MAC protocol: • suitable to: 1) REDUCE RADIO DC => to save energy (energy efficiency)
2) MAINTAIN NETWORK connectivity

• (RADIO) 2 WAYS (f sensor, processing)
-> T (Planning on Transmitter to reduce DC)
-> R (cannot do planning: msgs arrive asyncronously)
=> loose msgs if turn off the radio

• TRADE-OFF: energy vs (bandwidth & Latency) -> STRONG IMPACT on MESH net (like 802.15.4)

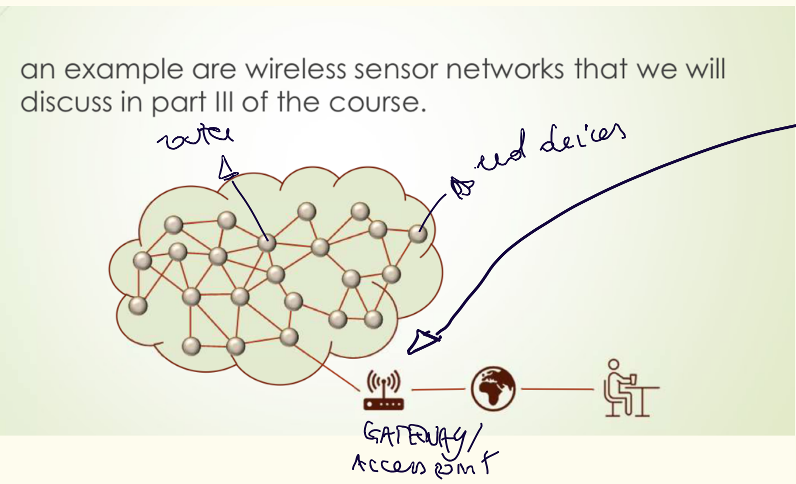
=> In mac 4 IOT, change objective: not only reduce collision,
(1) 4 energy eff => implement DC of radio
=> SYNC of diff nodes
=> to tell when is safe to turn off the radio
=> (2) Maintain net conn

• 3 approaches to energy efficiency

1) Explicit syncronization of nodes: implements a distributed protocol (S-MAC, 802.15.4)
=> devices agree on inactive period in the network
2) Implicit syncronization: using PREAMBLE SAMPLING (B-MAC)
3) Polling: asymmetry between nodes is strongly than 1 || 2

GUIDELINES: SYNCRONIZATION OF NODES
1) if devices are syncronized => can turn on RADIO simultaneously
2) when RADIOS are active => network is up
2) when RADIOS are inactive => no network

• Radios have low DC => inactive most of the time



• Network with large # devices
=> 1 (or few of them) get DOUBLE ROLE
• Gateway or AP to internet
_ others are at same time ROUTERS || END DEV.

• large multi-hop network
(in ZigBee up to 30'000 devices)
↳ syncronization protocol may delivers Troubles in that one kind.

S-MAC

- key! USES ONLY LOCAL SYNCRONIZATION

  (large multi-hop in wich devices, nodes implement syncronization only LOCALLY => exchanging sync packets among neighbours.

- CONSEQUENCE OF IMPLEMENT ONLY LOCAL SYNC:

  => device that are far away from each other, can be complitelly DESYNCRONIZED, lower exploiting locally sync magic reach/cover communications in big multi-hop.

IDEA: S-MAC SYNCRONIZATION

1) when a node is turned on:

   - listen over the radio if there are others S-MAC nodes around

2) if (there is around a node): (YES)

   A) => It will advertise its presence informing the potential neighbour about its OWN RADIO SCHEDULE

      ↳ specify: 1) PERIOD of RADIO'S ACTIVITY
      2) FRACTION OF TIME RADIO IS ACTIVE

   (=> es. tells you: every second and at the beginning of each second turn radio on for 15 ms)

   B) => SYNCRONIZE THE CLOCK => use the SAME SCHEDULE

else if (not hear nobody)

   C) => DECIDE YOUR OWN SCHEDULE

else (at certain point discover others nodes have ≠ schedule)

   D) => adapt their SCHEDULE

- NEXT:

  - Once you've your schedule & know your neighbour's schedule:

    => START WORKING => can TRANSMIT or Receive

    TRANSMIT ↓ In target node activity period

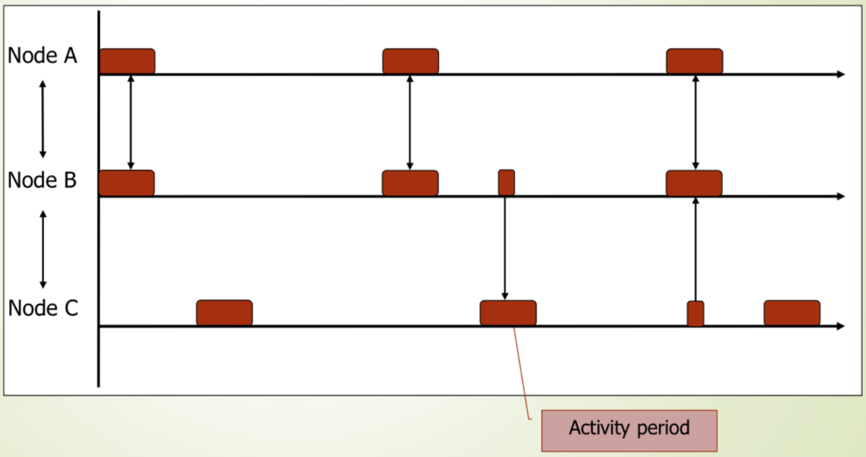
    Receive ↓ In your activity period

  - if (S/D activity period are same) => no problem

    else => turn on radio when destination node is active

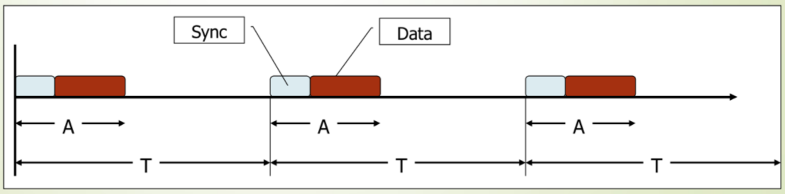
SYNCRONIZATION

Synchronization: S-MAC



• S-MAC avoid collision tecnique:

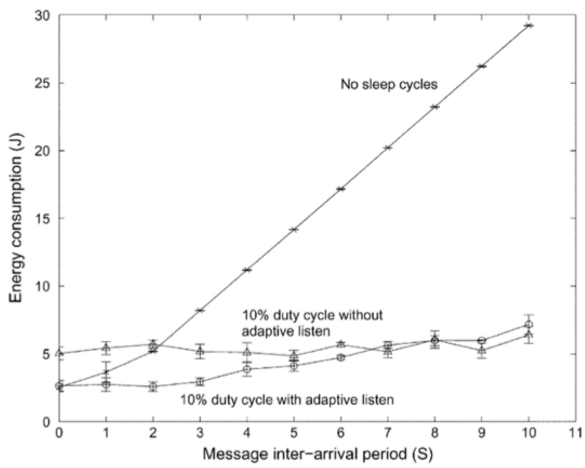
- avoided by RTS, CTS => 802.11
- period will be divided in SYNCRONIZATION & DATA TRANSMISSION



- frames are sent during the listen period of the receiver
  - Carrier sense before transmission
  - If the channel is busy and a node fails to get the medium, the frame is delayed to the next period
  - Collision avoidance based on RTS/CTS (as in IEEE 802.11)

aka NETWORK LIFETIME

Energy consumption: S-MAC



Aggregate energy consumption over a 10-hops path

* Adaptive duty cycle is a little optimization used by S-MAC: if a node overhears a RTS or a CTS it keeps its radio on until the end of the transmission just in case it is the next hop of the communication.

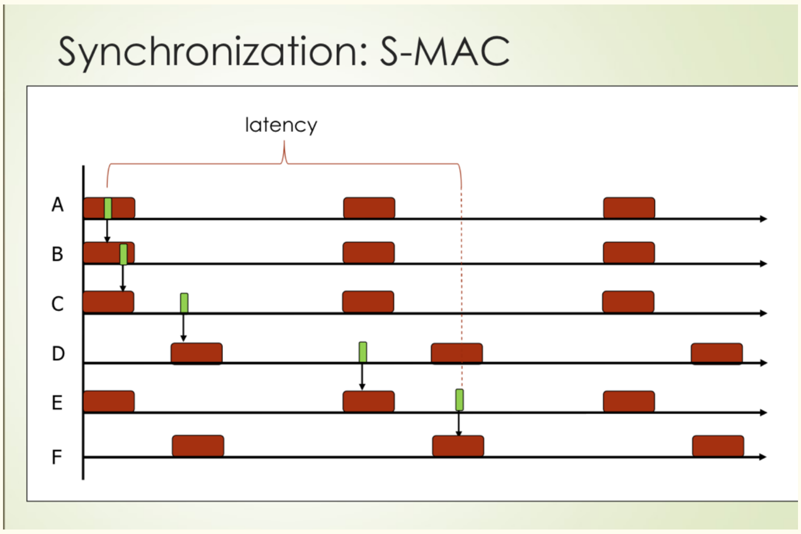
• Longer is inter-arrival period (s)
=> smaller is load on the network

NETWORK LATENCY: BIG PROBLEM OF S-MAC

• due to enabling SYNCHRONIZATION among the nodes

=> may have to wait if you have to forward packets to the next hop.

⇓

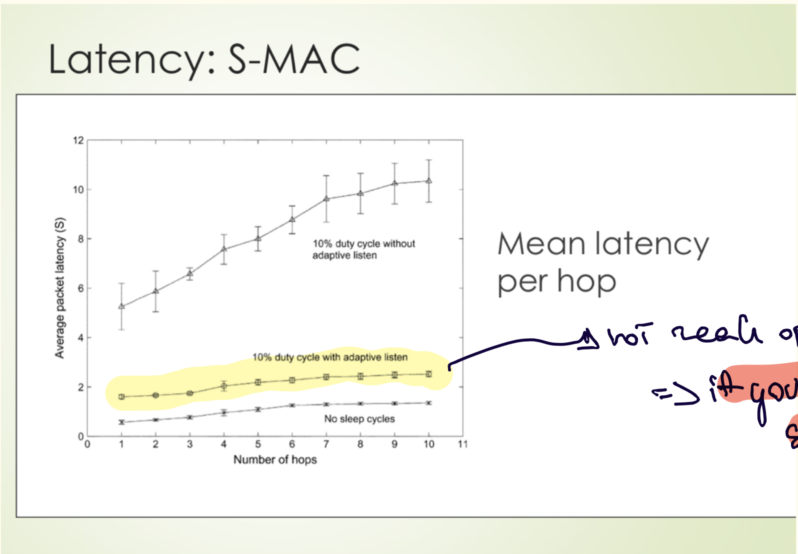


• nodes {A,..,F} are on chain along multi-hop path

• FIRST part of the PATH nodes are synchronized {A, B, C} => no delay

• message reach C, msg -> D cannot forward immediately, WAIT D's active period and so on -> F

=> CAN HAVE TRICKS:



-> not reach optimum but a compromise:

=> if you overhear a transmitted packet to someone else

=> turn RADIO ON => you may be the next hop

• MANTAIN SYNCHRONIZATION:

- In cheap devices, sync must be kept synchronizing the clocks
- However quality of clocks in IoT => shitty
- Clock is affected by ENVIRONMENTAL PARAMETERS (like temperature or shells)

  => makes diverge device's clock (drift)

=> FOR THIS REASON S-MAC includes an additional protocol to RE-SYNC the clock

(2) PREAMBLE SAMPLING (B-MAC): implicit synchronization

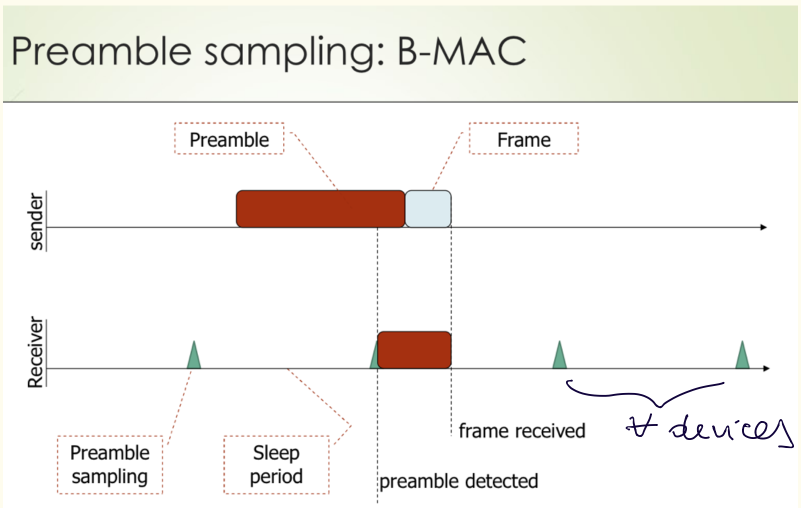
- give up: initial protocol to find the schedule & repeat periodically resync of the devices
- reduce protocol as much as possible to get SYNCR. implicitly by differentiating the behavior of the Receiver & Transmitter => B-MAC

1) Sender | can transmit whenever it wants (no effort to be sync with the receiver)

2) Receiver | perform a periodic activity => PREAMBLE SAMPLING (aka Low Power Listening, LPL)

3) PREAMBLE SAMPLING | activity on which time to time:
   1) turn on the radio
   2) listen to the channel
      if (don't find any transmission)
         => Turn-off the radio
      (Receiver will knows when a packet is transmitted
      else (R get a Transmission in the air)
         => turn-on the RADIO (to listen the transmission)

=> a) Spend more in Transmission, but save energy in reception
   b) Preamble should be short and cheap
   c) Cost of turn on/off the radio at the receiver are BALANCED BY a LOW RATE SAMPLING
   d) PREAMP: SHOULD BE LONGER THAN SLEEP PERIOD



- PROBLEM: if there's a Transmission and start to receive in the middle => NOT understand what is transmitted
- So => Transmitter start to T with a long preamble
  => to discover some packets will be send at the end of the period

- WORKS BECAUSE:
  duration → |preamble| > time interval between 2 sampling of the RADIO at receiver side

- NB: this protocol has JUST 1 PARAMETER
  => SLEEP PERIOD ∀ devices

- PREAMBLE SAMPLING NOT FREE: turn on the radio, listen => CONSTRAINED by underlying technologies (physical medium)

• SO

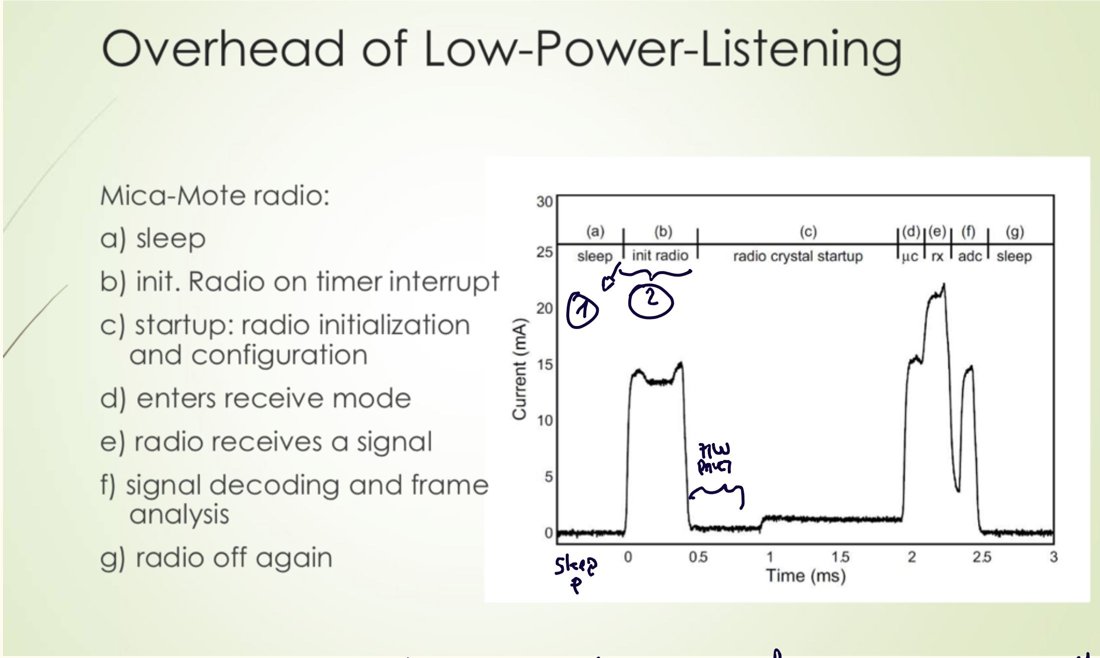
1) TURN-ON THE RADIO → CLOCK
2) Initialize the RADIO (HW part)
3) listening mode →

(≈ Analog/Digital conv.)

• From the signal, the radio must to discover each individual bit => needs to keep a syncronization with the signal (beginning/end of a bit sync)

⇓

to do, RADIO has an internal clock needed to be initialized (TRAIN, look at tree) must to see the leaves, branches...) (è il segnale pensa a whats della luce)



• Receiving consume more (activate other circuits

=> Turning on the radio keeps time and consume energy (don't want to do frequently

• BUILD an analytic model to access the lifetime of Transmitter & Receiver using B-MAC => estimation of DC

PROBLEM:

• DC in reception is simple: cuz there's the low-power listening, however, dealing with ASYNC transm.

NELLA TRASMISSIONE:

To model lifetime:

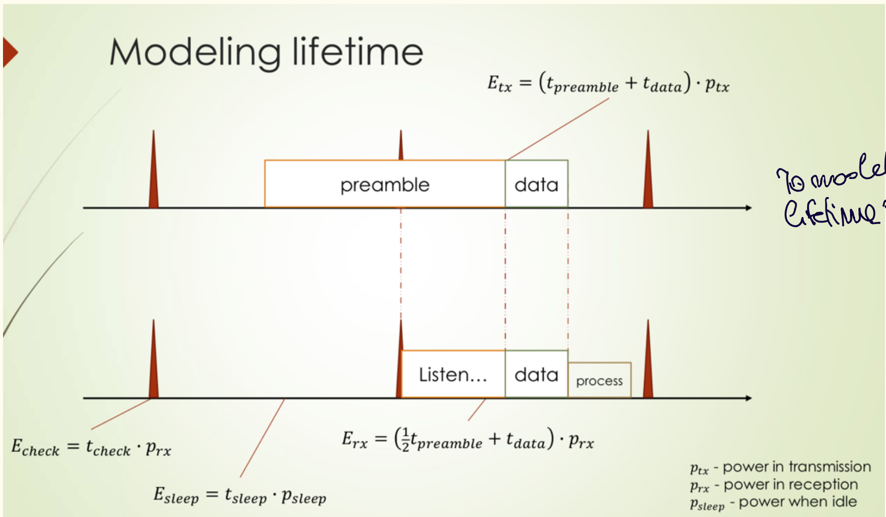
• Speed energy to T, R, LPL (preamble)
+ depend easy 4

DC →

• Energy for low-power listening:

• La durata dello LPL • CONSUMO ENERGETICO DELL'ATTIVAZIONE DELLA RADIO
E check

+ Energia spesa QUANDO LA RADIO È SPENTA (sleep mode) (E sleep)



DC
• TRANSMISSION

power to transmit (to keep radio in TRANSMIT MODE)

$$E_{TX} = (t_{preamble} + t_{data}) \cdot P_{TX}$$

time of transmission the preamble

time transmission of the data

DC
• RECEIVE

tempo per ricevere il dato

POWER IN RECEIVE MODE

$$E_{RX} = \left(\frac{1}{2} t_{preamble} + t_{data}\right) \cdot P_{RX}$$

=> on AVG. half the time to receive the preamble

$$E_{check} = t_{check} \cdot P_{RX}$$

duration of preamble

power to turn on RADIO

+ time while radio is off (Esleep)

• In order to model the LIFETIME => need assumption on how frequently they transmit & receive.

NON SAPPIAMO QUANDO UN DC, PUÒ AVVENIRE IN OGNI MOMENTO → ma dico in media T è fatta volte/secondo

=> IMPLICITAMENTE STO PRENDENDO la media TRA 2 pacchetti => così assumo DC in TRASMISSIONE

Modeling lifetime

Considero 2 fattori: stimo il DC per trasmettere/ricevere & DC preamble sampling & di conseguenza la frequenza del receiver

One transmitter and one receiver:

- f_{data} frequency of transmitted data (in hertz $\frac{1}{sec}$)
- f_{check} frequency of preamble sampling => PARAMETER OF B-MAC (QUANTO FREQUENTEMENTE FACCIO IL P.S. sampling)

Hence, at transmitter (only radio…):

- Duty cycle data: $DC_{tx} = f_{data} \cdot (t_{preamble} + t_{data})$
- Duty cycle check: $DC_{check} = f_{check} \cdot t_{check}$ → DC of preamble sampling
- Energy (in Joule) spent in t seconds:

$$ET(t) = t \cdot (p_{tx} \cdot DC_{tx} + p_{rx} \cdot DC_{check} + p_{sleep} \cdot (1 - DC_{tx} - DC_{check}))$$

Modeling lifetime

dell'evento

At receiver:

- Duty cycle data: $DC_{rec} = f_{data} \cdot (\frac{1}{2} t_{preamble} + t_{data})$
- Duty cycle check: $DC_{check} = f_{check} \cdot t_{check}$
- Energy (in Joule) spent in t seconds:

$$ER(t) = t \cdot (p_{rx} \cdot DC_{rec} + p_{rx} \cdot DC_{check} + p_{sleep} \cdot (1 - DC_{rec} - DC_{check}))$$

Thus, the lifetime:

- Transmitter: $lifetime = battery_{charge}/ET(1)$ → energy consumption in 1 sec
- Receiver: $lifetime = battery_{charge}/ER(1)$

• transmission can happen at every time, not a proper DC
(but if we say it happens on AVG f_{data} times per sec
=> implicitly I'm take avg of inter-arrive of 2 packets and assuming DC in transmission

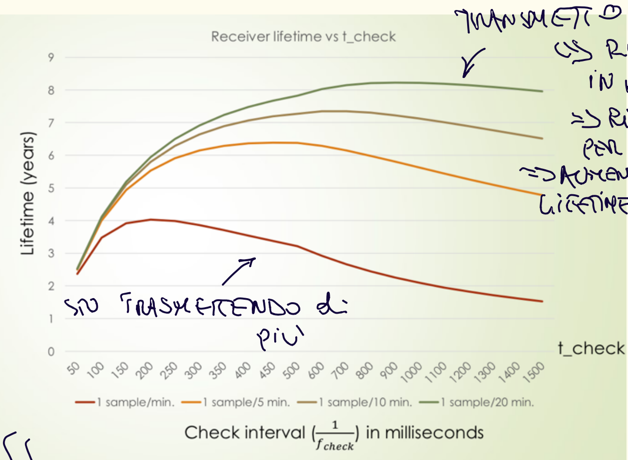
$DC = \frac{\text{activity time}}{\text{period}}$ frequency = period^{-1} (inverse of the period)

=> have to multiply by the frequency

Modeling lifetime

Lifetime of the receiver

- Disregards:
  - Processing, sensing & battery leaks
  - (only transmission)
- Battery of 3000 mAh



TRASMETTO MENO → RIDUCO DC IN RECEPTION => RIDUCO ENERGIA PER RICEVERE => AUMENTO LA LIFETIME

Durata della LPL

STO TRASMETTENDO di più

SE TRASMETTO di PIÙ LA LIFETIME SARÀ di MENO

• CURVES SHOW ENERGY CONSUMPTION at the RECEIVER for ≠ values of interval t_{check} and ≠ loads of the network

each curve, ≠ load of network

=> ogni curva rappresenta un different load della rete

→ transmitting more

if
• TRANSMIT LESS
=> reduce DC in RECEPTION and reduce energy to receive
=> increase LIFETIME

• TRANSMIT MORE
=> decrease LIFETIME

• curves are quite SIMILI, C'È UN TRADE-OFF

• => $\boxed{\text{TRADE OFF}}$: related to the size of checking interval (LPP)

=> if (checking interval is very small)
=> energy consumption dominated by CHECKING of the channel

if (checking interval is very large)
=> energy consumption dominated by the RECEIVAL of very long preambles

=> there's in between an optimal value: OPTIMAL TIME CHECK for wich the energy that you spend to implement preamble sampling and energy spended to receive long preambles, finds an equilibrium, (according to the load of the network)
(this for receiver, same 4 TRANSMITTER) ⓐ

TRADE OFF

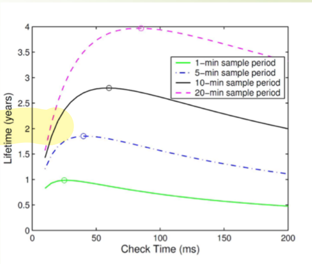
Δ if (perform preamble sampling frequently)
=> energy dominated by preamble sampling
if (enlarge the check interval) => reduce energy to preamble sampling
but for T to transmit very long preamble

=> Behavior with 1 T and 1 R

Nodes lifetime with B-MAC

NOT IMPORTANT

- Even considering several transmitters the lifetime trend does not change
- Node lifetime depends on:
  - the check interval (of preamble sampling)
  - the amount of traffic in the network cell.
- Each line shows the lifetime of a node at that sample rate and LPL check interval.
- The circles occur at the maximum lifetime (optimal check interval) for each sample rate



ⓐ SE TRANSMETTO DI PIÙ l'equilibrio è intorno 100 ms of TIME check, se trasmetto meno
=> equilibrio = 500 ms

Preamble sampling: B-MAC

- Strengths:
  - it is not a network organization protocol
  - it is simple to use and configure (only one parameter!) (DON'T REQ. SYNC, IS implicit)
  - it is transparent to the higher layers
- Potential drawbacks:
  - long check intervals imply long, expensive preambles
  - in the long run preamble sampling is not negligible
  - in some cases it may result more expensive than using some form of synchronization

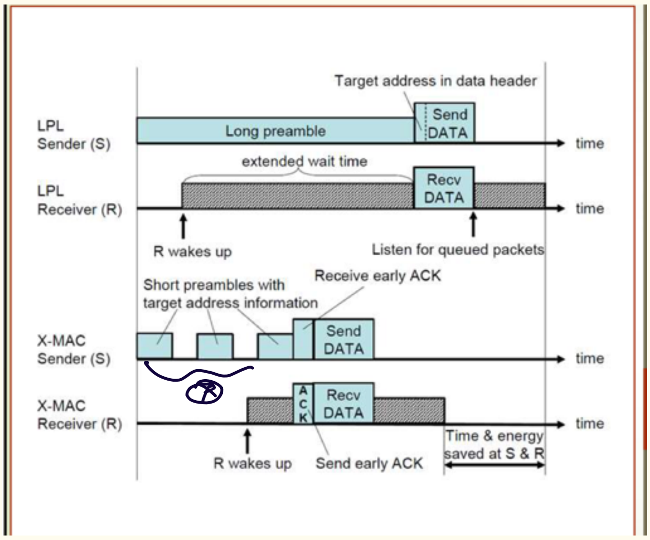
1 PROBLEM: all devices in the network should perform preamble sampling with same freq. => don't need clock synchronization, but need clock measure time in the same way

X-MAC : try to reduce very long preambles of BMAC

- Trasmit a preamble (con l'indirizzo del destinatario) from time to time, allowing the receiver to send back an acknowledge (ACK)

=> Receiver can stop the transmission of preamble in advance, inform to the transmitter he is active and Transmitter can send data immediatly (without waiting the rest of the preamble

X-MAC vs B-MAC



(*) are not simple preambles, but indicates the Recepient of your message, cuz you'll wait an ack from he.

- (*) 'I've a packet for node 10'
  "
  "

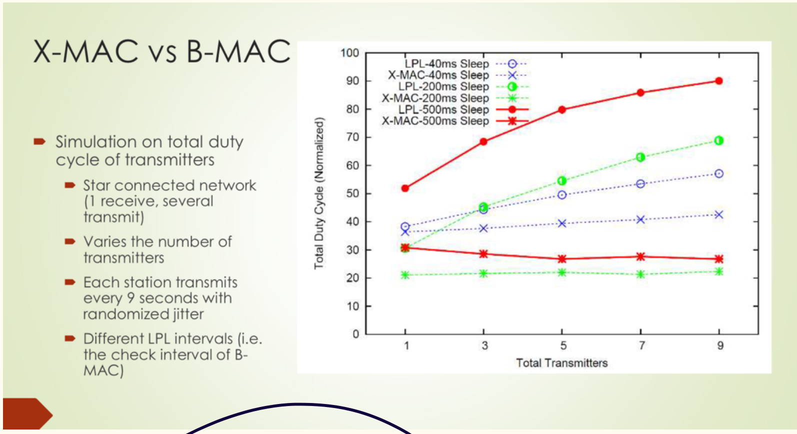
- node 10 send back an ack (in the interval of the transmitter

- ONE THING COMMON TO B-MAC & S-MAC:

- when you receive the frame, eventually, you will not turn off radio immidiatly but keep radio on for extra time:
  - -> cuz if there's a long preamble (B-mac) is possible someone else want to transmit but channel is busy (CSMA is still be there)
  - -> if all nodes, after receiving a frame turn off the radio
    - => I'll have to transmit the entire preamble before they can receive it again

  => cuz is probable that you transmit long P and keep radio busy for a long time, someone else wants to transmit on the receivers, after the transmission keep the radio on just to see if there's another packet that is transmitted

  ⇓ optimization to append to BMAC / S-MAC



- if check_interval is longer
  => devices will be busy for very long P in transmission (BMAC)
  ⇓
  that's where X-MAC has the best improvement

BOX-MAC: improvement!

- Preamble: instead of "I've a message to node 10" is the message itself => cut in lot messages are short

=> Receiver: just acknowledge the message itself

POLLING

Polling

- Used by Bluetooth & IEEE 802.15.4 (along with other methods)
- Can be combined with synchronization
- Asymmetric organization of the nodes:
  - one master node issues periodic beacons
  - several slave nodes that can keep the radio off whenever they want.
- If the master receives a message for a slave:
  - It stores the message and advertises the message in the beacon
  - When the slave turns on the radio:
    - waits for the beacon
    - recognizes that there is a pending message
    - requests the pending message to the master

- Just give for granted that nodes/roles are different, one node coordinates the others, others can save energy cuz are coordinated
- typically in STAR topology:
  - center is the coordinator (more energy)
  - other nodes at the edge of the star save energy (their radio is not ON all the time)

Figa Rotta

ciao

Sassino

[illegible]

Ciao

Porco Dio cane

(1) Expl synch: distributed sync.

node exchange infos and ~~tak~~

- implications in multihop context: strong impact on latency of convergence

⇓

not unit large ~~#~~ of dev.

⇓

1 takes role router/end dev.

~~e~~ ~~go~~

⇒ problem latency of sync. protocol

SMAC: key: uses local synch.

⇒ like By node: sync packet among their neighbour

⇓

2 node ~~far~~ away. may be not synchronize

- A node turn on 1st time: listen over radio for other ~~~~ S-MAC node working around ⇒ advertise all potential neigh the schedule of radio

period & fraction of time is active

⇒ synchronize clocks

⇒ listen to your neigh you take his schedule (just wait period of activity and transmit)

- if 2 dev. have ≠ schedule ⇒ pick just one of them

⇒ if don't hear nothing ⇒ set your schedule

if > of neigh use a schedule ≠ yours ⇒ adapt to the other's schedule ~~sched~~

• once decide schedule => can work = T Q R

↓ in source activity period

↓ in the activity period

if periods ≠ => turn on radio for target periods

• on sync, is rare => set up channel to avoid collision => RTS, CTS

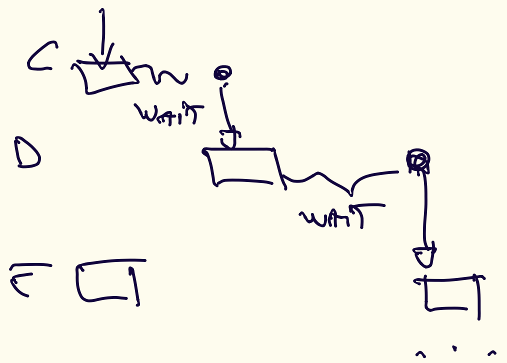
=> period / by synch.

ENERGY CONSUMPTION

> is msg inter-arrival period => < load on network

LATENCY (prob. of same)

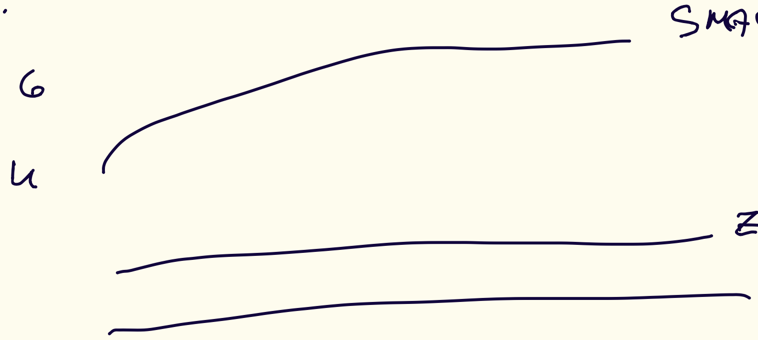
due to sync to wake => wait if you've send packet to next hop



=> WAIT => latency

• TRICKS:

SMAC (has sleep cycle)



z can see it! if hear a packet from to same edge

=> turn radio on if you're the next hop

=> can cut down a bit the latency

OMAINTAIN SYNC: keep by sync the clocks

↳ quality of clocks in IoT not really good

② clock affected by environ. param (temperature)
⇒ clocks of ≠ dev. can divergence

? ⇒ money to have cool
⇒ IoT cheap clock

Δ is significant

- using a dev with/without shell
⇒ variation of temperature impact on clocks
and on SMAC, ⇒ after a while (minutes) device diverges according to their clock variation
⇒ need to RE-SYNC the devices

---

② Preamble sampling: B-MAC

- no complex prot: like SMAC → schedule + resync periodically
⇒ reduce prot and obtain sync by differentiate the behavior send/rec
  - sender: T when he wants
  - REC. perform activity (Preamble sampling) (low power listening)
    ↳ from time to time, Turn on RADIO
    ⇒ low power list
    if no traffic ⇒ TURN OFF and WAIT for rex period (periodically will turn on)
- if rec. detect T on Air: keeps radio on
  - keep radio on until is performed T

- sender send data with very long preamble to know after R that will see a packet after preamble
(R will keep listening until T preamble end)
- IT works cuz |preamble| > time interval between 2 sampling
sync of information
T/R ⇒ R will catch the preamble

• if node same time T & R
will implement low power sleep
sent to T => period + packet
=> give more work to T than the receiver

• => sleep period will be same 4 all receivers (parameter of the protocol)
• τ parameter for the protocol

• preamble sampling not free, constraint by property of physical radio:
→ turn on
① → initialize it (RADIO) of HW
→ ~~use it~~ wait to stabilize radio's clock
↓
sync collects n
know by 1/0

② after sync
=> enter mode (R)
=> digital converter
=> sleep again

Model lifetime: assumption on how frequently T & R

f check: parameter of B-MAC of preamble sampling

• For T not proper DC, can be every time => need interval
packet sending
2 packets
ensure there's a DC in T

($f_{check} \cdot t_{turn}$): time to implement the check
DC check
→ freq of check

$E(\tau)$ energy you spend at T side
with R same but consider
$\frac{1}{2}$ of preamble size

~~E(1)~~ En(1) → in 1 sec

o Tradeoff: relate to | check interval |

↳ if small energy was dominated by preamble sampling

> reduce energy more by long preamble ~~sampling~~ sampling, but energy to transmit long preambles

=> optimal time check

---

snapshot: AT R side

all nodes ∈ net all sample of receive freq

=> read income time (sec) to all devices ~~(not checks)~~

req. sync of clock is small

=> enlarge preamble aligned with lower clock skew in ~~your~~ network

o build analytic model to optimize BMAC

o long preamble => decrease throughput of the net (wait for long preamble)

=> but ok in IoT since devices have not to T so much

=> if T rate is low => good BMAC

=> problematic if net is complex T a lot

X-MAC : TX send a packet say it have got preamble and RX can receive before end of the preamble (ack)

- ack Transm is few bytes

BMAC/SMAC : when you rx frame keep radio on with an extra time: if long P ~~want to~~ enter node wait to transmit (channel is busy)
(just to see if there's another preamble to listen)

optimization of both that can be performed

X-MAC vs B-MAC: LPL is BMAC compared with same check interval with different Ts

- BW lines with o : BMAC
  x : SMAC

=> reduction of DC (SMAC)
is larger if check interval is larger

dev. will be busy with BMAC, SMAC improve this fact

Box-MAC : X-MAC evolution
this time, rather than in preamble say [the msg for node 10]
=> the ~~frame~~ P is the message itself
=> cut got few bytes of the message (not long ≠ with (2))

just ack of the message transmitted by the receiver to transm

@ Z the message for node 10 and msg is here (ends when ack is received)

S : interframe space : gives priority to packet T on multi hop channels

POLLING Bluet or 802.4 its combo with SYN

↳ nodes are ≠

↳ roles ≠ → 1 coordinate → STAR CONNECTION (center)

↳ other save energy (edge of the star)

(don't keep SYN / DC (macro) always ON

IEEE 802.15.04 → std to build sensor network, ref sims to IoT

⇓ save energy

create to low power network

– ≠ bluetooth: enge multi-hop net

in Bluet. net is small

low tress bluetooth energy save

(connection 2 dev. close each oth)

Born first in ZigBee net → given in 15.4

IEEE 15.4 – infra-less (no BS)

– range: 200–500 mt open space / 20 mt in rooms

– operate on same freq. of 802.11 / Bluetooth (can live together)

– 3 bands: 1)

3) 802.11 also uses (sect) 2400–2483,5 MHz

extra rete

Physical layer

MNG
- ~~Dst~~ sensing: detecting every level in a channel (noise present on ch, consider busy when sensor is T)

LQI: choose a channel with good quality (like zigbee)

- The case of 2.4GHz: freq. band divided in ~~16~~ 16 channels each of them a data rate up to 5MHz

15.04 design to operate better even in noise env.

=> high SNR => better quality of channel

=> ~~...~~ low SNR => channel has more noise => ~~ex~~ expect higher T error (BER)

=> high SNR => low BER)

=> MORE ROBUST TO NOISE

- ~~Physical layer~~

Physical layer frame

- preamble is very small (7 bytes): transmission encoding to be R synchronized
  tells R
- SFD: frame is begin
- 7 start of frame edge
- payload up to 127 bits => not use all, long frames, not ~~...~~ get problem with collisions

MAC LAYER: - data service to T mac level here
- manag. serv.: sync, channel access
- time slot to communication that require a strict sync?

- Feature of 15.4: ~~???~~ define 2 dev. type correspond to ≠ roles in the net.:
1) full funct dev: implement full MAC protocol
2) reduced . . . " only minimal part of the Prot

roles are mapped like node in ZigBee (coord, router, end dev)
full, full, mac layer reduced f. dev

• In star topo or eg. router role act as low f. dev

1) Implements: ass/disassociations of devices (coord)
- synchron. of reduce f. dev

TOPOLOGY MAC LAYER: ① STAR ② MESH (or P2P)

- but 3 top. of ZigBee can be mapped in (3rd only at net layer, mac layer see JUST neighbor)

① 1) center: coord (full f. de)
2) other dev ⇒ edges (all behave like low f. dev. (also if are router (≠ ZigBee))

② P2P
1) support mesh or tree of ZigBee
coord (zig) / router ⇒ full
END DEV ⇒ low ⇒ attached to routers (lf. dev)

• so even if ~~[illegible]~~ full dev is located in SFAN
=> likely reduce ~~[illegible]~~ F. dev

---

channel access: implemented by explicit split of nodes by coor (SFAN top)

~ coord => periodic messages ~~[illegible]~~ => BEACON (?) super frames structure in the network

(as coordinator info dev in net when T start transmitting a beacon)

• without superframe structure => 15.04 unslotted (we use) unslotted

some energy is up to you => decision tree above the MAC layer

• Beacon: provides all param. struct

=>