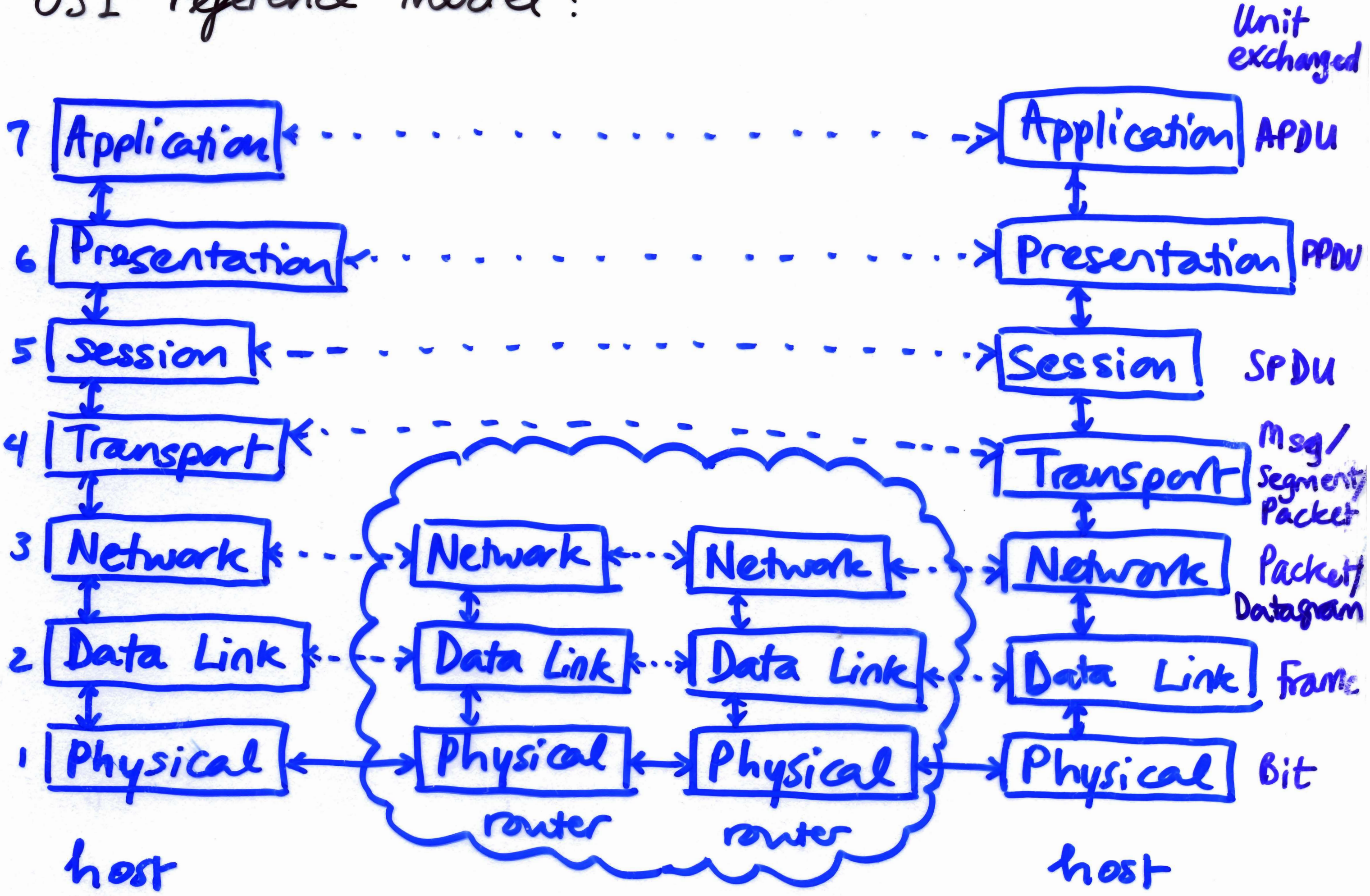
Open Systems Interconnect (OSI) Architecture

OSI reference model:



1. Physical layer: transmission of raw bits
   - concerned with mechanical, electrical, procedural interfaces eg. POTS, DSL, 10/100 BASE-T, SONET/SDH
2. Data Link layer: flow control, error handling via creation of frames, medium access (in broadcast media networks) eg. Ethernet, ATM, PPP, 802.11
3. Network layer: routing, addressing, eg IP, IPsec, ATM
4. Transport layer: establishing & deleting connections, end to end flow control eg TCP, UDP

5. Session layer : enhanced services, eg. synchronization of diff streams that are part of 1 application, inserting checkpoints into a data stream to be able to resume after a crash eg. SDP (Session Description Protocol)

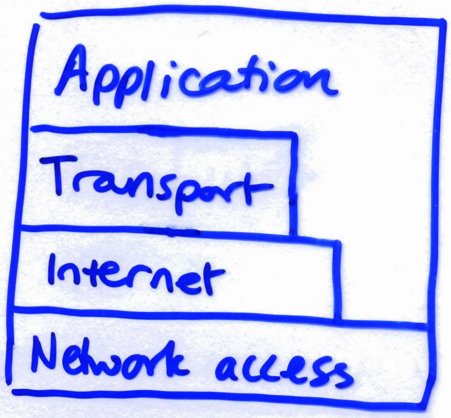
6. Presentation layer: syntax, conversion of data formats eg XDR (external Data Represⁿ)

7. Application layer : protocols providing commonly needed functionality to applications
   eg. FTP, HTTP, SMTP, DNS

- functions of layers 5, 6 & 7 are often embedded into the application in practice, rather than being implemented as separate common services

Internet (TCP/IP) Architecture

- Reference model : a description of existing protocols



- layering is not strict : application can bypass the defined transport layers & directly use TCP or an underlying network
- hourglass shape : wide at top & bottom, narrow in middle

- Network access layer: corresponds to data link + physical layers in OSI model
- Internet layer: single protocol — Internet protocol (IP) — which supports interconnect^n of different networks into a single logical internetwork
- Transport layer : 2 main protocols — Transmission Control Protocol (TCP) & User Datagram Protocol (UDP)
- Application layer: protocols enabling interoperation of popular applications eg. FTP, SMTP, HTTP, DNS

Architectural principles

- end-to-end principle (Saltzer et al 81)
  - originally assumed that the network should retain no state, & concentrate on speed & simplicity, while the maintenance of state & overall intelligence should be at the edges
  - not appropriate for large multicasts & broadcasts, especially over lossy networks

- robustness principle (Postel 81)
  - be conservative in what you do, be liberal in what you accept from others
  - software on other hosts may contain deficiencies that make it unwise to exploit legal but obscure protocol features

Physical layer encoding

- encoding of binary data into signals carried on a link
- assume a modulation scheme with 2 discrete signals, high & low, e.g. 2 different power levels on an optical link, or 2 different voltages on a copper link

non-return to zero (NRZ)

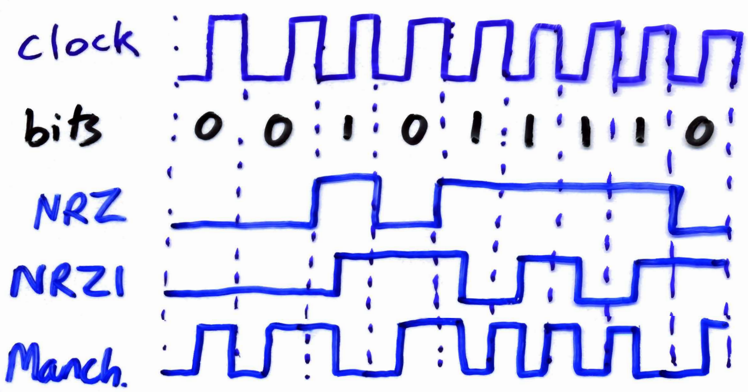
- map data value 1 onto the high signal & data value 0 onto the low signal
- disadvantages: too many consecutive 0s or 1s may cause
  - baseline wander (baseline at receiver drifts towards average of more recent signals resulting in erroneous detection)
  - clock drift (receiver synchronizes its clock with sender's via transitions)

non-return to zero inverted (NRZI)

- encode a 1 as a transition from the current signal, & a 0 by staying at the current signal
- solves problem of consecutive 1s, but not consecutive 0s

- **Manchester encoding**, eg. used by 10Base-T
  - encode a 1 as a high-to-low transition, & a 0 as a low-to-high transition (sender transmits XOR of NRZ-encoded data & the clock)
  - disadvantage: doubles the rate of signal transitions on link, so unsuitable for high data rates

Eg.



- **4B/5B**, eg used in 100Base-TX
  - encode 4 bits of data in a 5-bit code chosen s.t. each codeword has at most one leading 0 and at most two trailing zeros (at most 3 consecutive 0s in code)
  - transmit using NRZI
  - 80% efficiency

- <u>scrambling</u>. eg used for SONET payload
  - calculate XOR of data with a well-known bit pattern with many transitions

Link layer framing in packet networks

- a network layer packet is encapsulated in a link layer frame for transmission over a link
- receiving node's network adaptor must determine frame's start & end

• Byte-oriented protocols

- older approach to framing
- view a frame as a collection of bytes (characters)
- sentinel approach eg PPP
  - start (end) of a frame denoted by a special start(end)-of-text character
  - if the end-of-text character appears in the body of the frame, preceed it with an 'escape' character (character stuffing)
- byte-counting approach
  - include the no. of bytes as a field in the frame header instead of using an end-of-text value

Bit-oriented protocols

- view a frame as a collection of bits
- eg. High-Level Data Link Control (HDLC)
  - uses sentinel approach with bit stuffing
  - beginning & end of frame denoted with distinguished bit sequence 01111110
  - if 5 consecutive 1s have been sent, sender inserts a 0 before sending the next bit
  - if receiver sees 5 consecutive 1s,
    - if the next bit is 0, it is removed as a stuffed bit
    - if the next 2 bits are 10, it is the end-of-frame marker
    - otherwise it is an error & the frame is discarded

• Clock-based framing

- eg SONET (Synchronous Optical Network)
- all frames have the same size eg 810 bytes in STS-1 (lowest speed SONET link)
- receiver looks for a special bit pattern appearing every 810 bytes to sync up

Link layer error detection

implemented in dedicated hardware in adapters → can use more complex error detection schemes than transport layer error detection

- <u>cyclic redundancy check</u> often used at link layer
  - $(n+1)$-bit message represented by a polynomial of degree n

    eg 10011010 → $x^7 + x^4 + x^3 + x$

    76543210
  - sender & receiver agree on a diviser polynomial $C(x)$ — choice of polynomial affects what types of errors are detected
  - to send an $(n+1)$-bit message $M(x)$ using a degree-k divisor polynomial $C(x)$, we send an $(n+k+1)$-bit coded message $P(x)$ derived from $M(x)$ that is exactly divisible by $C(x)$
  - $P(x)$ can be formed as follows:
    1. Let $T(x) = x^k M(x)$ (i.e. append k 0s to $M(x)$)

2. Divide T(x) by C(x) & find the remainder

eg.

```
              11111001
1101 ) 10011010000      <- message M(x)
C(x)↑  1101
       ----
        1001
        1101
        ----
         1000
         1101
         ----
          1011
          1101
          ----
           1100
           1101
           ----
            001000
               1101
               ----
                101  <- remainder
```

3. Subtract the remainder from T(x) to obtain P(x)

eg. 10011010000 − 101 = 10011010101

- receiver divides the received polynomial by C(x)
  - if remainder is 0, concludes no errors
  - if remainder is nonzero, error detected
- polynomial C(x) chosen s.t. it is unlikely to divide evenly into common error polynomials E(x) (& therefore P(x) + E(x))

Common CRC polynomials

CRC	C(x)
CRC-8	$x^8 + x^2 + x + 1$
CRC-10	$x^{10} + x^9 + x^5 + x^4 + x + 1$
CRC-12	$x^{12} + x^{11} + x^3 + x^2 + 1$
CRC-16	$x^{16} + x^{15} + x^2 + 1$
CRC-32	$x^{32} + x^{26} + x^{23} + x^{22} + x^{16} + x^{12} + x^{11} + x^{10} + x^8 + x^7 + x^5 + x^4 + x^2 + x + 1$

- Ethernet & 802.5 use CRC-32
- ATM uses CRC-8, CRC-10 & CRC-32

- If x^k & x^0 terms have nonzero coefficients in C(x), can detect all single bit errors

  If C(x) has a factor with at least 3 terms, can detect all double bit errors

  If C(x) contains the factor $(x+1)$, can detect any odd no. of errors

  If C(x) has degree k, any burst error of length < k (ie. < k consecutive errored bits) can be detected