Data Link Layer

Introduction

Geoff Merrett

ELEC3227/ELEC6255: Networks

See Tanenbaum Chapter 3 (Data Link Layer)

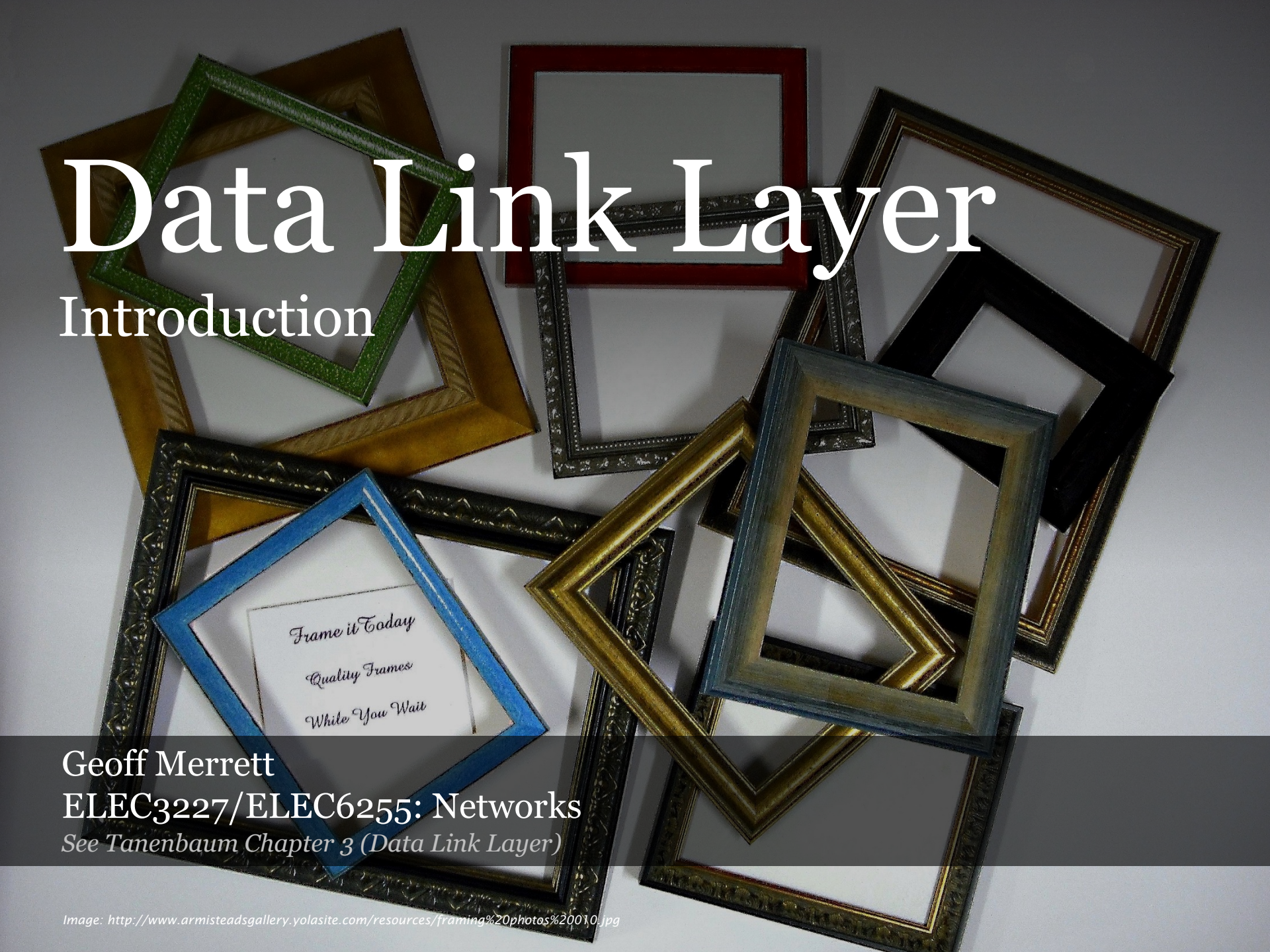
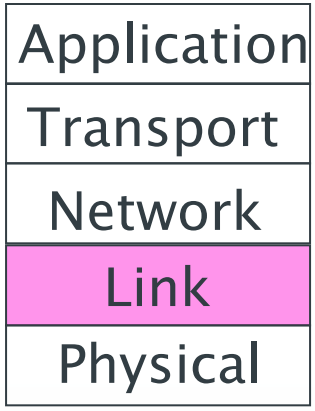


Image: http://www.armisteadsgallery.yolasite.com/resources/framing%20photos%20010.jpg

Outline

- Introduction to the Data Link Layer
  - Purpose
  - Service Provision
- Approaches to Framing
  - Introduction
  - Methods
    - Byte counting
    - Flag bytes with byte stuffing
    - Flag bits with bit stuffing
    - *Physical layer coding violations*



Data Link Layer - Purpose

To provide reliable, efficient communication of *whole units* of information between two *adjacent* machines.

- *whole units* = frames, rather than the bitts of the physical layer or the packets of the network layer.
- *adjacent* = connected by a link that acts like a wire, i.e. the bits are received in the same order as they are sent.

Data Link Layer - Challenges

- What might the challenges be in achieving this?

Data Link Layer - Functions

- Providing a well-defined service interface to the network layer
- Framing: encapsulates packets into frames for transmission
- Error control: dealing with transmission errors
- Flow control: regulating the flow of data

- The data link layer is often considered to have two sublayers:
  - The Logical Link Control sublayer (*the above functions*)
  - The Media Access Control (MAC) sublayer (*we'll come back to later on*)

Data Link Layer – Service Provision

- Unacknowledged connectionless
  - Independent frames are sent to destination without acknowledgement
  - No logical connection established/released
  - If a frame is lost due to errors, Link Layer does not attempt to detect or recover it
  - Suited to situations with
    - Low error rates (so overhead of leaving error control to higher layers is not significant)
    - Real-time traffic where latency is more important that errors
  - e.g. Ethernet

Data Link Layer – Service Provision

- Acknowledged connectionless
  - Independent frames are sent to destination and acknowledged
  - No logical connection established/released
  - Suited to unreliable channels
  - e.g. 802.11 (WiFi)

- Acknowledgements at the Data Link Layer
  - Never essential (but may be more efficient)
  - Could do at the Network Layer, or at higher layers
  - *Why do it where?*

Data Link Layer – Service Provision

- Acknowledged connection-oriented
  - Guaranteed receipt (once and only once), in the correct order, of all frames sent
  - Source and destination establish a connection before data transfer begins
    - Make Connection: initialisation of variables and counters, and handshaking
    - Data Transfer: communicate frames
    - Release Connection: free up resources
  - Suited to situations with
    - Long unreliable links (e.g. satellite)

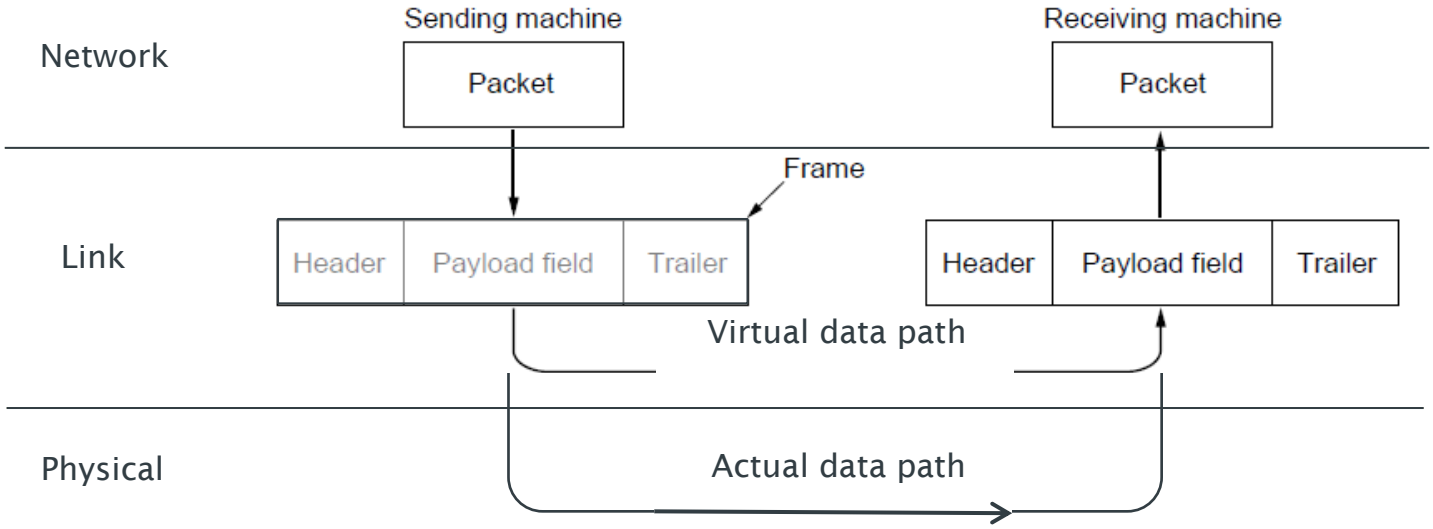
Electronics and
Computer Science

UNIVERSITY OF
Southampton

Framing

Frames

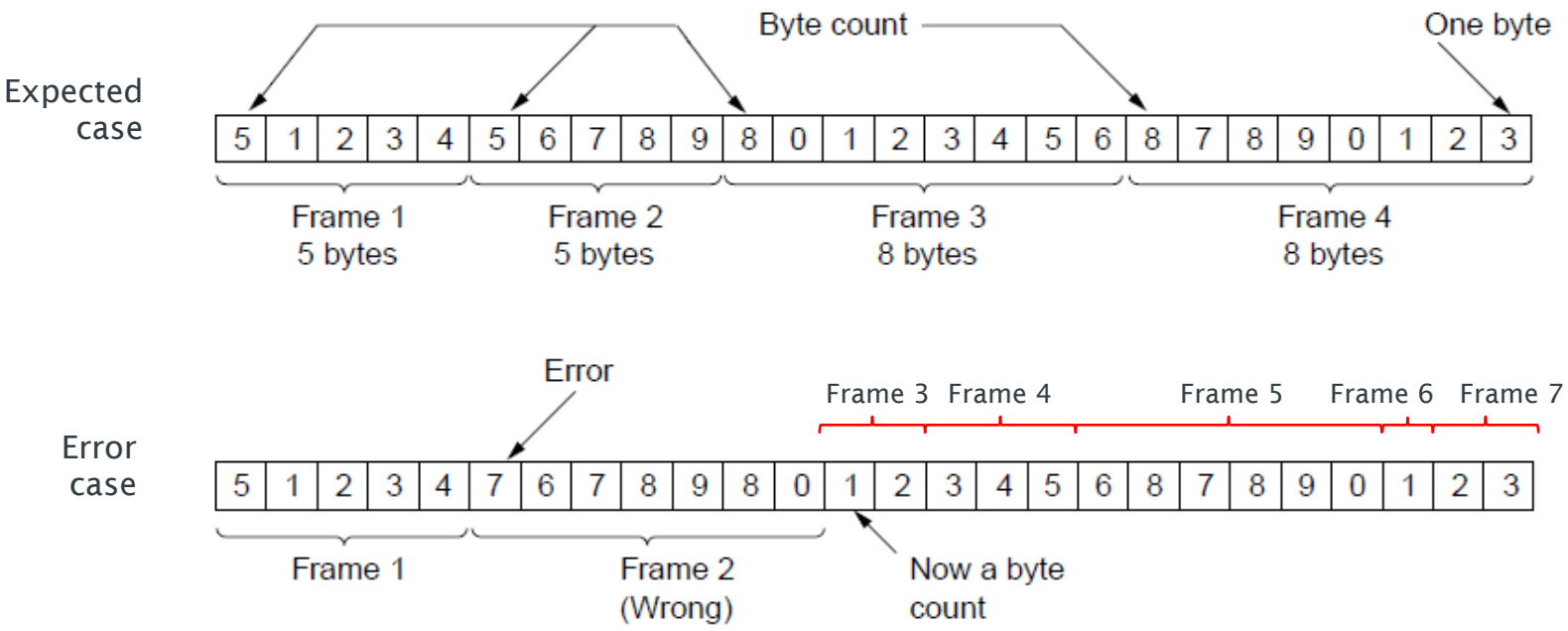
- Link layer accepts a packet from the network layer
- Encapsulates it into a frame (or possibly multiple frames)
- Frames sent using the physical layer
- Reception is the opposite process



- How do we break up a packet into frames?

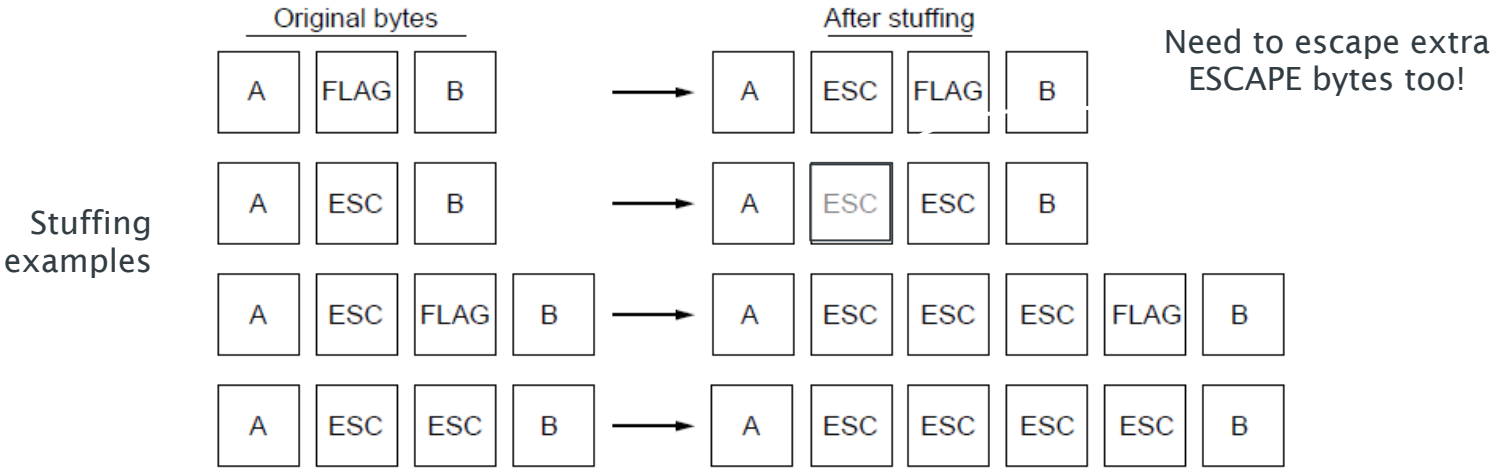
Framing – Byte count

- Frame begins with a count of the number of bytes in it
  - Simple, but difficult to resynchronize after an error



Framing – Byte stuffing

- Special flag bytes delimit frames
- Occurrences of flags in the data must be stuffed (escaped)
  - Longer, but easy to resynchronize after error



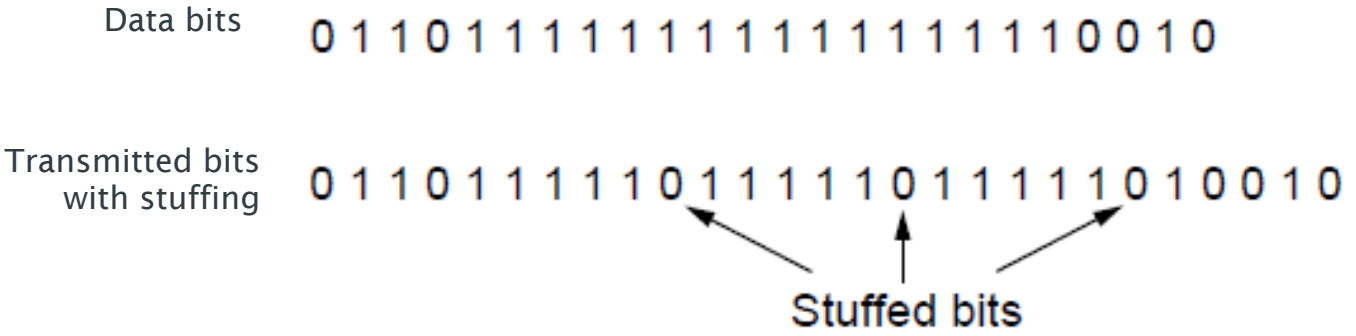
- Used in PPP (Point-to-Point protocol)

Framing – Byte stuffing

- What happens if an transmission error occurs (which effects the flags/escapes?)

Framing – Bit stuffing

- Stuffing done at the bit level:
  - Bit, rather than byte, means frames can have a non-integer number of bytes.
  - Frame flag byte is 01111110 (six consecutive 1s – not shown in the figure)
  - On transmit, after five 1s in the data, a 0 is added (stuffing)
  - On receive, a 0 after five 1s is deleted (de-stuffing)



- Side effect of stuffing (in both byte- and bit-stuffing) is that the size of the frame depends on the data being sent!
- Bit stuffing also used to ensure minimum number of transitions which aids synchronization (USB uses it for this).

Example Questions

ELEC3222 18/19 Exam Question

Byte/nibble stuffing (encoding)

A 16-bit packet, 1101 1111 1010 0110_2, is passed from the network layer to the data link layer for transmission.

1. The data link layer first applies interleaved even parity, with a width of 4 bits. **Calculate** the bit string produced.

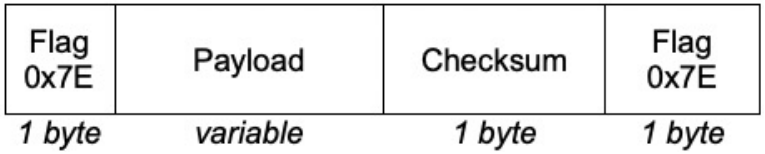
2. **State** how many errors this can detect and correct. **Compare** this in terms of error detection/correction and overhead when compared to non-interleaved even parity (i.e. a parity bit after every 4 bits)?

3. The data link layer then frames the data using nibble (4-bit) stuffing. The flag and escape nibbles are 0110_2 and 1111_2 respectively. Using your answer from (1), **calculate** the bit string produced. If you were unable to answer (1), assume the bit string was unchanged.

ELEC3222 17/18 Exam Question

Bit stuffing (decoding)

A Data Link Layer uses the frame structure below:



If the flag appears elsewhere in the frame, bit stuffing is used. The checksum is the modulo 256 sum of all bytes in the payload. The Physical Layer has received some data, and passes the following bit string to the Data Link Layer:

$$0111\ 1110\ 1100\ 1010\ 1111\ 10110\ 1100\ 1000\ 0111\ 1110_2$$

Determine whether or not any errors have been detected in the frame, and calculate the payload passed to the Network Layer (i.e. the packet contents).

Note: data has been spaced into 4bit groups, and the stuffed bit coloured in red, for the purposes of the lecture slides only – this was not the case in the exam paper!

ELEC3222 15/16 Exam Question

Bit stuffing (encoding)

A bit string, $1101\ 1111\ 101_2$, is passed from the network layer to the data link layer for transmission.

(a) ...

(b) Next, the data link layer frames the data using bit stuffing (the flag byte is 01111110_2). Using your answer from (a), calculate the bit string that is produced. If you were unable to answer (a), assume that the original bit string was unchanged by Hamming coding.

(c) ...

(d) ...

(e) ...

Questions?