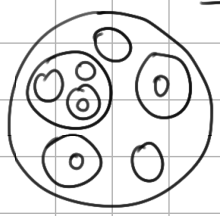
Network Access Layer

Hop 2 hop

Link Layer: Responsible for transferring datagrams from one node (Router/host) to physically adjacent node.

Layer 2

Link layer
→ Framing (Encapsulates datagram into a frame)



Definations:

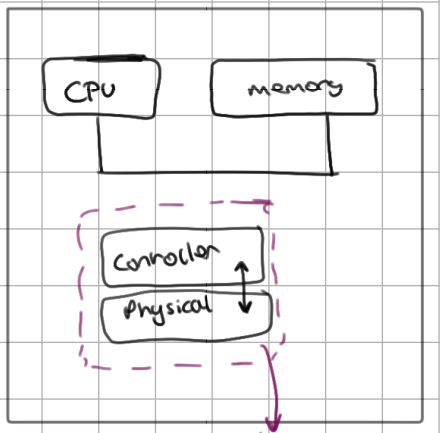
Flow Control: Pacing between adjacent sending and recieving nodes

NIC: Network interface card

Error Detection: noise + errors caused by signal attenuation
↳ Reciever detects errors + Signals transmission + drops frame

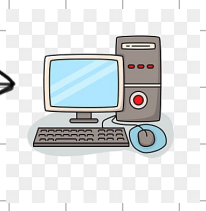
Error Correction: Reciever corrects bit errors w/o transmission.

Half-duplex and full-duplex: nodes can transmit but not at same time



Sending Side

Recieving End



① puts datagram into a frame

② Adds error checking bits and reliable, data transfer + flow control

① Looks for errors, reliable data, flow Control

② Extracts datagram, and sends everything too the upper layer.

ECE: Error Detection and correction bits.

Error Detection = not 100% reliable!!

Datagram

Sensitive Link

Everything in order??

no

Datagram

Addressing Scheme — MAC addresses.
↳ 12 hex digits

Data Link → switches
→ NIC

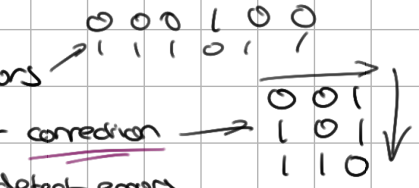
Every NIC has its own unique MAC address

Link Layer Services:

→ Framing: wraps up data w/ Header + trailer
→ Link Access: Determines when node can send data
→ MAC addressing: Frame uses this to identify sender/reciever LAN.
→ Reliable Delivery
→ Flow Control: Prevents sender from overwhelming reciever
→ Error Detection: Indentifies if bits are corrupted during transmission.
→ Error Correction: Fixes errors w/o retransmission
→ Duplex Modes: Half (one at a time) or Full (both at once) communication

Error Detection

Parity Bit: Adds 1 bit to detect single bit errors → only Detects
2D Parity: uses grid w/ rows and columns for correction → Corrects and Detects



checksum: Adds numbers in a segment to detect errors
↳ TCP/UDP Protocols

CRC (Cyclic Redundancy Check): Polynomial Division to detect burst errors very effective :)

Muliple Access Protocols: When multiple devices share the same channel.

Channel Partitioning: Divides channel into fixed portions (TDMA: time, FDMA: frequency)
Random Access: Devices send whenever they want, handle collisions (ALOHA, CSMA)
Taking turns: Devices take turns (e.g. token Passing)

TDMA: time division multiple access
→ Each device gets a timeslot
however unused slots = waste bandwidth

FDMA: Frequency Division
→ Each get a frequency band.
Good isolation but can waste unused bands

Random Access Protocols.

Pure ALOHA: sends anytime

slotted ALOHA: sends only on time slots

CSMA: sense channel before transmitting. If busy = waits

CSMA/CD: collision Detection. Abort on collision (ethernet)

CSMA/CA: Collision avoidance. Avoid collisions (wifi)

Ethernet: CSMA/CD Algorithm

1. NIC recieves datagram. Creates a frame
2. idle: start frame transmission
   Busy: waits until idle channel
3. NIC transmits frame w/o collision. NIC is then done with frame
4. NIC detects another transmission while sending: Abort, send jam signal
5. After aborting, NIC enters binary backoff

$$\text{efficiency} = \frac{1}{1 + 5\frac{t_{prop}}{t_{trans}}}$$

t_{trans} → time taken to transmit max-size frame

t_{prop} → max prop delay between 2 nodes in LAN.

Switches Vs Routers

	Switch	Router
Layer addressing	Data link (MAC address)	Network (IP address)
Tables	Forwarding	Calculates Routing Algorithm.

MAC address & ARP

MAC address → 48-bit Hardware

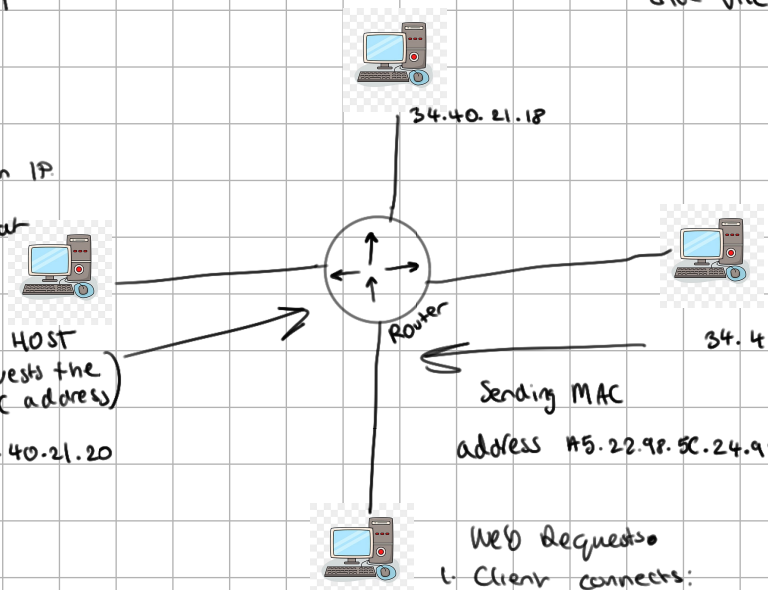
ARP → Address Resolution Protocol

= connects IP addresses to fixed physical machine (MAC).

- ARP translates 32-bit (IP) address to a 48-bit and vice versa.

LAN = Local Area Network.

When a new PC joins, it needs an IP. So it asks ARP to find an IP that matches the MAC address.



ARP vs DHCP

- DHCP prevents two IP addresses from being the same.

Web Requests:

1. Client connects:
   → uses DHCP to get IP
   → DHCP uses IP/ethernet/UDP
2. DNS Query:
   → needs MAC of router → uses ARP
   → sends DNS query via router to DNS ser
3. HTTP request:

→ Opens TCP connection to web server

→ Send HTTP GET over TCP

→ Web server responds w/ HTTP

Routing to another subnet: Device A wants to talk to B