CS348 Notes
Till but not including Layer 3
Till Midsem
Video Numbers: 3 - 15

OjMaha

I have prepared these notes by watching the videos from Networks Playlist. The following notes may be asynchronous and irrelevant to what Prof. Vinay teaches in class (cuz I do not pay attention during lectures lol). Further, these notes might not cover *everything* as explained in the video lectures. Consider these to be a supplemental read :). If you find any errors, do notify me so they can be edited.

Lec3

PHYSICAL LAYER

Simple method: +5V for 1, -5V for 0. → 2 wires close enuf.

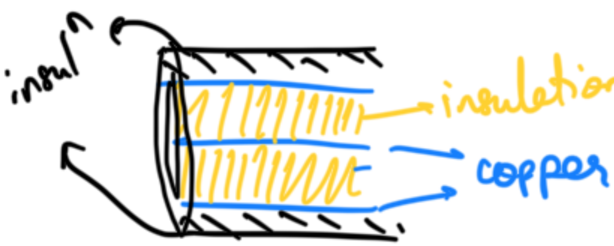
But cross-talking (emf induced due to $\frac{\partial B}{\partial t}$ (interference)) might occur.

So use twisted-pair cables → keep loop area small, cancel out effects.

- CAT-3: 10 Mbps over 100m
- CAT-5: 100 Mbps over 100m
- CAT-6: 1Gbps over 100m or 10Gbps over 50m

An ethernet connector has 8 wires ≡ 4 twisted pairs.

Co-axial cables:



V volt

Thin Net Coax: 100 Mbps, 200m → 0.2in

Thick Net Coax: 100 Mbps, 500m → 0.4in

Optic fibre:

TIRs, low attenuation.

We call the rays "modes". Multi-mode fibre: allows multiple modes to pass through → But will lead to interference.

So, reduce diameter s.t. only one mode passes through. (or change refractive indices)

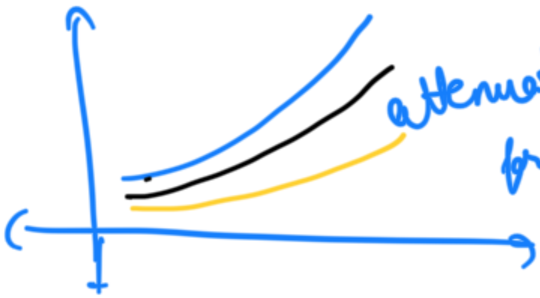
Attenuation in dB: $10 \log_{10}\left(\frac{P_{in}}{P_{out}}\right) = 20 \log_{10}\left(\frac{A_{in}}{A_{out}}\right)$. Generally calculated per

(P_{out}) (A_{out}) 1000 ft.

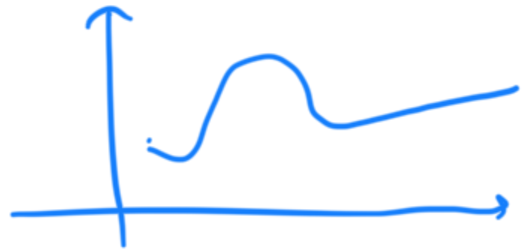
Attenuation is additive. ($A \xrightarrow{x\text{ dB}} B \xrightarrow{y\text{ dB}} C$ total $= x+y$)

dBm : $10 \log_{10} \frac{P}{1mW}$ $\Rightarrow$ Power in terms of dBm. dBW is another scale for absolute power. (1W instead of mW)

Cat 3 > Cat 5 > Coaxial (attenuation) ; here it increases with frequency.



attenuation v/s freq for



attenuation v/s freq for optical fibres $\rightarrow$ thus used for higher freqs.

Wireless Links:

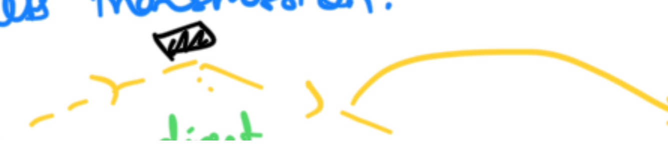
$P_{out} \propto P_{in} \cdot \frac{A}{d^2}$ theoretically. But $P_{out} \propto \frac{P_{in}}{d^{\alpha}}$ $\alpha \in (2,5)$ realistically.

Directional Antennas: instead of spherical signals, focus transmission along single dir^n.

$\hookrightarrow$ MIMO : 4 directional antennas to transmit and receive across 4 dir^n.

Massive MIMO: 5G.

Interference, Obstruction are hurdles in wireless transmission.



Further, multi-path can also occur. i.e.:

After reflection; signal gets flipped, interfering with og signal.

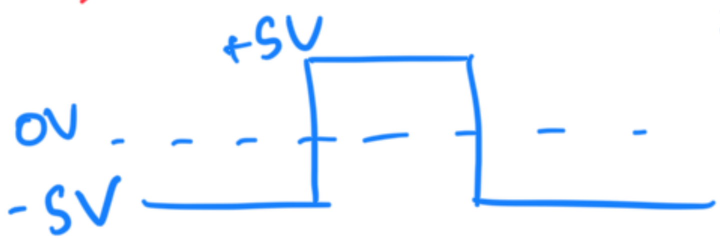
Lec4 +5+6

SIGNALLING

WIRED:

Line Coding:

1) Non-return to Zero (NRZ)



issues i) clock recovery.

might be erroneous as clock async.

issue ii) baseline wander: the amplifier may induce a DC-offset. more errors.

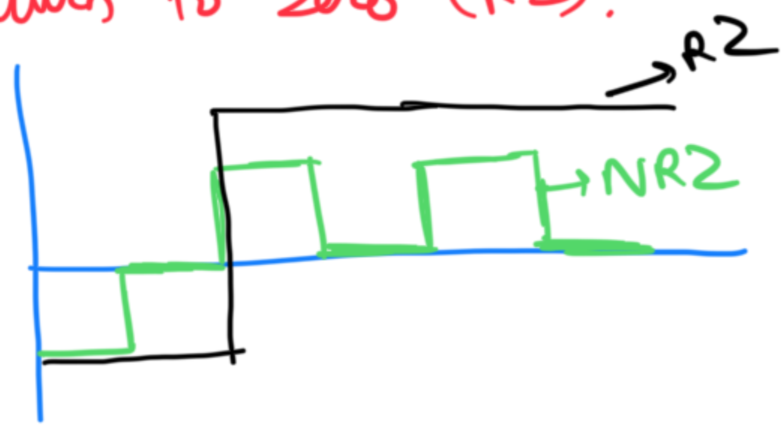
for this, add a High Pass Filter at receiver side.

↳ pushes offset a bit lower and pushes avg to 0.

Now, when we have a long string of 0s, the signal gets pushed up.

The high pass filter treats it like a dc-signal and filters it out.

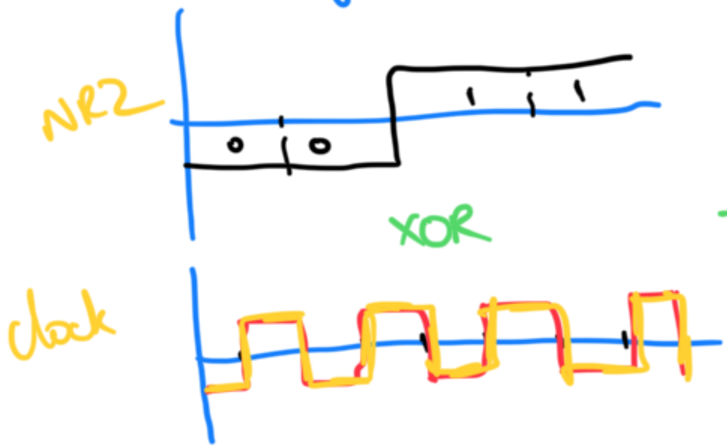
2) Return to Zero (RZ).



problem: 3 voltage levels.

3) Manchester Encoding (802.3 IEEE Ethernet)

XOR signal which has clock period half of sent bit period. i.e. each bit goes through 2 clock cycles.



→ treat −5 as 0
+5 as 1
↳ upward transition ⇒ 0.

No clock recovery issue.
No baseline wander. (avg 0 per bit)

But we need to know where a bit signal starts as well as correct polarity.
For this, send a preamble to help sync rec with sender.

4) Differential manchester encoding (used in IEEE 802.5 Token Ring LANs)

1 ⇒ 1st half of bit is <u>same</u> as last half of prev bit.

0 ⇒ 1st half of bit is <u>opposite</u> of last half of prev bit.

Here, we don't need to check polarity. Even with flipped polarity, result is same.

5) 4B/5B Coding

Take a 4 bit chunk. map it to 5 bits. → s.t. at most 3 consecutive

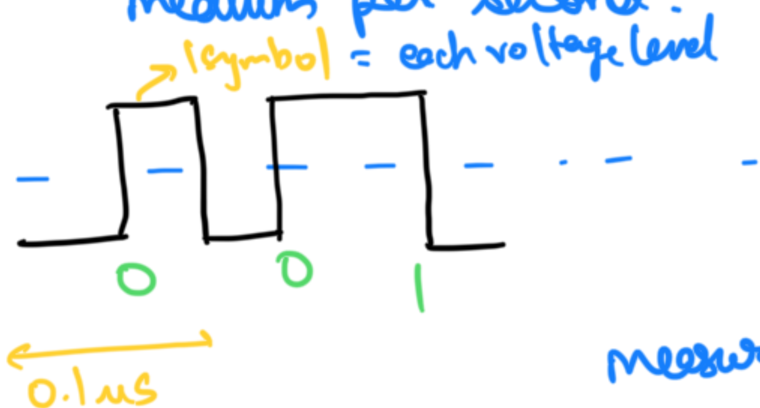
2^4 chunks mapped to 16 different 5 bit codes. 0s in transmitted message.
Baseline wander drops.

disadv. : sending 5 bits instead of 4. (redundant bits reducing efficiency)

Bit rate : max bits transmitted per second.

Baud rate : max no. of distinct symbol changes made to the transmission medium per second.

eg:-



bit rate = $1/0.1\times10^{-6}$ = 10Mbps

baud rate = $1/0.05\times10^{-6}$ = 20M symbols / second.

measures how efficiently I am using medium.

We know that attenuation in a wire ↑ with ↑ in freq.

Freq. range of a signal on a wire is related to baud rate.

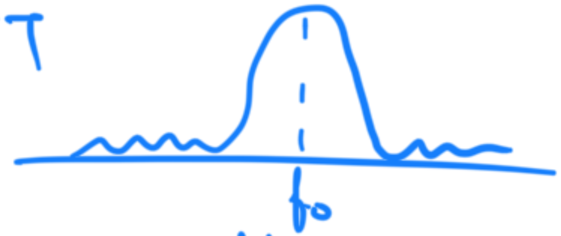
↳ usable frequency without causing too much attenuation.

WIRELESS MODULATION:

allot freq. bands for diff purposes.

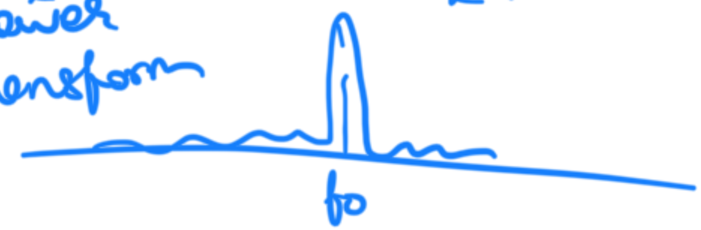
$A\cos(2\pi f_0 t) \to 1$. $A\cos(2\pi f_0 t + \pi) \to 0$.

option 1 : 1 wavelength of signal per bit ↳ FT



option 2 : 4 wavelengths of signal per bit. ↳ FT.

much narrower fourier transform



$$r(t) = \sum_i a_i\, s(t - \tau_i) + n(t)$$

τ_i → delay

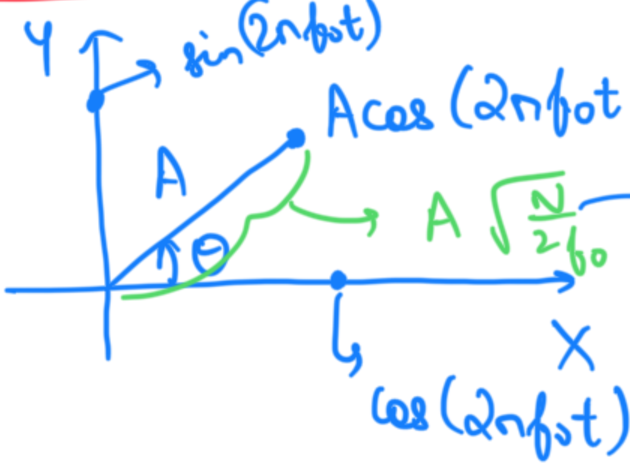
$\sum_i$ ↳ multipath

$n(t)$ ↳ additive white gaussian noise.

Bit error rate : fraction of bits recd in error.
↳ (BER)

Signals in vector space

→ change amplitude & phase to convey diff. bits



$r_x = \langle r(t), \hat{e}_x \rangle$

$r_y = \langle r(t), \hat{e}_y \rangle$

$$\langle f, g \rangle = \int_0^T f(t)\, g(t)\, dt.$$

$\hat{e}_x = \sqrt{\frac{2}{T}} \cos(2\pi f_0 t)$

$\hat{e}_y = \sqrt{\frac{2}{T}} \sin(2\pi f_0 t)$

$T = N/f_0$

N ↳ no. of wavelengths signal is sent for.

$r(t) = a \cdot s(t) + n(t)$

$s(t)$ → sent signal

$n(t)$ → AWGN

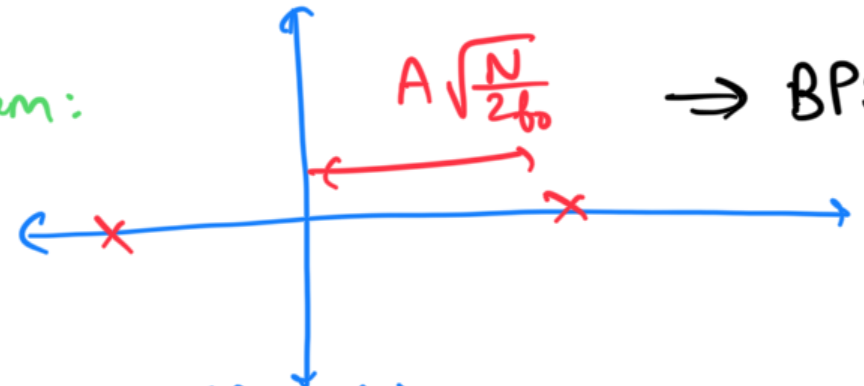
a ↳ attenuation

attenuation:

$$g(t) = A\cos(2\pi f_0 t - \theta) \Rightarrow g_x = A\sqrt{\frac{N}{2f_0}}\cos\theta \;;\; g_y = A\sqrt{\frac{N}{2f_0}}\sin\theta$$

Suppose, bit 0 : $s_0(t) = A\cos 2\pi f_0 t$
bit 1 : $s_1(t) = -A\cos 2\pi f_0 t$ } $0 \le t \le N/f_0$ (sent for N wavelengths)

Constell^n Diagram:



$\rightarrow$ BPSK (binary phase shift keying)

But recd is $r(t) = a\,s(t) + n(t)$

So detection @ receiver: (i) $r_x = \langle r(t), \hat{e}_x\rangle$ & $r_y = \langle r(t), \hat{e}_y\rangle$

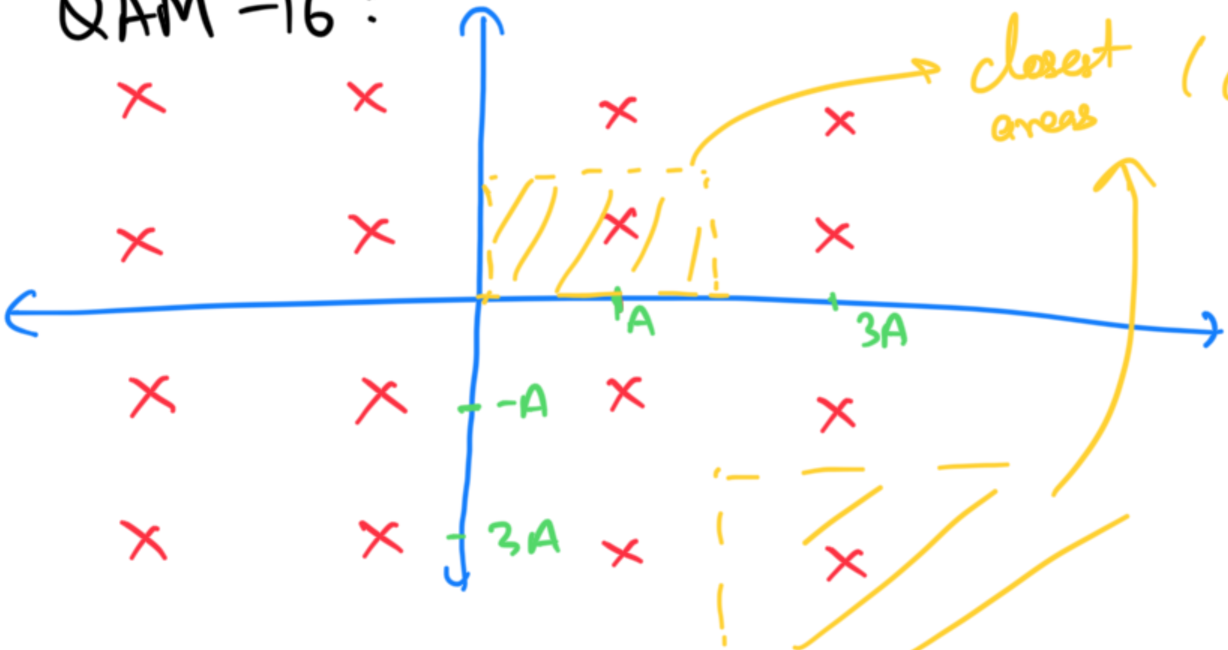
Then find which constell^n pt is closest.

remember ass^n 1 CS348.

Signal to Noise Ratio : $\frac{\text{Signal Power}}{\text{Noise Power}}$

QPSK (Quadrature ...) (2 bits @ a time)

QAM-16 : 4 bits @ a time.



closest areas (classified into this pt)

QAM-256 : 8 bits @ a time

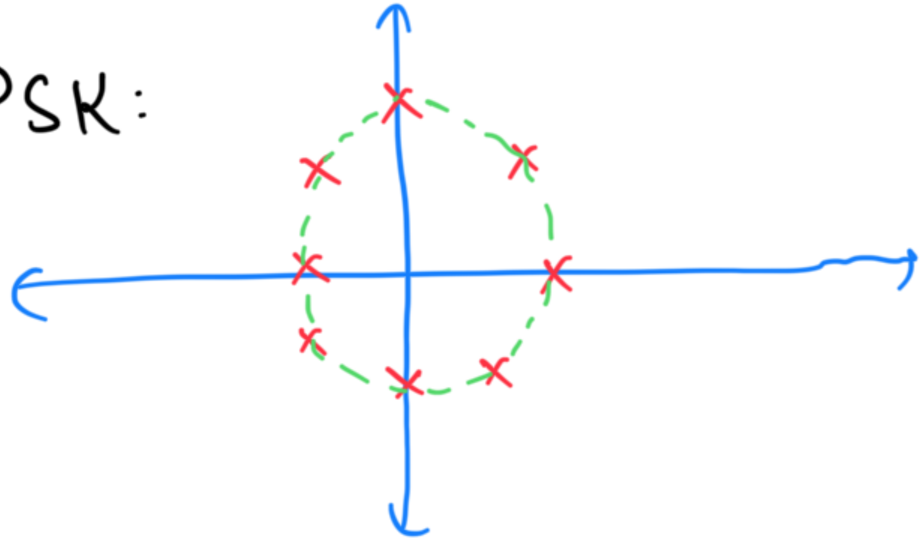
The constellⁿ diagram is determined by the recd signal power. i.e. attenuation.

↳ i.e. if more power can be transmitted / receiver can receive more power then we can use higher order QAMs.

BER is a fn of SNR. BER of QAM-16 > QPSK > BPSK because constellation points are closer.

if high SNR ; use QAM-16 . low SNR ⇒ use BPSK .

8-PSK:



Lec7+Lec8

Data Link Layer

The DLL tries to decode which bits mean what.
The PHY layer decodes symbols, the DLL decodes frames.

HDLC: (High Level data link control) used in WAN

Begin seq	Header	Body	CRC	End Seq
8 bits	16 bits	variable	16 bits	8 bits

→ FRAME

0 111 1110 → call this SEQ.

$\underbrace{\text{SEQ SEQ SEQ}}_{}$ HEADER... BODY... CRC, SEQ, SEQ

no data to send

once data ready to send.

↳ helps in clock synchronization

But what if SEQ appears in BODY or HEADER?

Bit stuffing: if you see 5 consecutive 1s ; insert a 0 from the sender side.

→ SEQ can never occur now.

The receiver will know to ignore a 0 if it has 5 cons. 1s preceding

↳ stuffed bit.

Ofc, dont bitstuff the actual SEQ.

What if bit-error during transmission?

At Receiver : 0 11111 | if
0 → assume due to bit-stuffing & remove
10 → assume END seq
11 → assume error and discard entire frame till another SEQ occurs.

cuz this type of seq. is not ever expected.

But there might be other bit errors as well na.

CRC (Cyclic Redundancy check) : error detection.

Append k bit CRC to n-bit data word to form a (n+k) bit codeword

There are 2^k data words. There are 2^{n+k} bit words out of which only 2^n are valid. If we receive a codeword that is not one of these 2^n, it means invalid data. (But now it may happen that enuf bit errors lead another valid codeword.)

Min^m Hamming Distance (HD) over all pairs of codewords is aka HD of a CODE

Galois Field GF(2)

Finite field with 2 elements: 0, 1.

Addition is XOR. Multiplication is AND. Subtraction is same as add^n.

If $a_0 a_1 \dots a_n$ is a codeword, then cyclic shift of these codewords are also codewords. → CRC is a cyclic based code.

Generating a CRC: → easy to generate

(k+1) bit long divisor / generator. (s.t. remainder is k bit long)

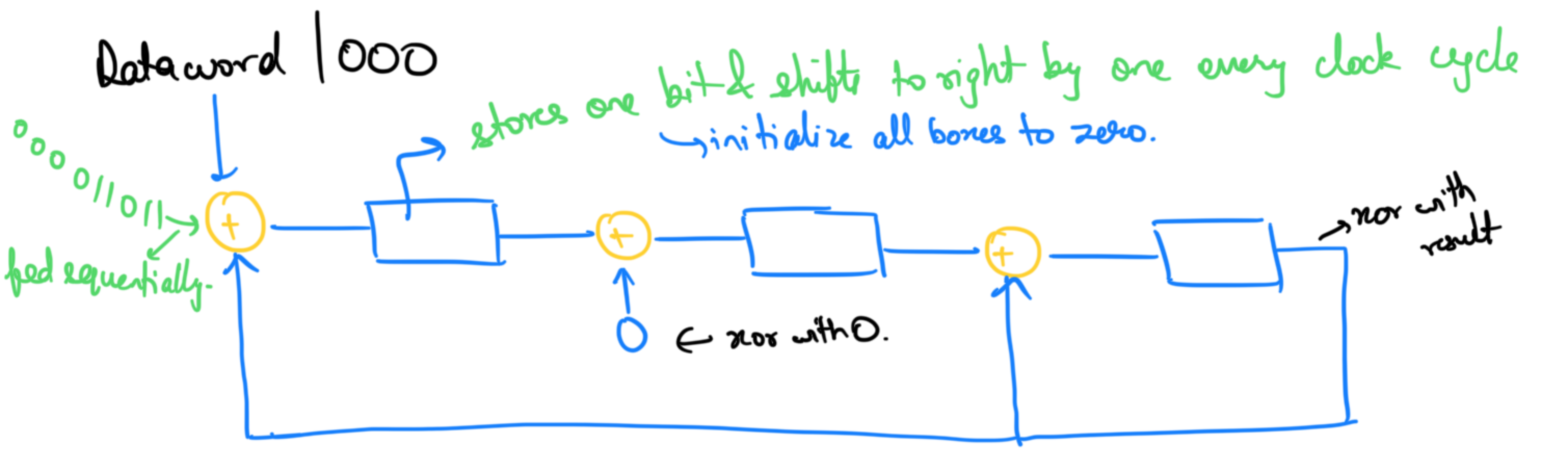
eg:- k=3, generator: 1101, dataword: 110110

```
          100011
       __________
1101  | 110 110 000      → append k bits of 0s.
      + 1101
        ----
         0001
       + 0000
         ----
          0010
        + 0000
          ----
```

Codeword: $\underbrace{110110}_{\text{dataword}}\ \underbrace{111}_{\text{CRC}}$

0100
+ 0000
1000
+ 1101
1010
+ 1101
0111 → CRC

We will try to implement a circuit that gives us the CRC.
We want our circuit to output the 3 bits encircled in green sequentially as we feed our data to it. In the end when data is over, we are left with the CRC.



essentially xors will be either with 000 or 101.

In the end we will have the CRC.

Essentially, take last k bits of divisor.
If leading bit=1, xor with these k bits. else xor with 0^k.

At Receiver end:

Option 1: recd: dataword & CRC. Append 000 to dataword, do division & check if CRC matches. if (mismatch) error;

Option 2: feed dataword + CRC into circuit. If end result is 000, no mismatch.

Polynomial Representation of CRC :

$1101 \to x^3 + x^2 + 0 \cdot x^1 + 1 \cdot x^0.$

$\to c(x)$

$1101 \times 11 \Rightarrow 10111$

$$\begin{array}{r} 1101 \\ \times \quad 11 \\ \hline 1101 \\ + \; 11010 \\ \hline 10111 \end{array} \Rightarrow x^4 + x^2 + x + 1$$

xor ← +

suppose codeword: $p(x) \to$ ...... $\underbrace{110}_{\text{CRC}}$.

Δ Error: $E(x) \to$ 00.....10100 3rd bit in error. 5th bit in error

Hence, receiver receives $P(x) + E(x)$.

Divide recd poly by $C(x) \xrightarrow{} $ generator. If remainder $!= 0 \Rightarrow$ bit error.

Now; we must ensure $[P(x)+E(x)] \div C(x) \neq 0$ if $E(x) \neq 0$.

$\rightarrow \frac{P(x)}{C(x)}$ is 0 because $p(x)$ is constructed that way.

Single bit errors: $C(x) = x^k + \dots + 1$

$E(x) = x^i$. $i \in \{0, \dots n+k-1\}$.

$\rightarrow k \neq 0$.

Consider $C(x) = x^k + \dots\dots + 1$ $\quad C(x) \mid E(x) \Rightarrow (x^k + \dots + 1) \cdot (x^m + \dots + x^q) = x^i$

$\Rightarrow$ not possible as 2 distinct powers in product.

Two bit errors: $C(x) = x^k + \dots + 1$ special form s.t. order $2^k - 1$.

$E(x) = x^j + x^i \quad (j > i)$

$= x^i (x^{j-i} + 1)$ [Try to write each poly. as a product of irreducible polys]

Consider $C(x) = x^k + \dots + 1 = f_1(x) \dots f_m(x)$. Clearly, $\exists\, l$ s.t. x^i is cancelled out by $f_l(x)$

i.e. x^r cannot be a factor of $c(x)$ so x^i in $E(x)$ can't be cancelled out

But, it is possible that $c(x) \cdot d(x) = x^r + 1$ for large values of r.

Here, $r = j - i$ = dist. b/w errors.

The smallest r s.t. $c(x) \mid x^r + 1$ is called its order (or exponent).

It is known that we can find $c(x)$ of $x^k + \dots + 1$ s.t. its order is $2^k - 1$.

for $k = 16$; as long as $j - i < 2^{16} - 1$; we are safe.

Odd No. of Errors: $c(x) \mid (1+x)$ or $c(x)$ has even no. of terms.

$E(x) \neq (1+x) \cdot g(x)$. Why? Consider LHS. $E(1) = 1$ as odd no. of terms.

Consider RHS $= (1+x) \cdot g(x) = (1+1) \cdot g(x) = 0$.

Similar reasoning for even no. of terms. Consider $E(1)$.

HDLC uses CRC-16-IBM with $CRC = x^{16} + x^{15} + x^2 + 1$

CRC-32 : $C(x) = \dots$

Burst of Errors:

$C(x) = x^k + \dots + 1$; $k > l-1$.

↳ continuous erroneous chunk of bits. l-bit burst of errors. i.e. $E(x) = x^{l+i-1} + \dots + x^{i+1} + x^i$

$E(x) = x^i \left(x^{l-1} + x^{l-2} + \dots + 1 \right)$.

Consider $C(x) = x^k + \dots + 1$. $\longrightarrow$ no factor x^p for any p.

If $\underline{\underline{l-1 < k}}$; $C(x) \nmid E(x)$. Bursts of $l < k+1$ will be detected.

Lec9

ARQ: (Automatic Repeat reQuest) → WiFi uses this.

↳ handles "reliability".

After sender sends last bit, receiver sends an ACK frame as a reply.

If sender receives ACK, successful transmission yay.

How long should sender wait for ACK frame? (2 or 3 times Round trip time)

(data is sent at speed of light)

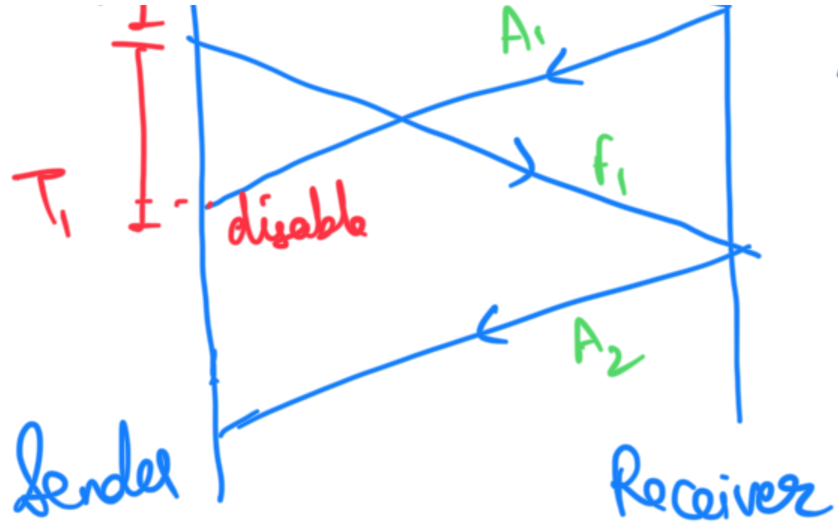
So, after sender sends first bit, it has a timeout for T_1 secs (wait for ACK). If it receives ACK within T_1, disable timeout and transmit the next frame. If however, the timeout expires, re-transmit the frame.

Can happen due to noisy transmission of data or noisy transmission of ACK

re-transmission of frame even though data recd correctly

If Timeout is small:

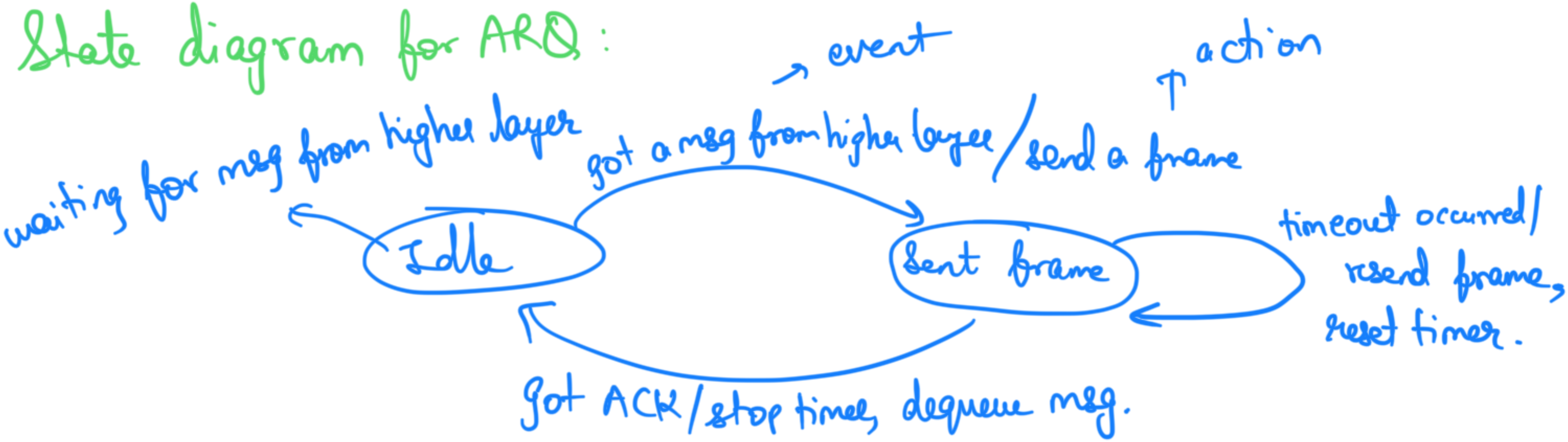
i.e. sender starts transmitting the same frame



for the 2nd time. Then it receives ACK from the 1st transmission and disables the 2nd timeout. (unnecessary retransmission)

If timeout value is too large, might end up wasting a lot of time waiting.

State diagram for ARQ:



Lec10

Medium Access (DLL)

Ethernet LANs: (wired)

→ use CSMA.

TDMA (time division multiple access)

↳ divide time into slots. One of the node is master. The master decides when slots begin and end. A: slot 1, B: slot 2, ... problem is many slots may become vacant. another problem is wut if master fails. Another is wut if a new node joins network.

CSMA (carrier sense multiple access) → it is a random access protocol.
↑ schedule not decided in advance.

broadcast: all nodes are destinations. unicast: single node is dest. multicast: multiple nodes.

The sender while preparing the frame mentions receiver.

The receiver discards frame if its not intended for them even tho it receives the frame.

$A \to B$
$C \to D$ simultaneously. Collision will occur.

We need a way for A & C to detect collision has occurred.
After detection, A & C back-off for a random wait time before transmitting.

problem i) still a collision may be repeated (if diff too small)
ii) won't work if too many people try to transmit at same time.

carrier sense: check if somebody else is transmitting before you transmit.
If medium is bz, ofc don't transmit. (note that simultaneous transmis. collision can't be avoided)

CSMA-CD: → IEEE 802.3

carrier sense: energy of signal in the wire > threshold (noise energy) ⇒ busy.

suppose A, C carrier sense; find that wire is not bz and start transmitting simultaneously. Now their signals get added up on the wire.

collision detection: if energy > threshold$_2$ ⇒ collision

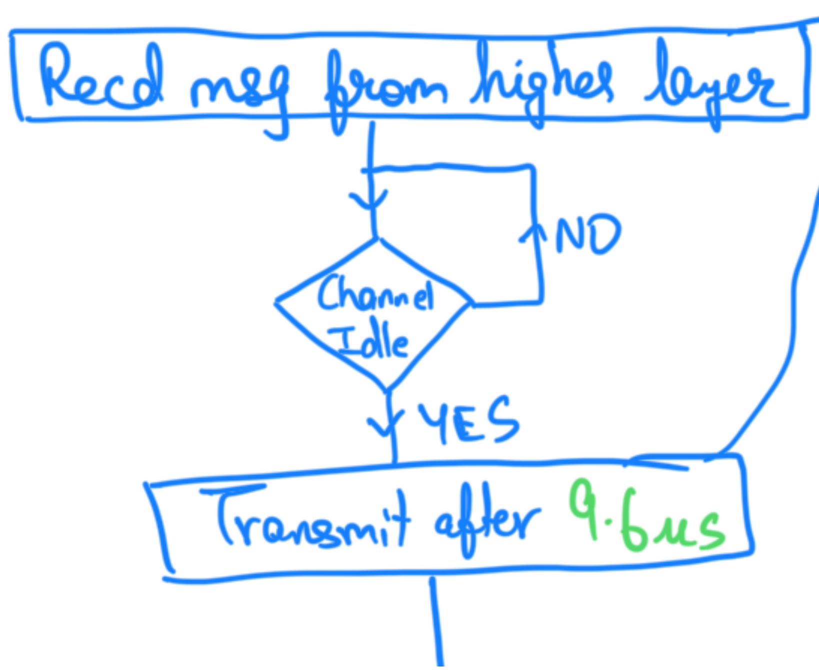
between energy for single & double transmission.

problem: say A detects collision first. and C is still in the process of detecting it.

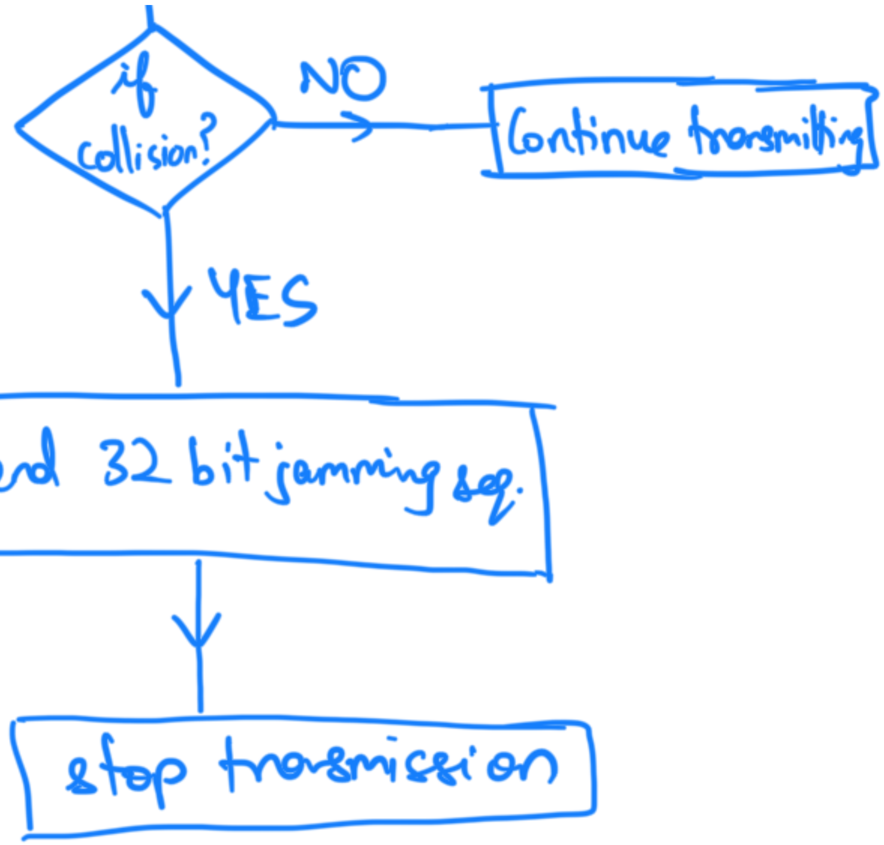
- Case (i). A stops transmitting ⇒ C may never detect collision → huge problem.
- Case (ii). A continues transmitting till end so C detects collision.
  - waste of resources.

so A sends a jamming signal (32 bit) to notify everyone that collision has occurred. Then A stops transmitting. Then C detects collision, sends it own jamming signal and stops transmitting.

flow chart:



Why wait for some time??
Suppose A→B & C→B.
B needs some time to process A's frame (CRC, etc) before it starts listening to C's frame.



frame details in 802.3 :

SFD: start frame delimiter (it is possible that few preamble bits are lost since receiver may not know when msg began)

Preamble	SFD	Destin^n MAC Addr	Source MAC Addr	Length	Payload	CRC
7 byte	1 byte	6 byte	6 byte	2 byte	46-1500 byte	4 byte

$\longleftarrow$ 64 to 1518 bytes $\longrightarrow$ (Destin^n MAC Addr to CRC)

IPv_4 : 2^{32} diff addresses (proved to be less). MAC : 2^{48} diff addresses.

IPv_6 : 2^{128} diff addresses.

Why keep a min^m frame size?

Consider A —— B —— C.

A wants to transmit to B. C also wants to transmit to B. (simultaneously)

A transmits message for 2 us and the signal reaches C in 12.5 us.

Similarly, C also does this. Now, since A had already finished transmitting the message when it recd C's signal, it won't detect a collision. Even C won't detect collision. Only B detects collision here.

We must provide some way s.t. the sender detects collision before his frame ends / collision occurs at receiver.

Thus, A's frame should last atleast 12.5+12.5 = 25 us long (Round trip A → C → A)

≈ 50 us is the RTT time usually. Considering delay due to repeaters as well.

64 bytes, 10 Mbps ⟶ 51.2 us. (2.5 km long wire)

↳ to ensure collision detection occurs before you stop transmitting.

✱ Why keep max frame size?

large frame sizes ⇒ i) more probability of bit errors

ii) others need to wait longer to get a chance to transmit

iii) mem. requirement @ NIC increase (to perform CRC check on whole frame)

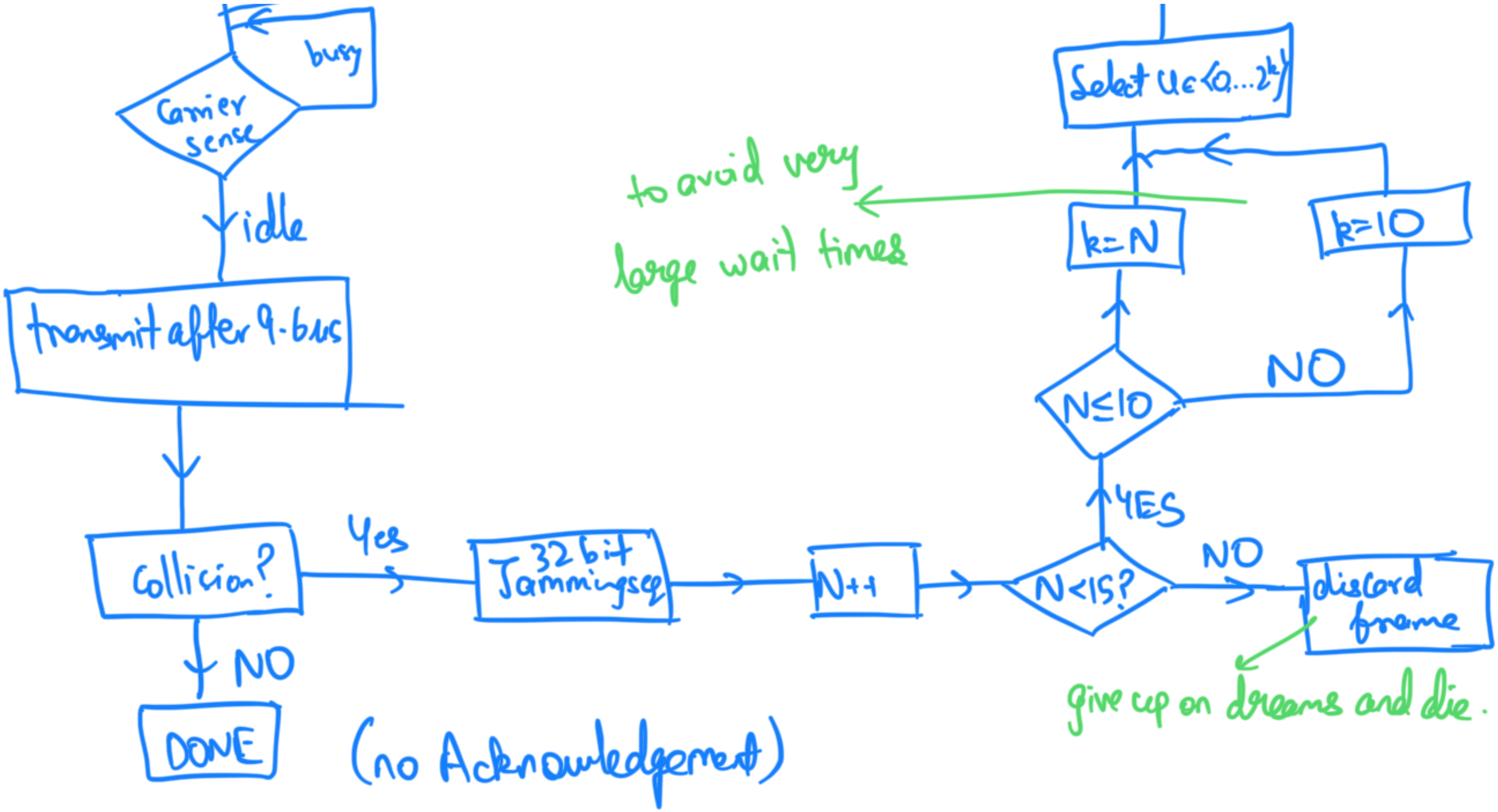
Exponential Back-off: increase interval from which wait-time is chosen exponentially.

wait-time $\in \Delta . U . \{0, 1, \dots, 2^k\}$ — uniform distrib^n

1^{st} collision: $k=1$; 2^{nd} collision: $k=2$ . . . . . .

This ensures all frames get through and reduces no. of repeated collisions.

flow chart.


busy
Carrier sense
idle
transmit after 9.6 µs
Collision?
Yes
NO
DONE
32 bit Jamming seq
N++
N<15?
NO
discard frame
give up on dreams and die.
YES
N≤10
NO
k=N
k=10
Select u∈{0,...,2^k}
to avoid very large wait times
(no Acknowledgement)

Lec11

WiFi (Wireless LANs)

WAN is wide area network.

The signal decay is very fast in wireless transmissions. If you use CSMA/CD, it is possible that C does not even hear A cuz signal too low. Hence, Carrier sense & collision detection might not work.

Carrier sense → based on energy

↓

still possible if devices close by

collision detection → based on signal recd or not.

→ signals recd will be orders of magnitude different. (A's own signal will be a few powers higher than C's)

is it possible to subtract own signal? point is if A transmits $a(t)$; it will receive $\beta . a(t)$ $(\beta \neq 1)$. i.e. A's receiver might not detect $a(t)$ as is.

How is collision detected?

if no ACK recd from access point (AP), assume collision.

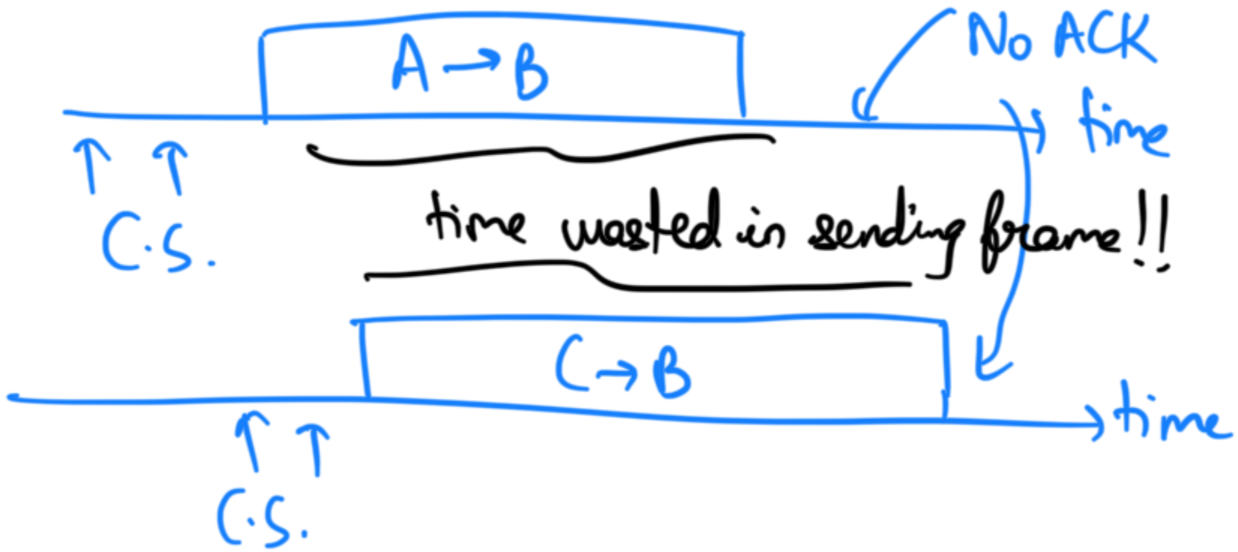
access point → hub linking devices together.

Note that all devices are able to reach their nearest AP but not necessarily able to reach each other.

↓

Hidden Terminal Problem:

B can hear A & C.
A & C can't hear each other.



Use Virtual Carrier Sensing:

Exploit fact that B can hear both.

RTS (request to send): short frame to tell receiver that I intend to send data.

CTS (clear to send): short frame to tell sender that sender can transmit.

ACK (acknowledgement): tell sender data recd successfully.

A → B : RTS ; B → A : CTS ; A → B : data ; B → A : ACK.

Note that C doesn't hear RTS. C hears CTS. C decides to stay silent even if it has data to send. C also receives ACK. Once it receives ACK, it can decide to send an RTS. But how long to wait? What if B never sends ACK? Hence, inside CTS, B sends NAV_{CTS} (Network allocation vector) to all other devices which contains how long medium should remain silent. But B doesn't know how long the message is. So, inside RTS, A sends another NAV_{RTS} mentioning this. Thus,

NAV_{RTS} : $CTS_{time} + Data_{time} + ACK_{time}$

NAV_{CTS} : $Data_{time} + ACK_{time}$

Note that peeps can hear only B. (i.e. CTS & ACK)

Rule: Anyone hearing an RTS or CTS should wait for NAV.

It is possible ∃ D who can hear A (thus the RTS) but not B (no CTS).

Since RTS_{NAV} is recd by D; it should be gupchup ekdum. The idea is that all devices reachable by sender and receiver should be silent while transmission is occurring since Wi-Fi is a two-way protocol.

In practice: why? cuz we want more & more people to hear it. if we use high then peeps far wont hear.

RTS and CTS are usually sent with lower modulation (BPSK) so their duration is high. And data is sent at QAM-256.

Lec12

Exposed Terminal Problem:

$D \longleftarrow A \qquad\qquad B \longrightarrow C$

D can receive from A but not from B.

Suppose A starts transmitting to D. B will carrier sense & decide not to transmit to C. Ideally, $A \to D$ & $B \to C$ can occur simultaneously. Because A & B are exposed to each other, nahi ho paaya sed.

Contention Window [CW]:

Each terminal has a contention window.

DIFS: distributed inter frame spacing

SIFS: short .....

why? explained later.

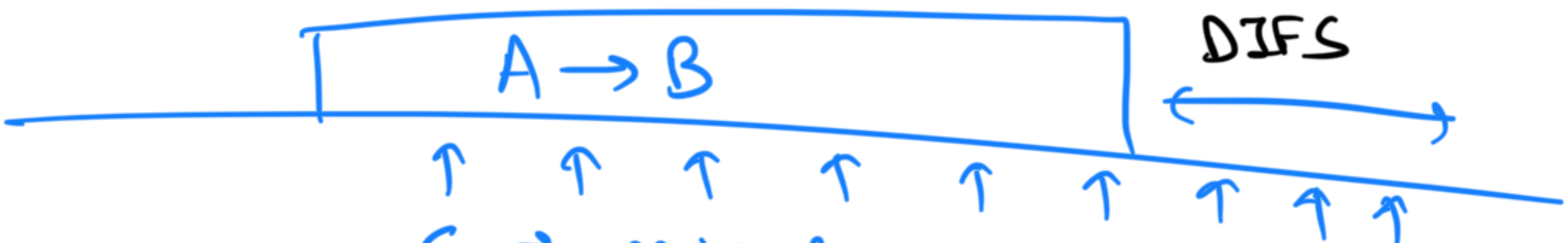
$\text{RTS} \underbrace{+}_{\text{wait for SIFS.}} \text{CTS} + \text{Data} + \text{ACK} \xleftrightarrow{\text{wait for DIFS.}}$ before having chance to transmit another frame

Why need SIFS???

Because when B hears RTS; B needs to shift circuit from being receiver to become RTS. B also needs to decode RTS. ↳ PHY/MAC processing.

Also, A needs to shift circuit from trans. to rec.

All of this will occur in SIFS time.



C → carrier senses.

C waits for atleast DIFS amt of time before trying to transmit.

Why? Because it is possible that C senses medium is free when SIFS period is on. We don't want C to start sending RTS here.

∴ DIFS > SIFS ensures this doesn't happen.

After this DIFS, then wait for random amt of time (if collision)

What should the random amt. of time be? (it should be a multiple of some number)

Divide time into slots of 9 us.

why?

Suppose C's wait time was one slot more. This time should be enuf for C to carrier sense, hear A and not transmit.

But now, what if the clock was slightly late? (async clock)

There is a possible offset (starting time for measuring slots).

Thus; prop. delay + possible offset + time to C.S. must be considered.

C must hear A + C.S. it → decide not to transmit.

else → switch circuit from RX to TX

receiver transmitter.

Considering all this; 9 us was chosen.

How long to wait?

$9 \mu s.$

$W \in \text{Uniform}\,[0, CW_{max}] \times \text{slot-time}$

rem. waiting time.

counter decrement. (DIFS $\neq$ idle slot)

For every idle slot, decrement W by 1.

wait times.

Imp: Suppose A & C both want to send a frame. Initialize w_A, w_c.

Let $w_A = 5$; $w_c = 3$.

When A waits for 3 slots; it hears C. (its wait counter becomes 2 secs)

Once C is done transmitting, everyone waits for DIFS time.

Let's say C again wants to send. Here, w_c is reinitialised (to say $w_c = 4$) but w_A is not reinitialized.

After the DIFS, A's counter decreases from 2. After this 2, C's counter freezes 2 (to rebegin baseline)

Whenever collision occurs; $CW_{max} = CW_{max} \times 2$

whenever ... $\Rightarrow$ ...

$\Downarrow$

i.e. timer expires together

$(W_{max} = \min(CW_{max}, \text{max. allowed value}))$

IEEE 802.11

MAC addr of sender, receiver.

DATA

frame checksum [CRC]

Frame:

Preamble	Signal			FCS

FCS → 4 bytes

$\longleftrightarrow$ MAC part of frame

modul^n used, coding rate

↓

if the rate is 3/4 it means N bits contain only $\frac{3N}{4}$ info bits

N/4 are redundant bits.

i.e. data becomes codeword with how many redundant bits.

		bit rate	Chanel Width
802.11 g →	64-QAM, rate 3/4	(54 Mbps)	20 MHz
802.11 n →	5/6	150 Mbps	40 MHz
ac →	256-QAM 3/4, 5/6	866 Mbps	160 MHz

ac → 256 QAM ... 3/4, 5/6 866 Mbps 160...

ax → 1024-QAM .... 5/6 1.2 Gbps 160MHz

these are for SISO
↳ single

MIMO (multiple input multiple output)

DIFS = SIFS + 2x slot_time

PIFS = SIFS + slot_time

↳ point co-ordination function. (don't ask y, it is not mentioned)

QoS: Quality of Service:

We can keep separate CW_{max}, max values, initial values of wait times to give higher priorities.

		CW_{max} Initial	Max Value
802.11c:	Voice	3	7
	Video	7	...

We need cross-layer

decreasing priority.	Video	1	15	interaction
	Others	15	1023	

Lec13

CSMA CD/CA is very effective for operating in unlicensed bands as it is decentralized.

But for licensed bands, there would be unnecessary waits if we used CSMA. eg:- Airtel has specific frequency bands.

One way is TDMA already mentioned before.

Uplink: users to Base Station $\rightarrow$ B.S.
(UL)

Downlink: Base station to User
(DL)

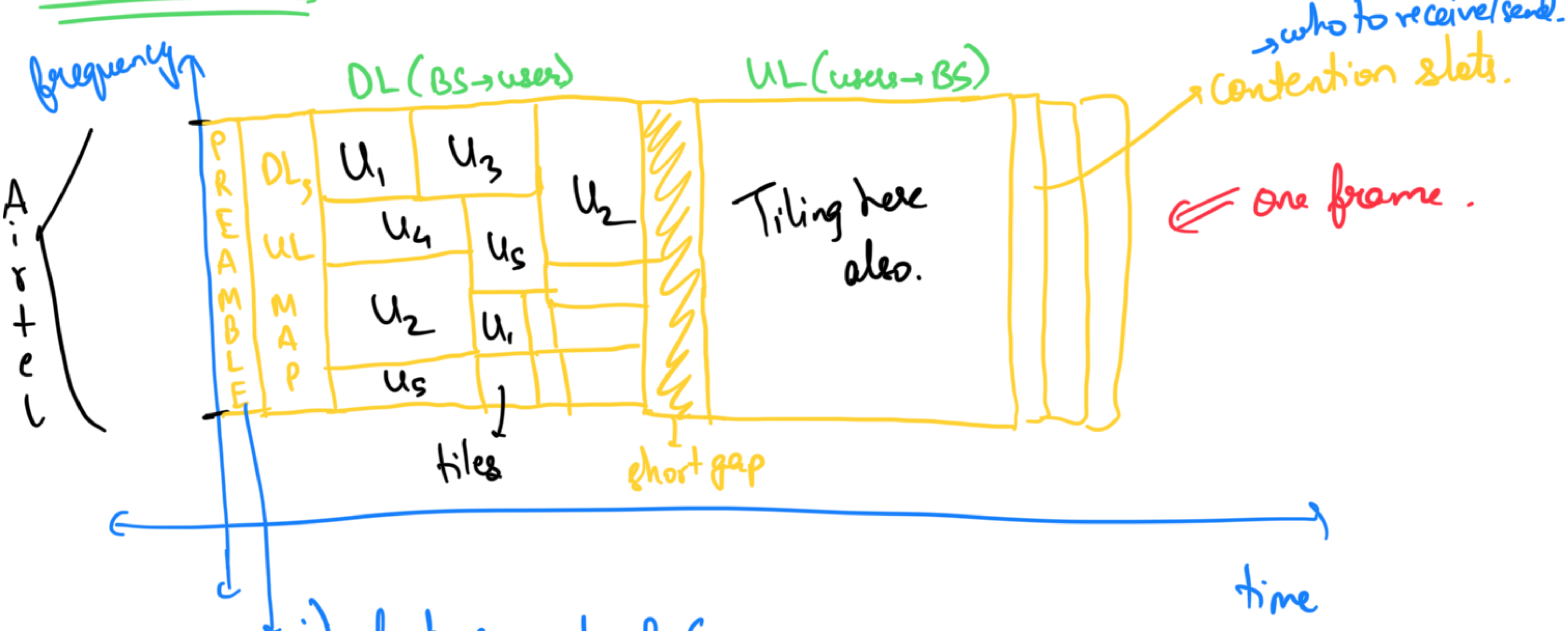
A set of slots is called a frame. First few slots of a frame contain UL-map or DL-map $\rightarrow$ schedule of user-slots.

FDMA: freq. division multiple access.

Another way is to split the allotted frequencies into narrow freq. bands and allot each band to user. Hence, only the frequency bands corresponding to user $\rightarrow$ BS and BS $\rightarrow$ user are relevant. Here, we need guard band.

some freq unallocated b/w Airtel & Jio s.t. overlap doesn't occur. Else interference go brrrr.

OFDMA: (orthogonal.) 4G-LTE uses this. TDMA + FDMA ka mix.



i) clock sync to B.S.

ii) indicate frame start

iii) attenuation & phase changes

the base stn tells the users ki itna atten^n I'm facing from you.

Then the user also gives feedback. This info is used to construct the DL & UL map. → contains tiling info essentially. i.e. some users might find transmitting at a certain freq. is good, for a certain time is good.

But how does B.S. know whom to allot slots? If I don't want to send / receive data; I shouldn't be allotted slots na.

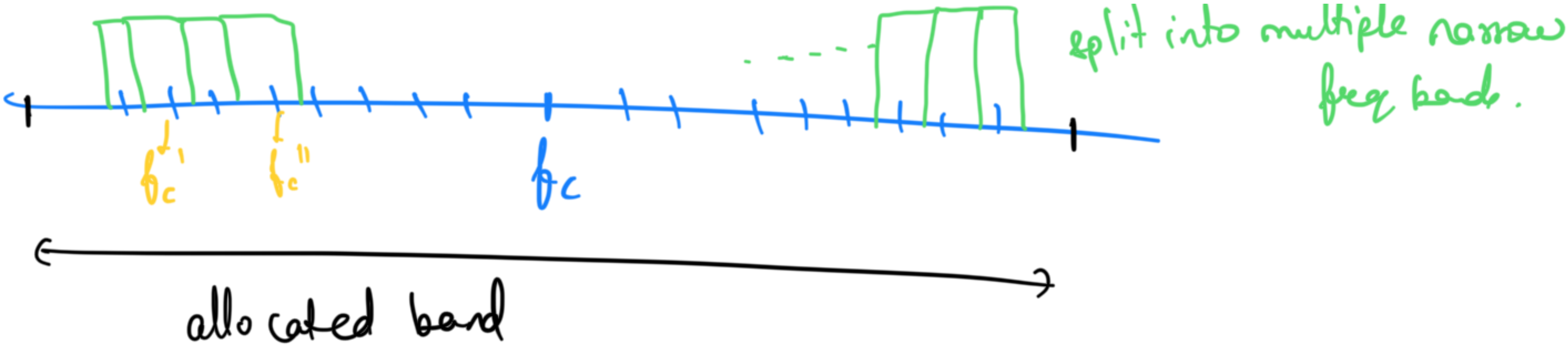
For this, there are contention slots. In these slots, anyone can transmit data. The data transmitted here is : I want to receive / send data toh please allot me tiles in the upcoming frames.

Since anyone can transmit in these slots, use CSMA for regulation.

here the ACK is simply the DL/UL map that user receives in next frame. (or ACK is sent as a separate DL reply from B.S.)

OFDM → WiFi

OFDM: orthogonal freq. division multiplexing. (≠ OFDMA)



On this channel, we can modulate f_c', f_c'' using modulation schemes like BPSK, QPSK and even transmit them together. The modulation schemes might even be different for f_c' & f_c''.

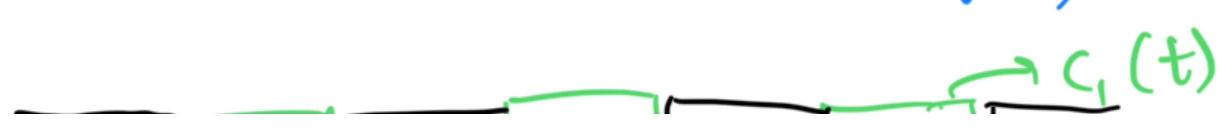
CDMA: Code Division multiple access.

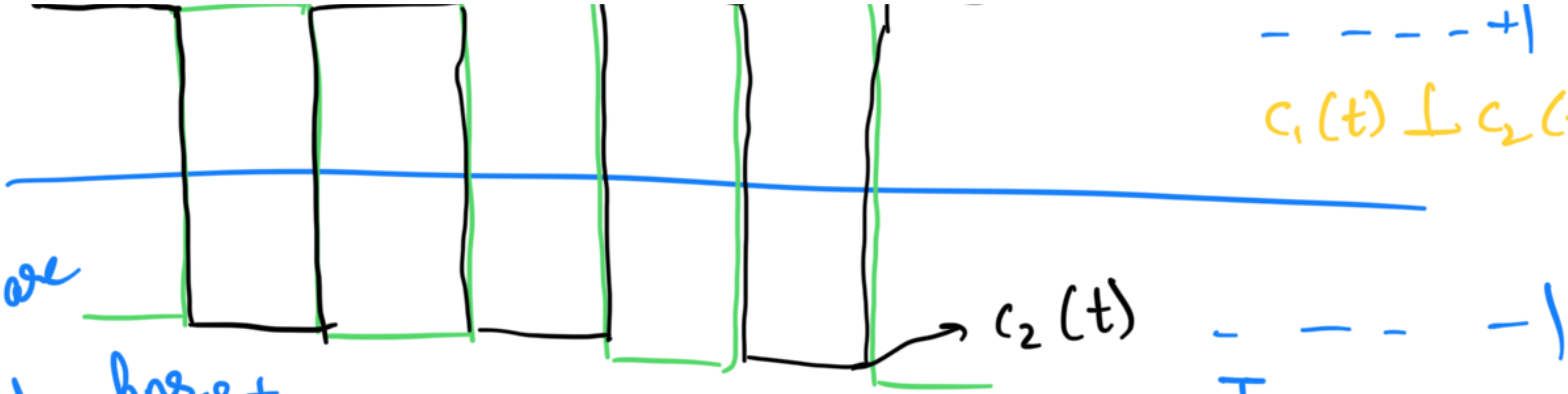
Multiple users can be assigned same time-freq. domain.

Spreading code:

$$\left.\begin{aligned} u_1 &: c_1(t) \times A\cos(2\pi f_c t) \\ u_2 &: c_2(t) \times \left(-A\cos(2\pi f_c t)\right) \end{aligned}\right\} 0 \le t < N/f_c$$

$c_1^2(t) = 1$





$c_1(t) \perp c_2(t)$.

Here, $c_1(t)$ & $c_2(t)$ are just random fns. s.t.

$$\int_0^T c_1(t)\cdot c_2(t)\,dt = 0$$

U_i sends $s_i(t) = c_i(t) \times A\cos(2\pi f_c t)$

B.S. receives $r(t) = \sum_{j=1}^{n} s_i(t)$

So, if BS wants to figure out what the first user is sending:

Note that $\int_0^T s_i(t)\cdot c_j(t)\cos(2\pi f_c t) = 0$ when $i \neq j$. since $c_i \perp c_j$

& $$\int_0^T s_i(t)\cdot c_i(t)\cos(2\pi f_c t) = \int_0^T \frac{A}{2}(1+\cos 4\pi f_c t)\,dt = \frac{A\cdot T}{2}$$

use a low-pass filter to remove $2f_c$.

CDMA and OFDM are robust to multi-path.

↳ 3G ↳ wifi, 4G LTE.

Lec14

Switching

→ L2-switch

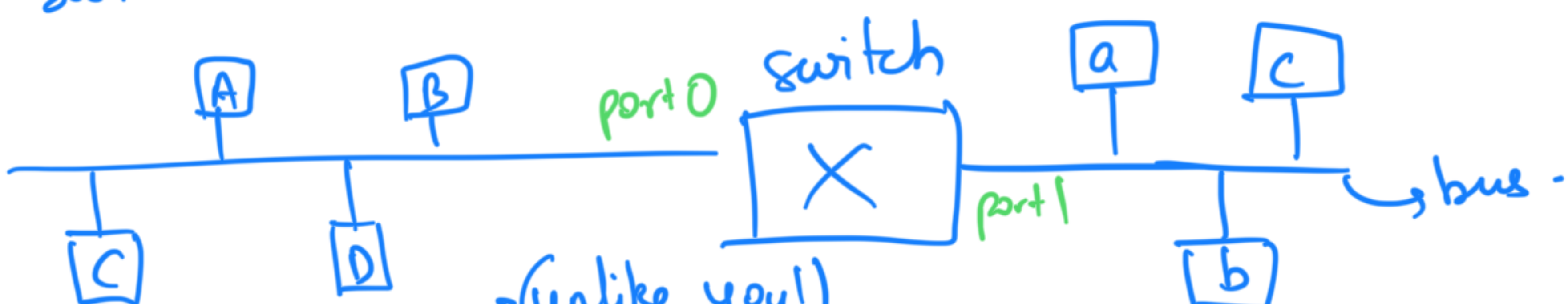
Layer-2 switches (ethernet) ⇒ use MAC addresses to switch.

Layer-3 switches (routers) ⇒ use IP addresses to switch.

↳ we care filhaal only abt this.

aka bridges
↑
(Ethernet switches)

A switch will connect 2 LANs together.



A switch is intelligent. (unlike you!) It only forwards message if intended receiver is on other side. Else it does nothing.

One way to do it is manually feed a "forwarding table" to the switch. (which node on which port.)

Problem: i) may be tough as many devices
ii) what if we unplug a device & switch port.

We need a way to automatically determine.

Initialize table:

Dest	Port #

Suppose A → B.

Dest	Port
A	0

switch knows where A is but dk abt B. So it forwards.

The switch learns.

Then B → A.

Dest	Port #
A	0
B	0

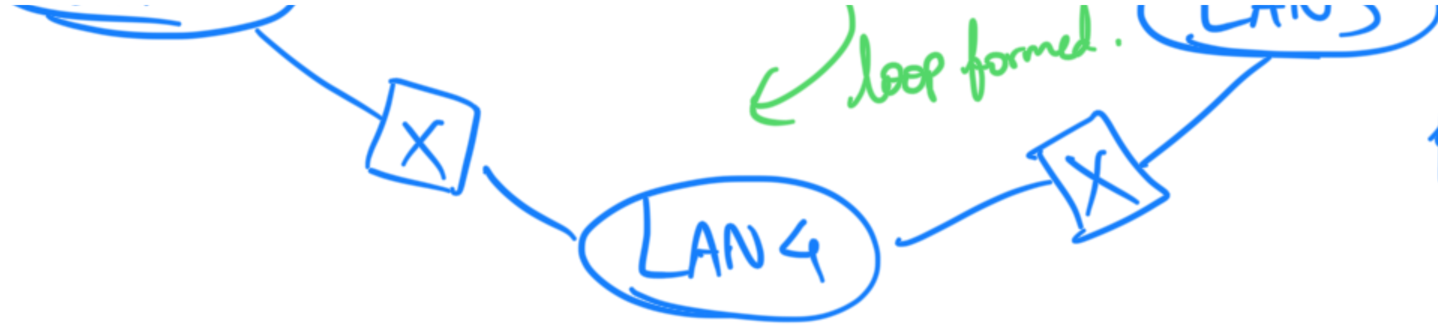
Each entry in table has an expiry time. Delete if it expires. Extend time if it keeps hearing.

Switch got info that B is on 0. Don't forward.

A switch can also have more than 2 ports. Use similar logic.

Multiple LANs can be connected with multiple switches.

It is possible that in the loop, data



keeps getting forwarded indefinitely.

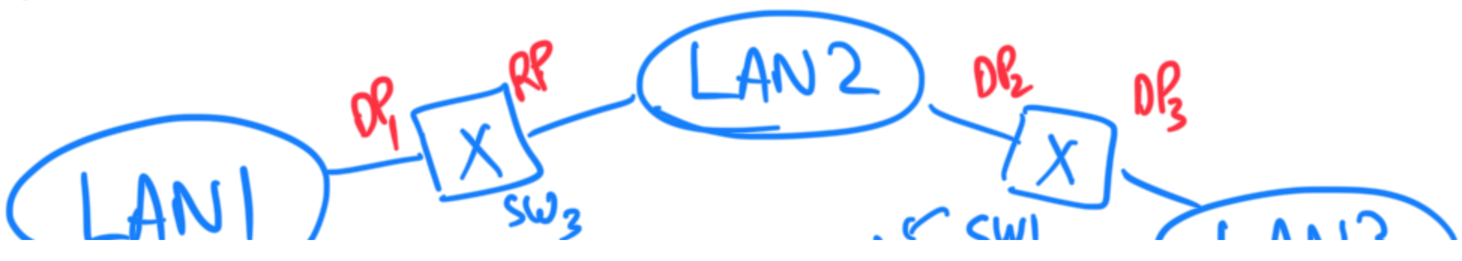
To prevent this; use spanning tree protocol (SPT) (disable some ports s.t. loops are removed)

i) elect root bridge

ii) each bridge finds which port is closest to root and assigns it as root port. (we need a tie-breaking rule for this) (RP)

iii) for each LAN, there may be multiple bridges to choose from to forward frame to root. Elect one of the ports from each LAN to be the designated port (DP)

Any port that is not DP or RP is disabled. → bc leafs don't forward.



DETAILS:

[reasons? mat poochna MC]

1) Elect Root.

Each bridge has a BRIDGE ID : $(0 - 61440)$

default is 32768
can select only multiples of 4096.

$\underbrace{(0-61440)}_{\text{Configurable Part (2 bytes)}}$ $\underbrace{\quad\quad\quad}_{\text{MAC Address (6 bytes)}}$

choose smallest MAC.

lowest becomes root

How to elect tho?

Each bridge tells it neighbours: (Y, d, X)

X → my ID

Y → smallest ID I heard till now

d → distance to Y.

eventually after a few iterations:

SW1: $(1,0,1)$; SW2: $(1,1,2)$; SW3: $(1,1,3)$; SW4: $(1,2,4)$

Y

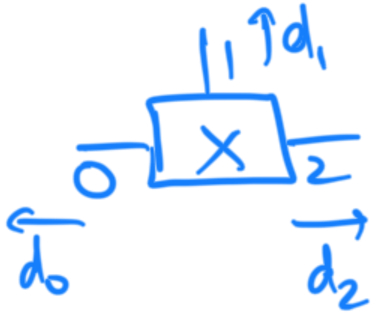
d ———— $(Y_2, d_2, 2)$

__Case I__: $Y < Y_2$ then $Y_2 = Y$; $d_2 = d + dist(X, 2)$

Case II: $Y = Y_2$ but $d + dist(X, Z) < d_2$

then $d_2 = d + dist(X, Z)$

2) Root Port



if more than port has same smallest distance; break ties on ID of neighbor of port. smaller id gets priority.

3) Designated Port

Same logic as above.

Sometimes the LANs have diff speeds. In that case instead of discrimination on basis of distance; assign "cost" to each LAN and discriminate on that basis. Higher speed ⇒ lower cost

Lec15

Layer-2 switching provided a way to connect two LANs. It prevented multiple collisions if the LAN was directly connected. Thanks to the switch's intelligence.

But, it is not possible to scale it to billions of devices.

Why? The spanning tree might not be optimal. Some ports are not being used so optimal paths may get discarded. Hence, poor resource utilization. Further, too many forwards. Forwarding Rate $\propto$ no. of hosts.

The forwarding table will be of size $O(N)$. Lookup also tough.

↳ flat addressing. → map from MAC address to port number.

↳ 2^{48} diff MAC addresses.

Even a small LAN may have a wide range of MAC addresses.

↳ Layer 3 switching uses IP addresses : hierarchical addressing

thus removing flat addressing problem.

Another issue is that stability. If root fails then SPT has to be reconstructed. If "hello" messages from root don't come → reconstruct SPT.

Same problem if any switch fails.

For larger LANs → reconstruct SPT more often.

There is no common addressing scheme or communication protocol globally making scalability of Ethernet Switching a pain in the ass.