Sensors

Sensors could be used in several scenarios: in ecosystem monitoring they can also be attached to animals and, therefore, move. Data, in this case, can be collected through:

- infrastructure;
- mobile collectors (so the scenario is completely mobile).

The protocol forecast that:

- the physical layer implements some signal processing algorithm;
- the MAC layer provides nodes synchronization and localization (very important because to reconstruct the signal in reliable way, a time reference and also knowledge about where the sensor picked up the sample are needed);
- the *layer 3* implements routing algorithms for the topology: indeed, when possible, sensors are turned off to save energy, so they are inactive and do not contribute to form the topology; it means that the topology changes over time.

Although, to reconstruct the signal, Nyquist's theorem requires a regular sampling, it is not possible starting from measure collected by sensors because the sampling is not regular. To deal with this problem, usually, more sensors than required are deployed.

MAC solutions

Properties of MAC protocols for sensors networks are:

- collision avoidance;
- energy efficiency;
- scalability and adaptivity;
- channel utilization (backoff is an useless waste of time resources in this terms);
- latency;
- throughput (depends on the application: voice, data, images);
- fairness: nodes closed to the sink can use less power to deliver data with respect to farthest nodes, but they have to deal with more traffic because farthest nodes use them as relay nodes (*energy hole* problem).

In general, nodes waste energy when they listen to the channel in an useless manner, due to:

. overhearing;

. collisions;

. control overhead;

. idle time when events to be sampled do not occur.

To reduce those issues, it is possible implement TDMA (that requires time synchronization among nodes and slot allocation) plus contention protocols.
MAC protocols can be classified into:

. schedule based (TDMA, FDMA, CDMA);

. contention based (CSMA).

An example of scheduled based protocol is LEACH (Low Energy Adaptive Clustering Hierarchy). It uses clusters with star topology as Bluetooth and:

. TDMA within clusters;

. CDMA among clusters (not easy to implement: needs distributed codes).

Since head clusters consume a lot of resources (because only them are nodes entitled to communicate with the base), within each cluster, there is the so called *cluster head rotation*. This protocol suffers of scalability issues.

Contention based - SMAC

This protocol tries to trade between *energy saving* and *latency* and *fairness*. Major components are:

. coordinated sleeping;

. collision avoidance;

. overhearing avoidance.

Coordinated sleeping Since idle listening consumes energy useless, sensors periodically listen and, for the rest of the time sleep (duty cycle). Nodes have different schedule (active and off periods) that share with their neighbors.
Regions that overlap, have nodes in between different schedules:

. if they adopt one schedule, the network may result partitioned;

. if they adopt both schedules, they have to be active all the time.

Each node is entitled to broadcast its schedule every few periods of listening and sleeping: new nodes tries to adopt already existent schedules. In general, after an update, a re-synchronization phase takes place.

To optimize the coordinated sleeping, it is possible introduce the *adaptive listening*: it reduces the multihop latency due to sleep periods by waking up a node, for a short period, when it listens a CTS.

Collision avoidance It is realized in a similar way as 802.11 (DCF), but, here, RTS and CTS are mandatory.

Overhearing avoidance To avoid receiving packets not for it, a node should keep sleep when other communicate. Actually, only immediate neighbors of sources and destinations could do it, for a period specified in the *duration* field of the packet.

Contention based - BMAC

Basic features are:

. CSMA without RTS/CTS;

. optimal ACK;

. Low Power Listening (optional, but in practise always implemented).

CCA procedure It is a procedure that measures the power level of the channel and compare it with the estimated noise power level.

Before transmitting, a node extracts a random backoff, then sample the power level: if it is below the noise floor, the channel is assumed to be free, so the transmission starts immediately. In the other case, five samples, are taken: if no outliers are found, the channel is busy otherwise it is idle.

LPL procedure To minimize listen costs, nodes wake up at check time and perform CCA: if the channel is busy, they power on to receive the packet then go to sleep; if the channel was idle, after a timeout they go to sleep.

If nodes sleep too much, the danger is that sources and destinations do not meet; to deal with this fact, the preamble is set long as the check time, but it consumes more resources to detect it, so this protocols is suitable for small load.

802.15.4

Devices can be classified accordingly to:

- FFD (Full function devices): beside implement physical and MAC layer, they also have routing functionalities;
- RFD (Reduced function devices): they only implement physical and MAC layer.

Topologies can be:

- star (the center is called PAN, Personal Area Network coordinator: it needs to be FFD, while other nodes are RFD);
- P2P (all devices are FFD);
- cluster-tree (combination of the previous two: to perform security needs a coordinator, that is the PAN).

The physical layer uses, as frequencies bandwidth, spread spectrum and modulations:

- 16 channels at 2.4 GHz, DSSS (Direct Sequence Spread Spectrum) with 32 chip code and O-QPSK (constellations with 16 points) up to 2 Mchip/s, 250 kbit/s;
- 10 channels at 915 MHz, DSSS with 15 chip code and BPSK up to 600 Kchip/s, 40 kbit/s;
- 1 channel at 868 MHz, DSSS with 15 chip code and BPSK up to 300 Kchip/s, 20 kbit/s.

As usual, higher frequencies allow to cover a small area with respect to lower frequencies and, as 802.11, it is possible to combine together two channels to reach higher bit rates (up to 500 kbit/s).

The MAC preamble takes 7 Bytes:

- 1 to specify the addressing mode;
- 4 to specify the address accordingly to the previous field;
- 2 to specify the data sequence number.

By using the *beacon-enabled-mode*:

- superframes are composed of 16 slots: the first is reserved for the beacon;
- the coordinator, PAN, sometimes broadcast the beacon to synchronized nodes;

- time is divided into slots called *backoff periods* (not related to 802.11's backoff).

Superframes are furthermore divided into:

- beacon: first slot;
- contention access period (CAP): CSMA-CA to transmit, but no RTS/CTS and optional ACK;
- contention free period (CFP): similar to TDMA, slots are assigned by the PAN to nodes.

Slotted CSMA-CA Algorithm

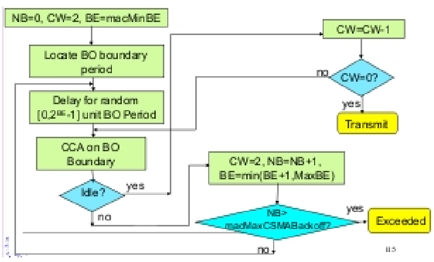
Variables

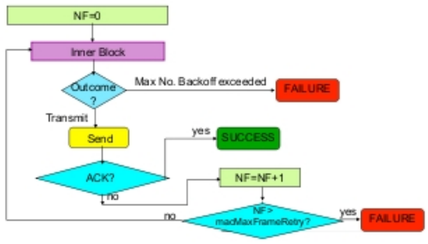
- **NB**: number of times the CSMA-CA was required to backoff while attempting a current transmission;
- **CW**: contention window length, which defines the number of backoff periods that needs to be clear of activity before a transmission can start (as before not related to 802.11's contention window);
- **BE**: backoff exponent, which is related to how many backoff periods a device shall wait before attempting to assess the channel;
- **NF**: number of times a packet has been transmitted.

Procedure

- As in BMAC, a node that wants to transmit, extracts a random number of backoff period that it has to wait;
- after having wait, it has to sense the channel for CW periods and, if the channel is idle, it can transmit in the next available backoff period otherwise, if the channel is busy, it has to wait for another random number of backoff periods;
- ACKs are sent, as in 802.11, after SIFS without using CSMA-CA;
- unlike 802.11, the backoff is always decremented, even if the channel is busy: no freeze procedure.

Diagrams





Failures Packets can be discarded for two reasons:

- when the maximum number of attempts to transmit expire, $macMaxCSMABackoff = 4$;
- when the transmission fails (no ACKs received): $macMaxFrameRetry = 3$.

To save energy, the PAN, can impose a sleep period during which no one can receive or transmits packets.

Unslotted CSMA-CA Algorithm

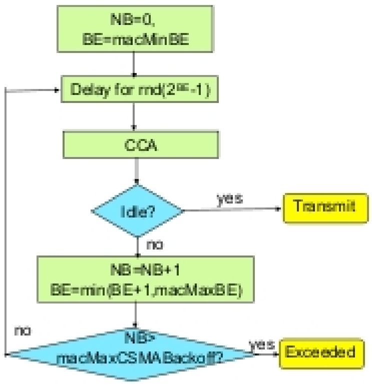
Variables

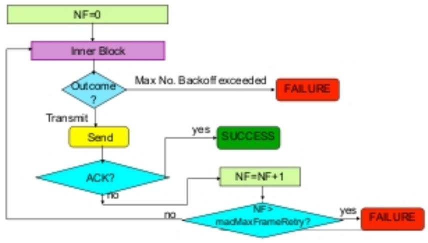
- **NB**: number of times the CSMA-CA was required to backoff while attempting a current transmission;
- **BE**: backoff exponent, which is related to how many backoff periods a device shall wait before attempting to assess the channel;
- **NF**: number of times a packet has been transmitted.

Procedure

- A node that wants to transmit, has to wait for a random number of backoff periods;
- after that, it has to perform CCA;
- if the channel is idle, it can transmit;
- if the channel is busy, it has to extract another random number that represents the number of backoff periods that it as to wait again.

Diagrams





Observations

- The sink can receives packets only if they are separated by a SIFS time: this is the time needed to *process* packets.
- For every attempt of transmission (NF < *macMaxFrameRetry*), enter in the inner block implies that NB is set to 0, therefore every attempt has the same number of chances to be transmitted.