的錯誤）就足夠了。

2.1.4 Flow Control

2.1.4流量控制

Flow control addresses the problem with a fast transmitter and a slow receiver.

流控制解決了快速發送器和慢速接收器的問題。

It provides a method to let an overwhelmed receiver tell the transmitter,

它提供了一種讓不知所措的接收者告訴發送者的方法，

"Hey! You transmit too fast. Please wait!" The simplest method is stop-and-wait.

"嘿！您傳輸太快。請稍候！"最簡單的方法是停止並等待。

The transmitter transmits one frame, waits the acknowledgement from the receiver, and transmits the next.

發送器發送一個幀，等待來自接收器的確認，然後發送下一幀。

This method results in very low utilization of the transmission link. Better methods are introduced as follows.

這種方法導致傳輸鏈路的利用率很低。更好的方法介紹如下。

Sliding Window Protocol

滑動窗口協議

An improvement is the sliding window protocol. The transmitter can transmit a window of frames without cknowledgements. When the acknowledgements return from the receiver,

一種改進是滑動窗口協議。發送器可以發送沒有確認的幀窗口。當確認從接收方返回時，

the transmitter can move forward to transmit more frames.

發射器可以向前移動以發射更多幀。

To track which outgoing frame corresponds to a returned acknowledgement, each frame is labeled with a sequence number. The range of sequence number should be large enough to prevent a sequence number

from reappearing soon; otherwise, ambiguity will happen, since we have no means to tell whether the sequence number represents an old or a new frame.
為了跟踪哪個傳出幀對應於返回的確認，每個幀都標有序列號。序列號的範圍應足夠大，以防止序列號很快出現。否則，將產生歧義，因為我們無法分辨序列號是舊幀還是新幀。
For example, suppose the window size of the transmitter is 9, meaning the transmitter can transmit up to 9 frames, say frame no. 1 to no. 9, without acknowledgements. Suppose the transmitter has transmitted 4 frames and received an acknowledgement that indicates the first three frames are successfully received.
例如，假設發送器的窗口大小為9，這意味著發送器最多可以發送9幀，即幀號。 1至否。 9，恕不另行通知。假設發送器已經發送了4幀並接收到一個確認，該確認表明前三個幀已成功接收。
The window will slide 3 more frames, meaning 8 more frames (i.e., frame no. 4 to no. 12) can be transmitted without acknowledgements.
該窗口將再滑動3幀，這意味著可以在沒有確認的情況下再發送8幀（即第4到第12幀）。
The window that contains the frames from frame no. 1 to no. 9 now contains those from frame no. 4 to no. 12. In other words, the window slides along the sequence of frames. Sliding window flow control is also a very important technique in Transmission Control Protocol (TCP), which is an excellent and most practical example that adopts sliding window. We will introduce its application in TCP in Chapter 4.
包含框架號中的框架的窗口。 1至否。現在9包含第9幀中的內容。 4到無。 12.換句話說，窗口沿著幀序列滑動。滑動窗口流控制也是傳輸控制協議（TCP）中的一項非常重要的技術，它是採用滑動窗口的出色且最實用的示例。我們將在第4章中介紹其在TCP中的應用。

Other Approaches
其他方法
There are still other methods to implement flow control. For example, the mechanisms in Ethernet include back pressure and PAUSE frame. However,
還有其他方法可以實現流量控制。例如，以太網中的機制包括背壓和暫停幀。然而，
to understand these methods require the knowledge of how these protocols operate.
要了解這些方法，需要了解這些協議的工作方式。
We leave these flow control techniques to later sections.
我們將這些流控制技術留給後面的部分。

2.1.5 Medium Access Control
2.1.5媒體訪問控制
Medium access control, also simply referred to as MAC, is needed when multiple stations share a common physical medium.
當多個站共享同一物理介質時，需要進行介質訪問控制，也簡稱為MAC。
It includes an arbitration mechanism that every station should obey in order to share fairly and efficiently.
它包括每個站都應遵循的仲裁機制，以便公平有效地共享。
We summarize the techniques into three categories below.
我們將技術概括為以下三類。

Contention-based Approach
基於競爭的方法
Multiple stations contend for the use of medium in this approach.
在這種方法中，多個站點爭用介質。
A classical example is ALOHA,
一個經典的例子是ALOHA，
in which stations transmit data at will. If two or more stations transmit at the same time, called a collision, their frames will be garbled,
在哪個站點隨意發送數據。如果兩個或兩個以上的電台同時發送信號（稱為衝突），則它們的幀會出現亂碼，
making the throughput low.
降低吞吐量。

A refinement is the slotted ALOHA, in which a station is allowed to transmit only in the beginning of a time slot.

一種改進是帶時隙的ALOHA，其中僅允許站在時隙的開始進行發送。

Further refinements include Carrier Sense and Collision Detection.

進一步的改進包括載波偵聽和碰撞檢測。

Carrier sense means the station senses if there is transmission (in signal called carrier) over the shared medium.

載波偵聽意味著站點偵聽共享媒體上是否存在傳輸（在稱為載波的信號中）。

The transmitter will wait politely until the shared medium is free. Collision detection shortens the garbled bit stream by stopping transmission once a collision is detected.

發射機將禮貌地等待，直到共享媒體可用為止。一旦檢測到衝突，衝突檢測就會通過停止傳輸來縮短亂碼。

Contention-free Approach Contention-based approach is inefficient if a collision cannot be detected in time. A complete frame might have been garbled before the transmitter is aware of the tragedy.

無爭用方法如果無法及時檢測到衝突，則基於爭用的方法效率不高。在發射機意識到悲劇之前，一個完整的幀可能已經出現亂碼。

There are two common contention-free approaches: round-robin and reservation-based.

有兩種常見的無競爭方法：輪詢和基於保留。

In the former,

在前者中

a token is circulated among stations one by one to allow fair share of the medium.

令牌在站點之間一個接一個地循環，以公平分配媒介。

A station that owns the token has the right to transmit its frame. The most typical examples are Token Ring,

擁有令牌的站有權發送其幀。最典型的例子是令牌環

Token Bus and FDDI.

令牌總線和FDDI。

Their mechanisms are similar despite different structures.

儘管結構不同，但它們的機制相似。

The latter somehow reserves the channel before the transmitter actually transmits the frame. A well-known example is the RTS/CTS mechanism in IEEE 802.11 WLAN.

後者以某種方式在發送器實際發送幀之前保留了信道。一個眾所周知的示例是IEEE 802.11 WLAN中的RTS / CTS機制。

We shall talk more about this mechanism in Section 2.4. Using reservation is a tradeoff because the process itself is an overhead.

我們將在2.4節中進一步討論該機制。使用保留是一種折衷，因為流程本身是開銷。

If the cost of a frame loss is insignificant,

如果丟幀的成本微不足道，

e.g. a short frame,

例如一小幀

a contention-based approach may be better.

基於競爭的方法可能更好。

If only two stations are on a point-to-point link,

如果點對點鏈路上只有兩個站，

the access control might not be necessary,

可能不需要訪問控制，

if it is a full-duplex link.

如果是全雙工鏈接。

We shall talk further about full-duplex operation in Section 2.3.

我們將在第2.3節中進一步討論全雙工操作。

2.1.6 Bridging

2.1.6橋接

Connecting separate LANs into an interconnected network can extend the coverage. An interconnection device operating in the link layer is called a MAC bridge,

將單獨的LAN連接到互連的網絡可以擴展覆蓋範圍。在鏈路層中運行的互連設備稱為MAC橋，

or simply bridge. Bridging interconnects LANs as if they were in the same LAN,
或只是橋樑。橋接互連局域網，就像它們在同一個局域網中一樣，
and has been standardized in the IEEE 802.1D Standard.
並已在IEEE 802.1D標準中進行了標準化。
The bridge knows whether it should forward an incoming frame,
網橋知道是否應轉發傳入的幀，
and to which interface port it should forward.
以及應該轉發到哪個接口端口。
For plug-and-play operation and easy administration,
為了實現即插即用的操作和易於管理，
which port a destination host belongs to should be automatically learned.
應自動了解目標主機屬於哪個端口。
As the topology of a bridged network gets larger,
隨著橋接網絡的拓撲變大，
network administrators may inadvertently create a loop in the topology.
網絡管理員可能會無意中在拓撲中創建一個循環。
IEEE 802.1D stipulates a Spanning Tree Protocol (STP) to eliminate loops in a bridged network.
IEEE 802.1D規定了生成樹協議（STP），以消除橋接網絡中的環路。
There are also issues such as logically separating LANs,
還有一些問題，例如邏輯上分隔局域網，
combining multiple links into a trunk for higher transmission rate,
將多個鏈路合併到一個主幹中以提高傳輸速率，
and specifying the priority of a frame. We shall introduce the details in Section 2.6.
並指定幀的優先級。我們將在2.6節中介紹細節。

2.1.7 Packet Flows
2.1.7數據包流
The link layer has the physical link below it and the network layer above it.
鏈路層在其下具有物理鏈路，在其上具有網絡層。
During packet transmission,
在數據包傳輸期間，
it receives a packet from the network layer,
它從網絡層接收到一個數據包，
encapsulates the packet with appropriate link information such as MAC addresses in the frame header and the frame check sequence in the tail,
使用適當的鏈接信息（例如，幀頭中的MAC地址和尾部中的幀檢查序列）封裝數據包，
and transmits the frame through the physical link.
並通過物理鏈路傳輸幀。
Upon receiving a packet from the physical link,
收到來自物理鏈路的數據包後，
the link layer extracts the header information,
鏈接層提取標題信息，
verifies the frame check sequence,
驗證幀檢查序列，
and passes the payload to the network layer according to the protocol information in the header.
並根據報頭中的協議信息將有效載荷傳遞給網絡層。
But what are the actual packet flows between these layers?
但是這些層之間實際的數據包流是什麼？
Continued from a packet's life in Section 1.5,
繼續第1.5節中小包的規定，
we illustrate the packet flows for both frame reception and transmission by Open Source Implementation 2.3.
我們通過開放源代碼實現2.3說明了幀接收和傳輸的包流。

Open Source Implementation 2.3: Packet Flows in Call Graphs

開源實施2.3：調用圖中的數據包流

The packet flow of the link layer includes the following two paths.

鏈路層的分組流包括以下兩個路徑。

In the reception path,

在接收路徑中

a frame is received from the physical link and then passed to the network layer.

從物理鏈路接收幀，然後將其傳遞到網絡層。

In the transmission path,

在傳輸路徑中

a frame is received from the network layer and then passed to the physical link.

從網絡層接收幀，然後將其傳遞到物理鏈路。

Packet flow in the reception path

接收路徑中的數據包流

Strictly speaking,

嚴格來講，

part of the interface between the link layer and the physical link is located in hardware. The interface for Ethernet,

鏈路層和物理鏈路之間的接口的一部分位於硬件中。以太網接口，

for example,

例如，

will be introduced in Open Source Implementation 2.5. We here introduce the code in the device driver to emphasize the software part to transmit or receive frames.

將在開放源代碼實施2.5中引入。我們在此介紹設備驅動程序中的代碼，以強調軟件部分來發送或接收幀。

When the network interface receives a frame,

當網絡接口收到幀時，

an interrupt is generated to signal the CPU to deal with the frame. The interrupt handler allocates the sk_buff structure with the dev_alloc_skb() function and copies the frame into the structure. The handler then initializes some fields in sk_buff,

產生一個中斷來通知CPU處理該幀。中斷處理程序使用dev_alloc_skb（）函數分配sk_buff結構，並將幀複製到該結構中。然後，處理程序會初始化sk_buff中的某些字段，

particularly the protocol field for use by the upper layer,

特別是上層使用的協議字段，

and notifies the kernel about the frame arrival for further processing. Two mechanisms can perform the notification: (1) the old function netif_rx() (2) the new API net_rx_action() for handling ingress frames since kernel version 2.6. The former is pure interrupt-driven,

並將幀到達通知內核，以進行進一步處理。有兩種機制可以執行通知：（1）舊功能netif_rx（）（2）新API net_rx_action（）自內核版本2.6起用於處理入口幀。前者是純粹的中斷驅動，

while the latter uses a mix of interrupts and polling and is more efficient.

而後者將中斷和輪詢混合使用，效率更高。

For example,

例如，

when the kernel is handling a frame and another new frame has arrived,

當內核正在處理一個幀並且另一個新的幀到達時，

the kernel can keep handling this frame and frames in the ingress queue without being interrupted until the queue is empty.

內核可以繼續處理該幀以及進入隊列中的幀，而不會被中斷直到隊列為空。

Generally,

通常，

the CPU load is lower with the new API at high traffic loads according to some benchmark results.

根據一些基準測試結果，在高流量負載下使用新API時，CPU負載較低。

Therefore,

因此，

we focus on the new API herein.

我們在此重點介紹新的API。

When the kernel is interrupted by new frame arrival,

當內核因新幀到達而中斷時，

which may involve one or more frames depending on the driver design,

根據驅動程序的設計，可能涉及一個或多個框架，

it calls the net_rx_action() function to poll a list of interfaces from a software interrupt NET_RX_SOFTIRQ,
它調用net_rx_action（）函數從軟件中斷NET_RX_SOFTIRQ中輪詢接口列表，
The software interrupt is a bottom-half handler,
軟件中斷是下半部處理程序，
which can be executed in the background to avoid occupying the CPU too long for processing the frame arrival.
可以在後台執行該操作，以避免佔用CPU太長的時間來處理幀到達。
The polling is executed in a round-robin fashion with a maximum number of frames that are allowed to be processed.
輪詢以循環方式執行，並具有允許處理的最大幀數。
The net_rx_action() function invokes the poll() virtual function (a generic function which will in turn call the specific polling function on a device) on each device to dequeue from the ingress queue. If an interface is unable to clear out its ingress queue due to the limitation of the number of frames or the available execution time of net_rx_action(),
net_rx_action（）函數在每個設備上調用poll（）虛擬函數（一個泛型函數，它將依次調用設備上的特定輪詢函數）以從入口隊列中退出。如果接口由於幀數或net_rx_action（）的可用執行時間的限製而無法清除其入口隊列，
it must wait until the next poll.
它必須等到下一次民意調查。
The poll() virtual function in turn calls netif_receive_skb() to process the frame. When net_rx_action() is invoked,
poll（）虛擬函數依次調用netif_receive_skb（）來處理幀。調用net_rx_action（）時，
the L3 protocol type has already been in the protocol field of sk_buff (set by the interrupt handler).
L3協議類型已經在sk_buff的協議字段中（由中斷處理程序設置）。
Therefore,
因此，
netif_receive_skb() knows the L3 protocol type and can copy the frame to the L3 protocol handler associated with the protocol field.
netif_receive_skb（）知道L3協議類型，並且可以將幀複製到與協議字段關聯的L3協議處理程序。
Up to now,
到現在，
the reception is complete,
接待結束，
and the L3 protocol handler takes over the frame and decides what to do next.
然後L3協議處理程序接管幀並決定下一步要做什麼。
Common L3 protocol handlers are ip_rcv,
常見的L3協議處理程序為ip_rcv，
ip_ipv6_rcv and arp_rcv,
ip_ipv6_rcv和arp_rcv，
which handle IPv4,
處理IPv4，
IPv6 and ARP,
IPv6和ARP，
respectively.
分別。

Packet flow in the transmission path
傳輸路徑中的數據包流
The transmission path is symmetric to the reception path.
傳輸路徑與接收路徑對稱。
The function net_tx_action() is the counterpart of net_rx_action(),
函數net_tx_action（）與net_rx_action（）對應，
and it is called when some device is ready to transmit a frame from the software interrupt NET_TX_SOFTIRQ.
當某個設備準備好從軟件中斷NET_TX_SOFTIRQ發送幀時調用該方法。
Like net_rx_action() from NET_RX_SOFTIRQ,
就像NET_RX_SOFTIRQ的net_rx_action（）一樣，

the bottom-half handler net_tx_action() can manage tasks that takes time,

下半部處理程序net_tx_action（ ）可以管理耗時的任務，

such as releasing the buffer space after a frame has been transmitted.

例如在發送幀後釋放緩衝區空間。

The net_tx_action() performs two tasks: (1) ensuring the frames waiting to be sent are really sent by the dev_queue_xmit() function,

net_tx_action（ ）執行兩項任務：（1）確保等待發送的幀確實由dev_queue_xmit（ ）函數發送，

and (2) de-allocating the sk_buff structure after the transmission is completed.

（2）在傳輸完成後，取消分配sk_buff結構。

The frames in the egress queue may be scheduled for transmission with queuing disciplines.

可以調度出口隊列中的幀以與排隊規則進行傳輸。

The qdisc_run() function selects the next frame to transmit,

qdisc_run（ ）函數選擇要傳輸的下一幀，

and called the dequeue() virtual function of the associated queuing discipline. Figure 2.4 summarizes the interfaces above and below the link layer.

並稱為相關排隊規則的dequeue（ ）虛擬函數。圖2.4總結了鏈路層上方和下方的接口。

For the hardware interfaces between the MAC and the PHY,

對於MAC和PHY之間的硬件接口，

please refer to Open Source Implementation 2.5: CSMA/CD for a typical Ethernet example.

有關典型的以太網示例，請參閱開放源代碼實施2.5：CSMA / CD。

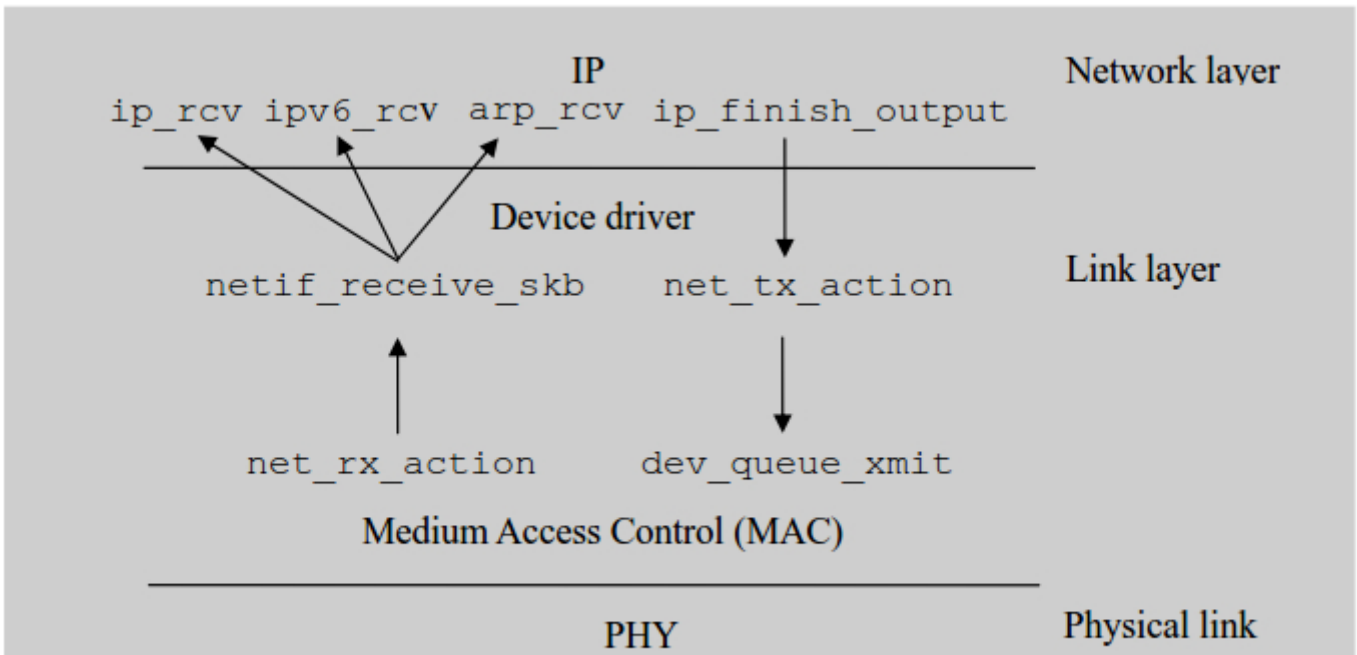

Figure 2.4 The interfaces above and below the data-link layer.