# Wireless Sensor Networks

Paolo Pagano (SSSA, Pisa)

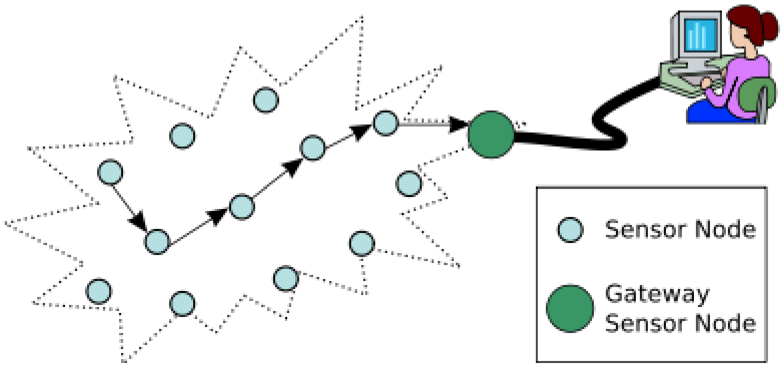
Buzzwords:

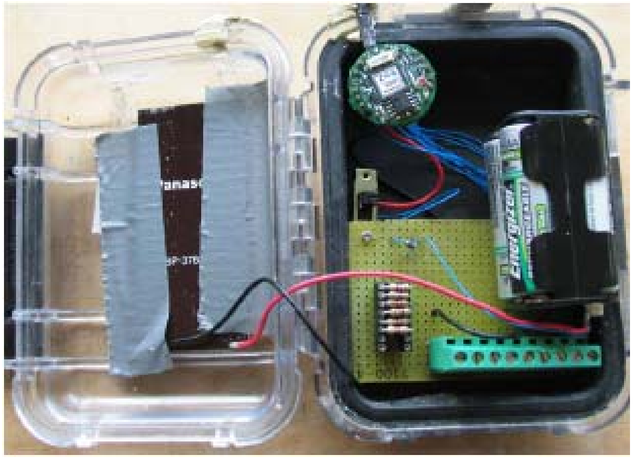
- ubiquity
- pervasiveness
- Wireless
- mobility
- smart spaces
- M2M
- distributed
- embedded
- dynamic
- energy


Thanks to André Cunha and Màrio Alves

# What is a WSN?

- WSN = [Wireless Sensor Networks (Wikipedia)]
- Set of nodes:
  - Sampling data from the environment;
  - Communicating wireless;
  - Autonomously powered (usually by batteries);
  - Low cost:
    - to deploy large number of units;
    - no infrastructure (cabling for data transmission and power).

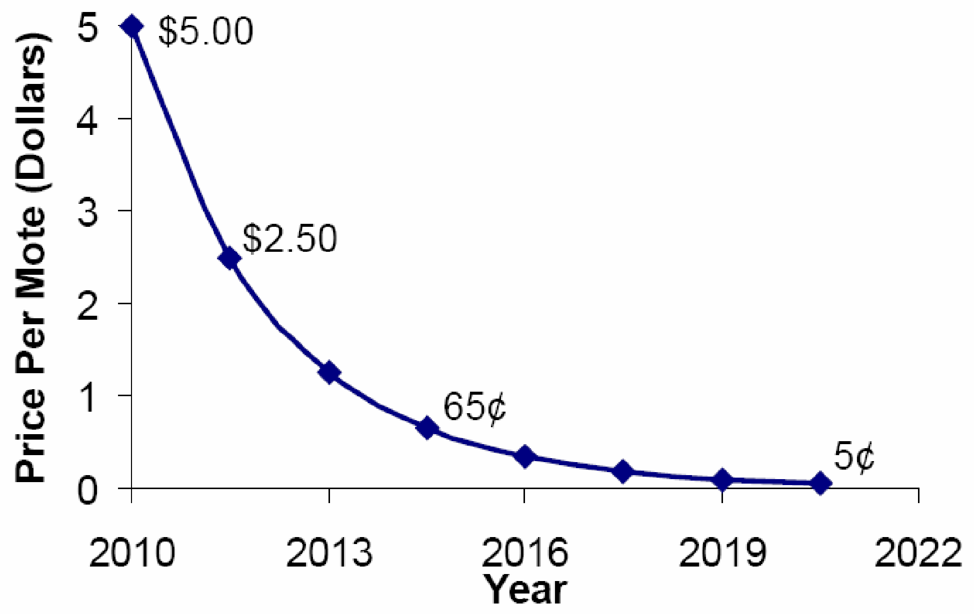


Thanks to Giuseppe Lipari for his contribution on these slides

# Historical background

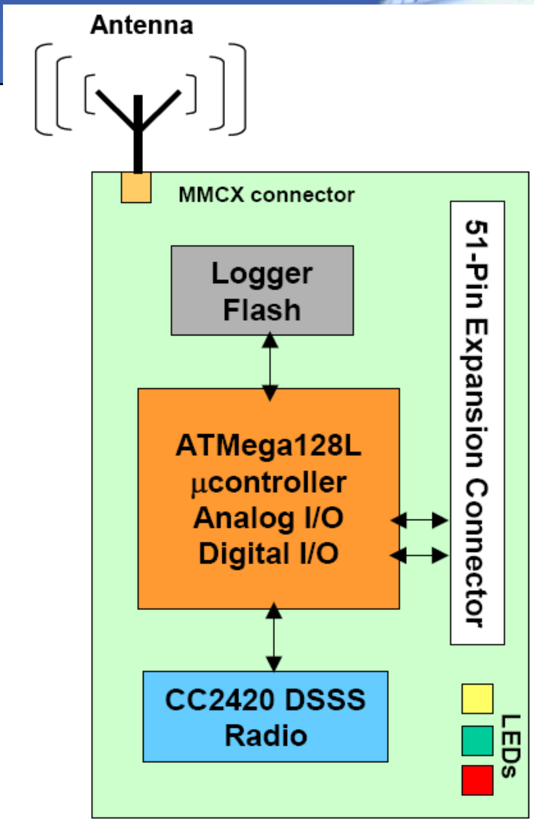
- A research initiated at UC Berkeley by K. Pister and ended in 2001:
  - Smart Dust Home Page
- The research was pushed by military applications such as monitoring of large “hostile” and unstructured areas;
- Now applied to civilian contexts:
  - Environmental monitoring:
    - Ecosystems like forests, seas, etc.;
    - Prevention (firing, contamination, etc.).
  - Structural monitoring in seismic areas;
  - New Areas:
    - telemedicine, health care, anti-intrusion, crime detection ,etc.




- The main idea is to disseminate agricultural fields, buildings, industrial plants by smart (programmable and self configuring) controllers.
  - Following the Moore law such devices will cost 5 ¢ in 2020;
  - a 1 M motes Network will cost 50,000 $ = 33,000€ (as a car today).

# Device equipments

- The devices are equipped by:
  - Processor (4-40 MHz, 8-32 bits arch., 4KB-... RAM and typical 128KB Flash);
  - Radio (250 Kbps maximum);
  - Sensor modules.
- The strongest constraints are:
  - power consumption;
  - memory capacity;
  - speed;
  - network bandwidth.




WeC 1/00

Rene 11/00

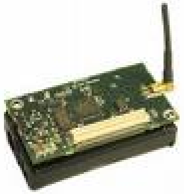
Mica 1/02

Mica2 9/02

Mica2dot 9/02

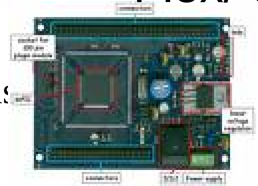
SPEC 5/03

Mica-Z/ 05



Telos-B/ 05

Flex/ 07

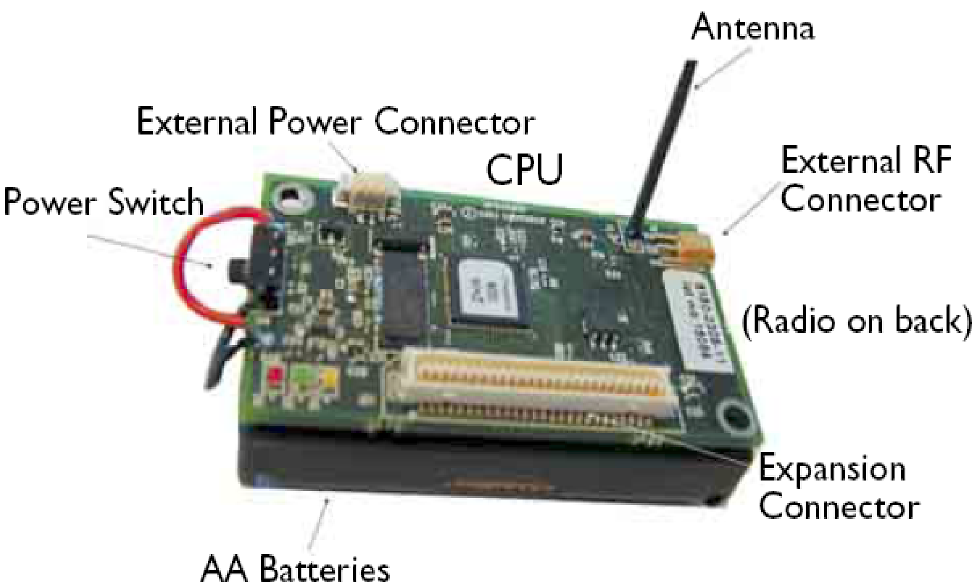


Imote2/ 07

# Node architecture (Mica-Z)

**Table 1-2.** *Mote Product Summary*

| Mote Hardware Platform | | IRIS | MICAz | MICA2 | MICA2DOT |
|---|---|---|---|---|---|
| Models (as of April 2005) | | XM2110 | MPR2400 | MPR400/410/420 | MPR500/510/520 |
| MCU | **Chip** | ATMega1281 | ATMega128L | | |
| | **Type** | 7.37 MHz, 8 bit | | | 4 MHz, 8 bit |
| | **Program Memory (kB)** | 128 | | | |
| | **SRAM (kB)** | 8 | 4 | | |
| Sensor Board Interface | **Type** | 51 pin | | | 18 pin |
| | **10-Bit ADC** | 7, 0 V to 3 V input | | | 6, 0 V to 3 V input |
| | UART | 2 | | | 1 |
| | Other interfaces | DIO, I2C | | | DIO |
| **RF Transceiver (Radio)** | Chip | RF230 | CC2420 | CC1000 | |
| | Radio Frequency (MHz) | 2400 | | 315/433/915 | |
| | Max. Data Rate (kbits/sec) | 250 | | 38.4 | |
| | Antenna Connector | MMCX | | | PCB solder hole |
| **Flash Data Logger Memory** | Chip | AT45DB014B | | | |
| | Connection Type | SPI | | | |
| | Size (kB) | 512 | | | |
| **Default power source** | Type | AA, 2x | | | Coin (CR2354) |
| | Typical capacity (mA-hr) | 2000 | | | 560 |




- A set of sensors for:
  - illumination;
  - temperature;
  - humidity.

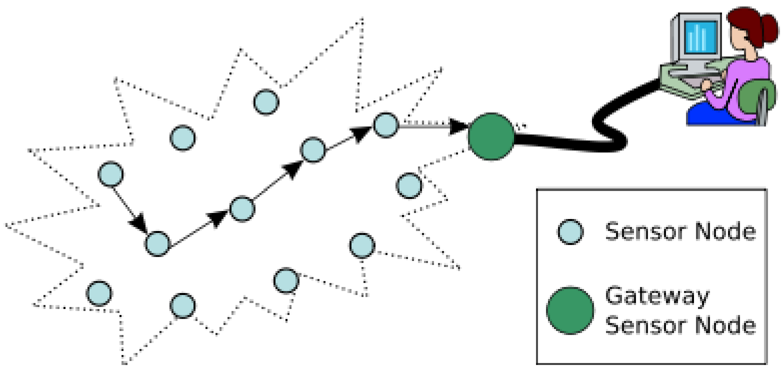
# All in all...

- WSNs are a technological solution for implementing a:
  - low cost,
  - self configuring,
  - fully customizable,

Distributed System...

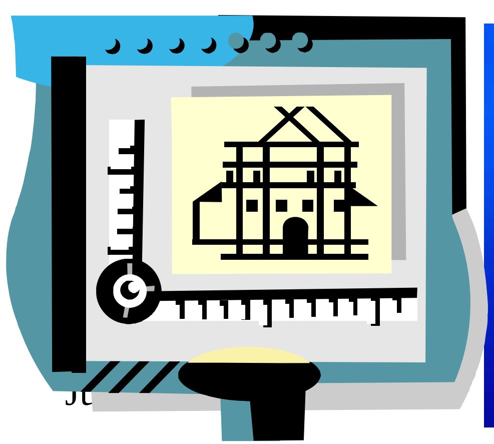
... do you remember Gianluca's lecture?

# A Distributed System




- Components of a Distributed System:
  - “intra-node” elements:
    - computation (concurrent programming, event handling, service-oriented network interface, etc.);
    - I/O w/ peripherals.
  - “inter-node” elements:
    - pkt transmission;
    - communication paradigm (P2P, fixed target, multiple targets, etc.);
    - medium access;
    - routing.

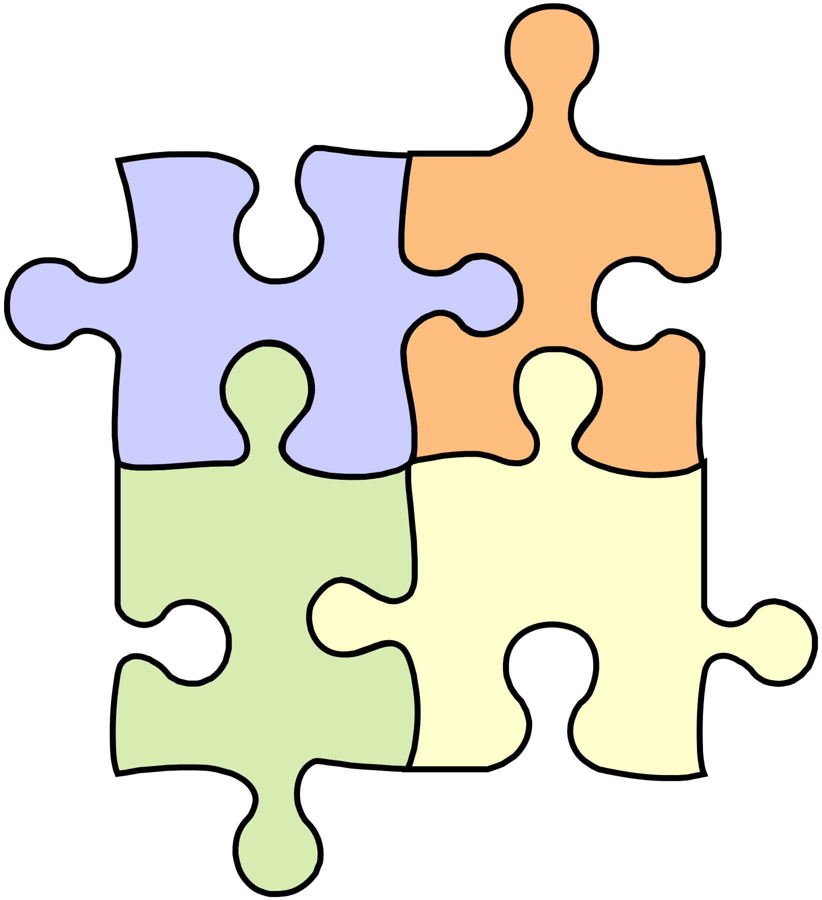
Node Architecture:



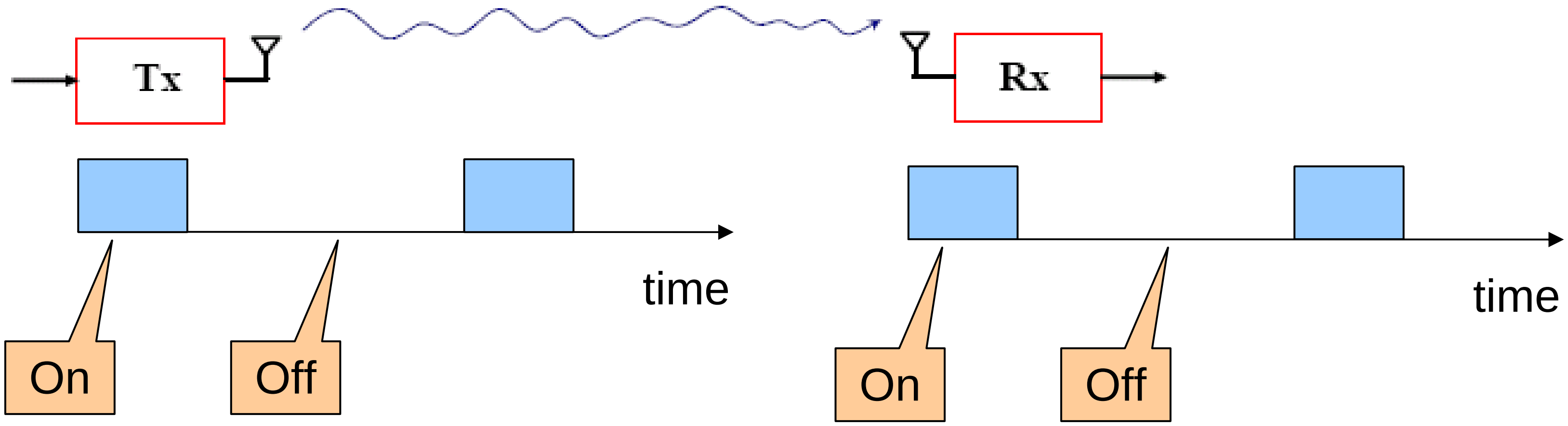
Network Architecture:

# Open research topics

- I would like to deploy a Sensor Network:
  - to maximize the device autonomy (power efficiency);
  - to enforce connectivity (topology management);
  - to profit of self-configuring capabilities (MAC & Network layer issues);
  - to extract the needed information (data management);
  - to update the running code (code management);
  - to act on the node profile (node architecture, OS-related issues).



# Power Management




- Radio power consumption is large in transmission and reception modes;
- Research in Electronics to reduce the energy consumption;
- It is convenient to implement sleep/awake duty cycles.

MICA mote Berkeley

| | | | |
|---|---|---|---|
| Transmit | 720 nJ/bit | Processor | 4 nJ/op |
| Receive | 110 nJ/bit | ~ 200 ops/bit | |

WINS node RSC

| | | | |
|---|---|---|---|
| Transmit | 6600 nJ/bit | Processor | 1.6 nJ/op |
| Receive | 3300 nJ/bit | ~ 6000 ops/bit | |

# Topology control and connectivity

- Problem statement:
  - In a densely deployed wireless network, a single node has many neighboring nodes with which direct communication would be possible when using sufficiently large transmission power. This is, however, not necessarily beneficial: high transmission power requires lots of energy, many neighbors are a burden for a MAC protocol, and routing protocols suffer from volatility in the network when nodes move around and frequently form or sever many links.
- Research trends:
  - Defines how to organize the nodes in groups;
  - How to set the power and channel transmission to define broadcast regions;
  - Connectivity is a strict term for point-to-point data path formation.

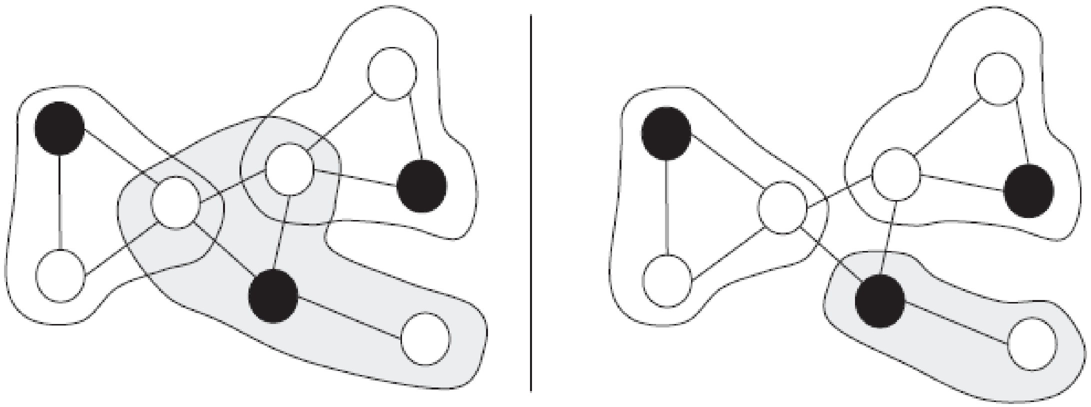
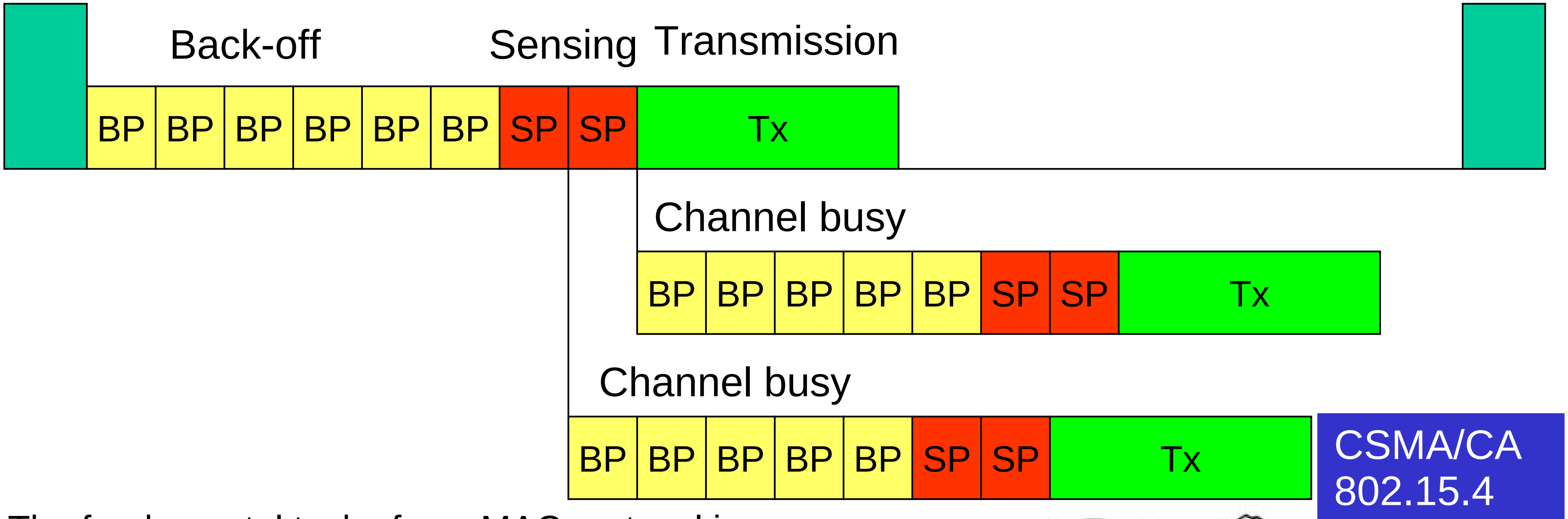


**Figure 10.18** Maximum independent set induces overlapping or nonoverlapping clusters (example adapted from reference [59])

# Data & Code management

- From ETH Summer School on WSN:
  - http://www.vs.inf.ethz.ch/events/dag2005/program/lectures/marron-2.pdf
- 4 categories of MiddleWares exist:
  - Classic middleware:
    - As “Impala” focuses on communication primitives, to tune long-running applications or to update parts of the code (versioning support);
  - Data-centric middleware:
    - As “TinyDB” abstracts the WSN as a distributed DB: code is immutable, the net is in charge of periodically sensing the environment;
  - Virtual Machines:
    - As “Maté” provides ports where pieces of code a.k.a. capsules (limited in number) may migrate through; continuous update of running WSNs; overhead?
  - Adaptive middleware:
    - As “TinyCubus”, a framework to wire together services, modules provided by the OS and by the user to minimize some cost metrics.

# MAC layer

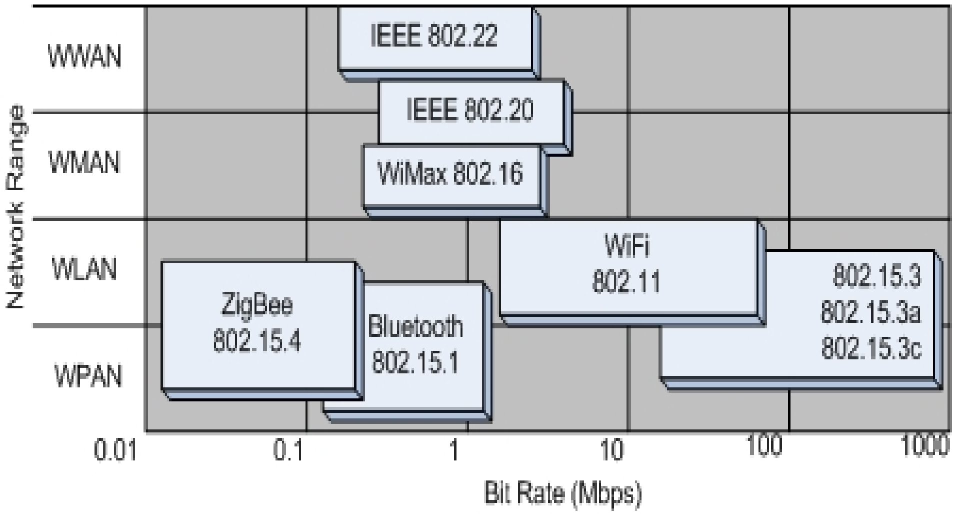



The fundamental task of any MAC protocol is to regulate the access of a number of nodes to a shared medium in such a way (*) that certain application-dependent performance requirements are satisfied.

Some of the traditional performance criteria are delay, throughput, and fairness, whereas in WSNs, the issue of energy conservation becomes important.

(*) Protocols and Architectures for Wireless Sensor Networks.
Holger Karl and Andreas Willig
 ISBN: 0-470-09510-5

# Communication Protocols



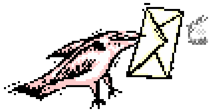

- Standard IEEE 802.15.4
  - Incompatible with other wireless technologies:
    - Bluetooth, Wi-Fi, WiMax, etc.
    - Different frequencies and protocols.
  - 2.4 Ghz available worldwide
    - 16 channels
  - Short range:
    - 50 - 100 meters outdoor
    - 5 - 10 meters indoor

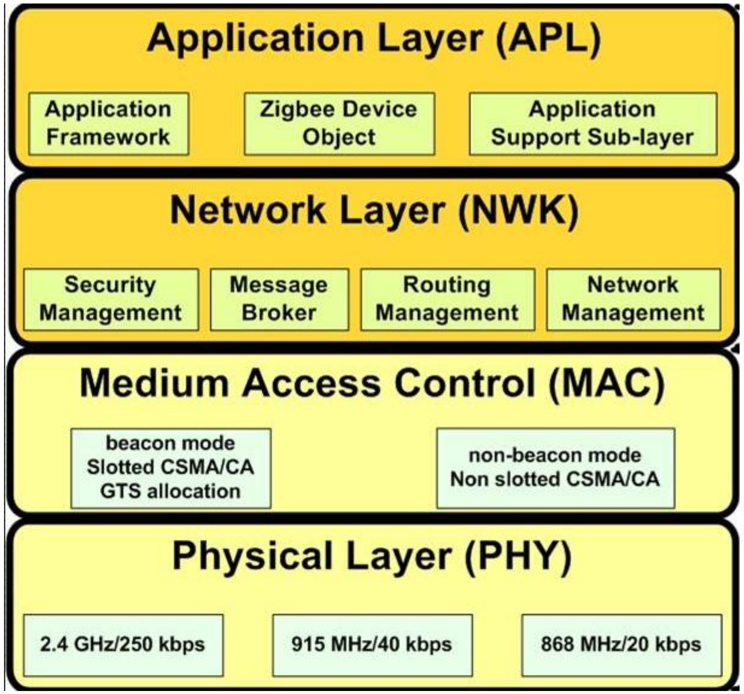
The industrial, scientific and medical (ISM) bands defined by the ITU-R are :

- 6.765–6.795 MHz (centre frequency 6.780 MHz)
- 13.553–13.567 MHz (centre frequency 13.560 MHz)
- 26.957–27.283 MHz (centre frequency 27.120 MHz)
- 40.66–40.70 MHz (centre frequency 40.68 MHz)
- 433.05–434.79 MHz (centre frequency 433.92 MHz) in Region 1 (Europe)
- ***868-868.8 MHz (centre frequency 468.40 MHz) in Region 1 (Europe)***
- ***902–928 MHz (centre frequency 915 MHz) in Region 2 (Americas)***
- ***2.400–2.500 GHz (centre frequency 2.450 GHz)***
- 5.725–5.875 GHz (centre frequency 5.800 GHz)
- 24–24.25 GHz (centre frequency 24.125 GHz)
- 61–61.5 GHz (centre frequency 61.25 GHz)
- 122–123 GHz (centre frequency 122.5 GHz)
- 244–246 GHz (centre frequency 245 GHz)

# Network layer

- The structure is the message which is delivered to a defined location:
  - Naming and addressing;
  - Path formation and discovery (routing);
  - Data flows.
- It's not obvious that in WSNs such layer exists.
- Some implementations exist (Zigbee).





# Network architecture: OS

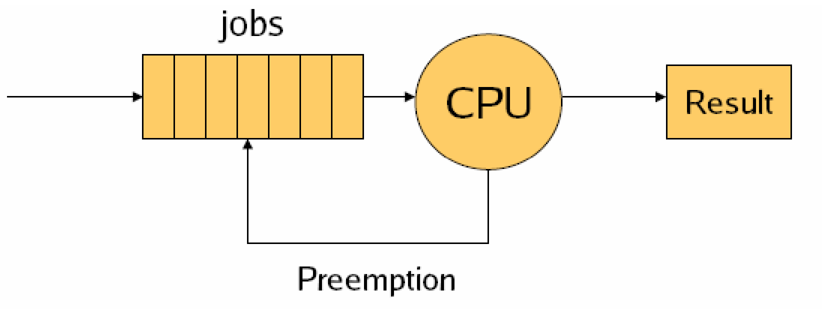
- An Operating System is a program that:
  - Provides an “*abstraction*” of the physical machine through a simple interface;
  - Each part of the interface is a “*service*”
- An OS is also a resource manager;
  - With the term “resource” we denote all physical entities of a computing machine;
  - The OS provides access to the physical resources;
  - The OS provides *abstract resources* (for example, a file, a virtual page in memory, a thread, an alarm).
- What are the services needed in WSN?
  - Filesystem and virtual paging seem not to be applicable;
  - What about memory management and scheduling?

# Node architecture

- The devices are usually limited in memory (a few Kbytes);
- Traditional OSs occupy many Mbytes:
  - Fully customized OSs;
- Constrained applications:
  - it is impossible to keep in memory many data structs;
  - short historical series;
  - the data must be sent out as soon as possible to clean the main memory.
- Typical figures are:
  - 1 Kbyte (OS) + 2 Kbytes (Network Stack) + 1 Kbyte (Application)

# OS: Multiprogramming

- In a device involved in many activities, it is natural to enforce concurrent programming:
- Design of a Multiprogrammed system:
  - Who decides when a job is suspended?
  - Who decides who is to be executed next?
- w/o OS, these tasks were carried out by the application itself:
  - Each job could suspend itself and pass the "turn" to the next job (*coroutines*);
  - However, this is not very general or portable!
- Today, the OS provides the multiprogramming services
  - The *scheduler module* chooses which job executes next depending on the status of the system




Examples of scheduling policies:

RoundRobin (time based)

FirstComeFirstServed
(activation-time based)

Rate Monotonic (period based)

Fixed Priority (priority based)

Earliest Deadline First (deadline based)

# The OS (for WSN) galaxy

A Crossbow MicaZ mote

TinyOS

Nano-RK (Carnegie Mellon University):

http://doi.ieeecomputersociety.org/10.1109/RTSS.2005.30

Erika Enterprise

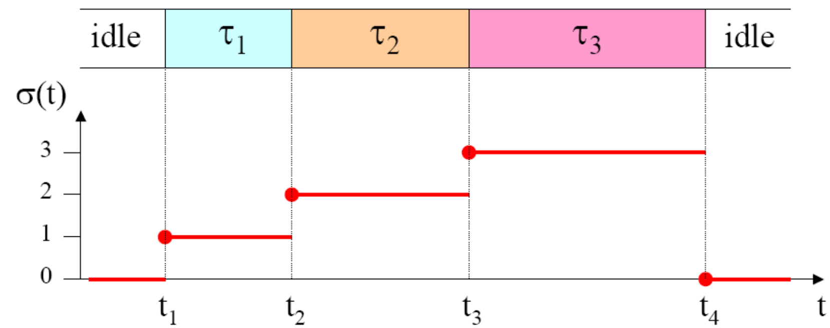
*a modular and optimized RTOS for 8, 16, 32 bit microcontrollers*

Evidence

Contiki :
A Dynamic Operating System for Memory-Constrained Networked Embedded Systems

# Node programming

- de facto standard: TinyOS;
- NesC programming language;
- Open Source;
- Continuous  upgrade by UCB and others;
- Module-oriented programming:
  - Very rich library:
    - Sensor data handling;
    - Communication;
    - Power saving.
  - Multi-programming:
    - Concurrent execution of code units (tasks).




$$\sigma(t) = \begin{cases} k > 0 & \text{if } \tau_k \text{ is running} \\ 0 & \text{if the processor is idle} \end{cases}$$

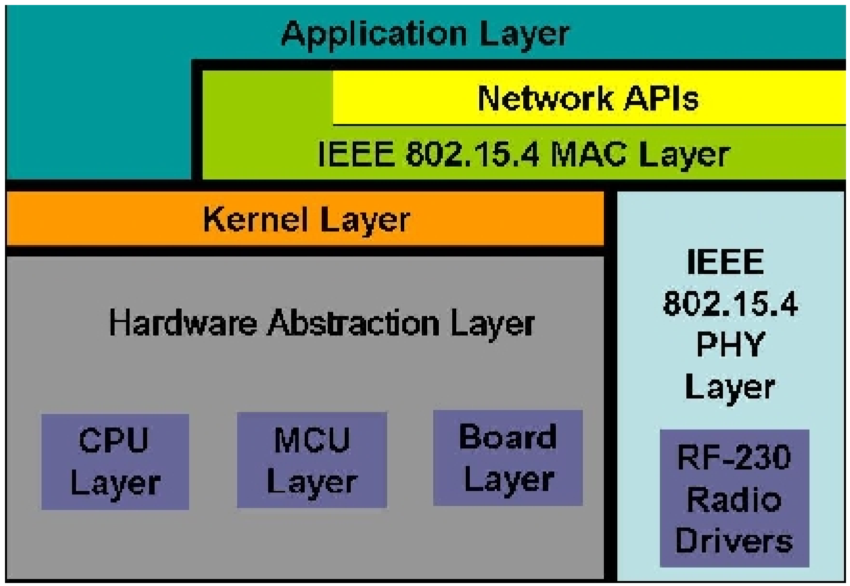
# OS: State of the art

- pros:
  - open source;
  - communication protocols already implemented;
  - minimum RAM and ROM footprint.
- cons:
  - non-preemptive scheduling policy (FCFS)

# ERIKA

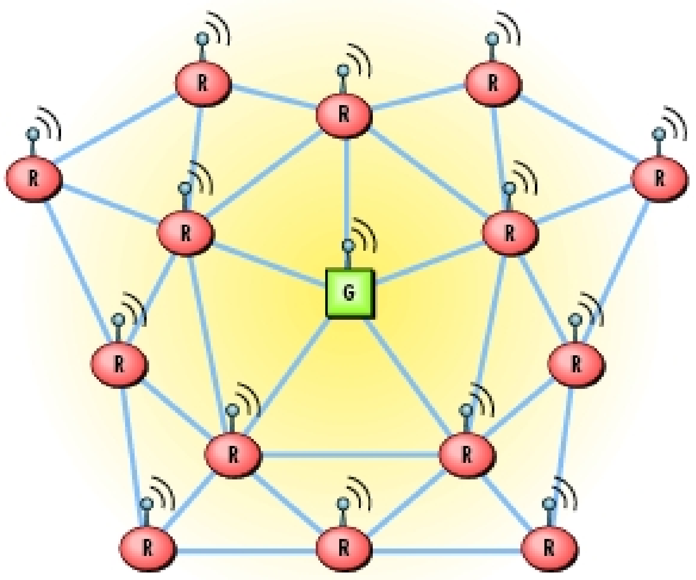
## Embedded Real tIme Kernel Architecture

- standard OSEK/VDX (used in Automotive);
- common memory model;
- priority driven scheduling;
- fully customizable;
- layered architecture;
  - Kernel layer;
  - Hardware abstraction layer:
    - MCU Layer;
    - CPU Layer;
    - Board Layer.

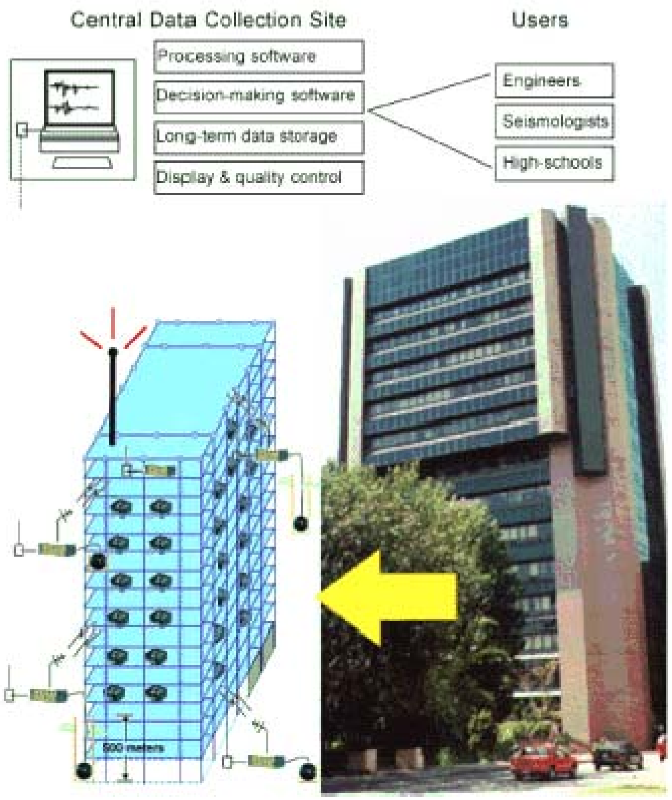
# New generation of WSNs

- **Wireless Sensor Networks**:
  - initially proposed for **environmental monitoring**;
  - start to be adopted in new domains (e.g. process control in assembly areas, telemedicine, health care) where **OS** plays a crucial role;
  - nodes initially devoted to **unique** specialized activities, can now be programmed in **multi-tasking**;
  - the **concurrent programming** is handled by the OS;
  - the services provided by the OS (scheduling, communication, resource allocation) **must be suited for the (distributed) application**.



- In these scenarios the network might:
  - have real-time constraints;
  - respect Quality of Service guarantees.

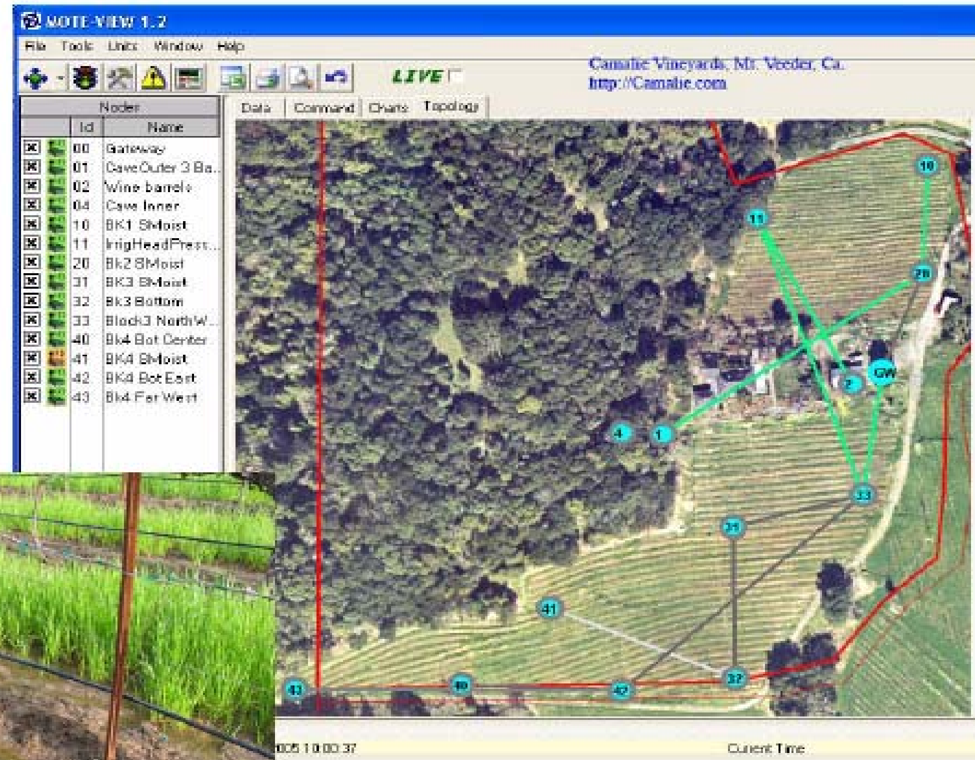
# Application: Structural Monitoring




# Application: agriculture (1/2)

- Camelie Vinyards : water saving policy in irrigation

- **The Deployment**
  - Area: 4.4 acres
  - 20 Nodes
  - Total cost of hardware: about $5,000



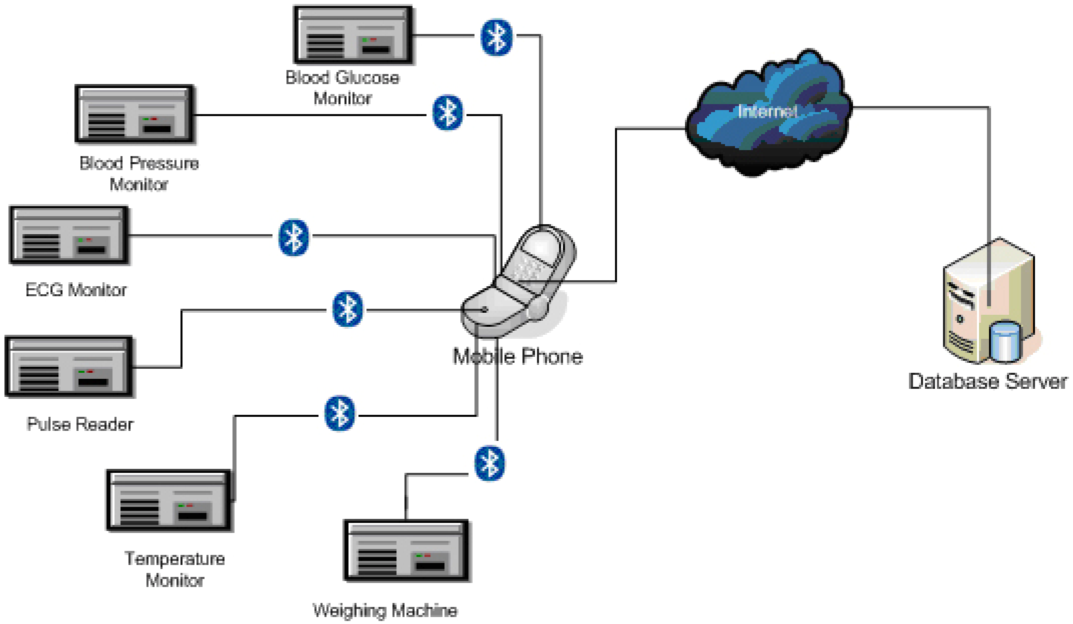
# Application: agriculture (2/2)

- In Vineyards, WSN's have been used to measure temperature, relative humidity, solar radiation, leaf wetness and soil moisture
  - Pickberry Vineyard
- EU funded project:

http://www.cobis-online.de/ist2006/5_Manes.pdf

# Application: Health care (telemedicine)

- Sensors are placed on the patient's body;
- Micro-controllers send critical data to a PDA interfaced with Database Server in order to compare the patient conditions with an on-line profile and eventually promptly reacts (dialing emergency numbers).






YouTube

Real-Time Systems Laboratory

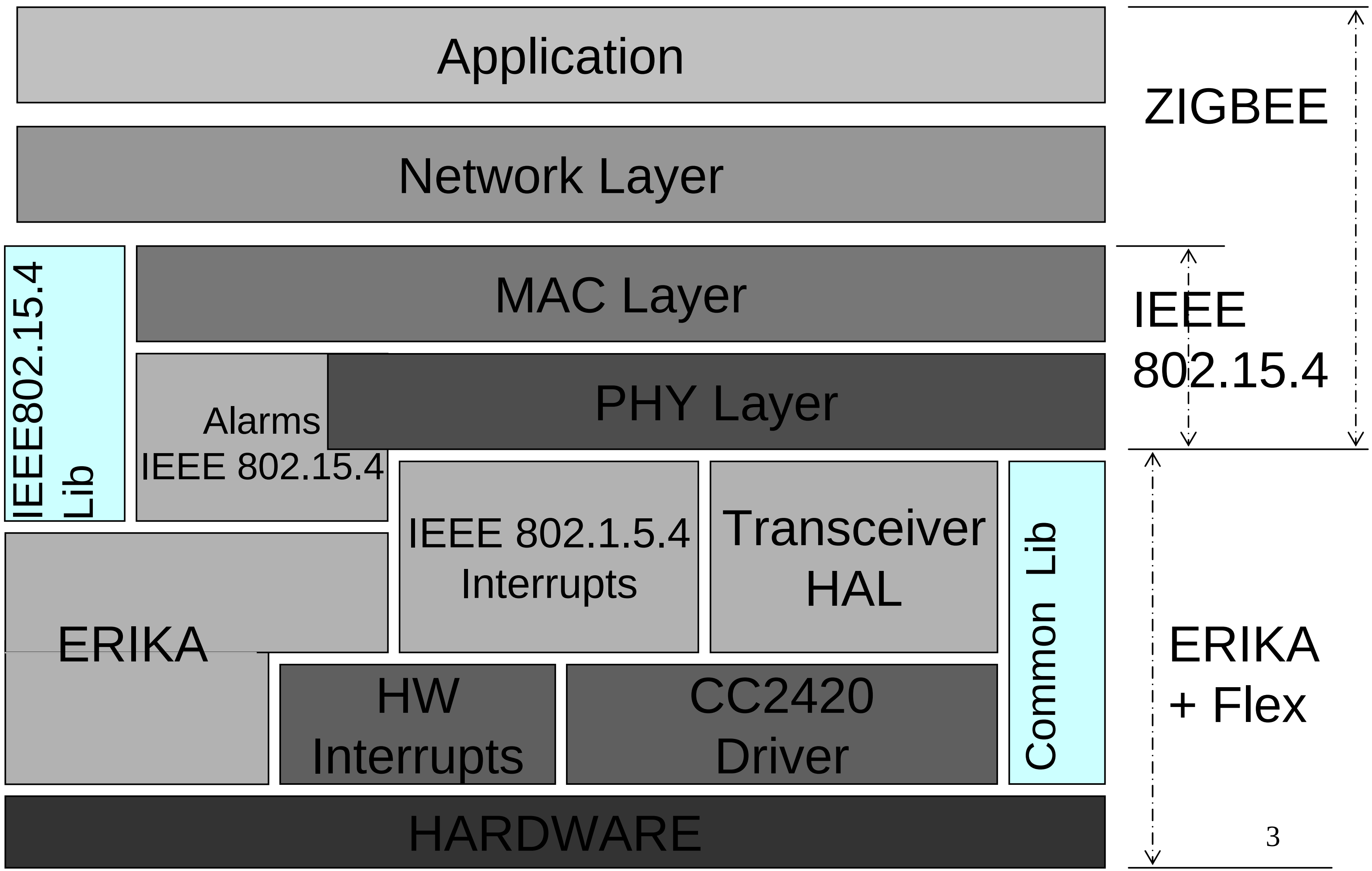
# IEEE 802.15.4 stack implementation over ERIKA real-time kernel

Mangesh Chitnis, André Cunha, Paolo Pagano

# Outline

- *ERIKA + IEEE802.15.4 Architecture.*
- *TASKS and Alarms.*
- *Memory Management.*
- *IEEE802.15,4 supported functionalities.*
- *Future Work.*

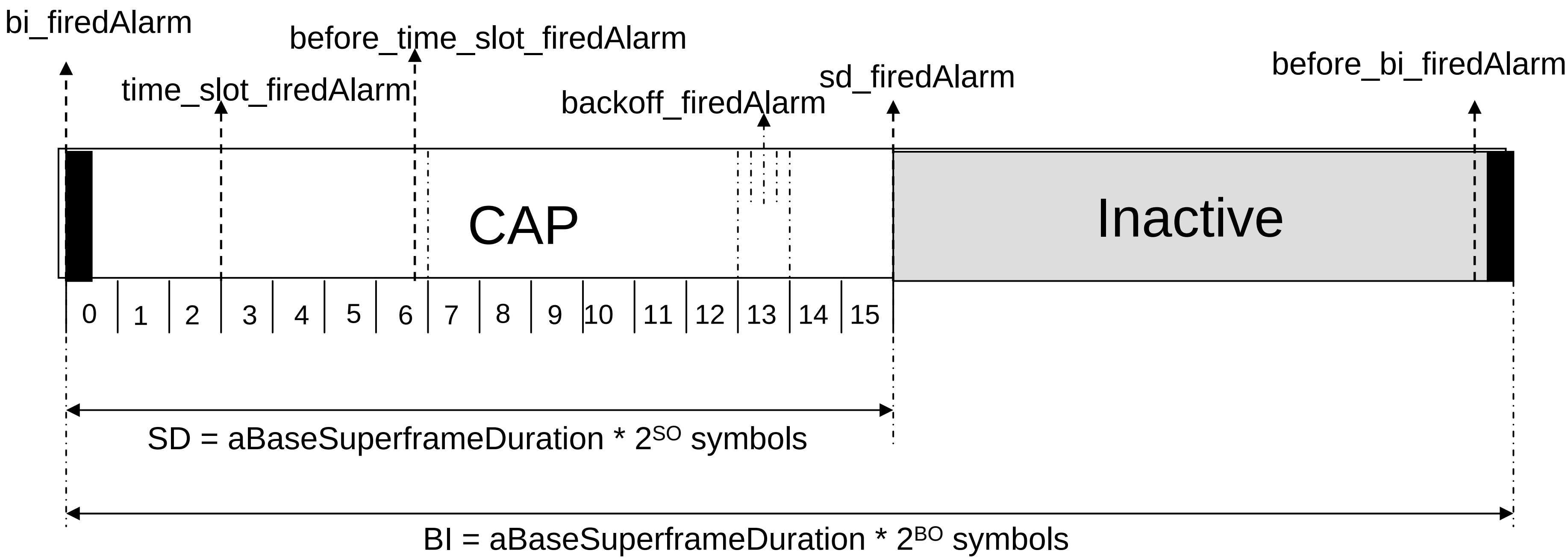
# ERIKA + IEEE802.15.4 Architecture



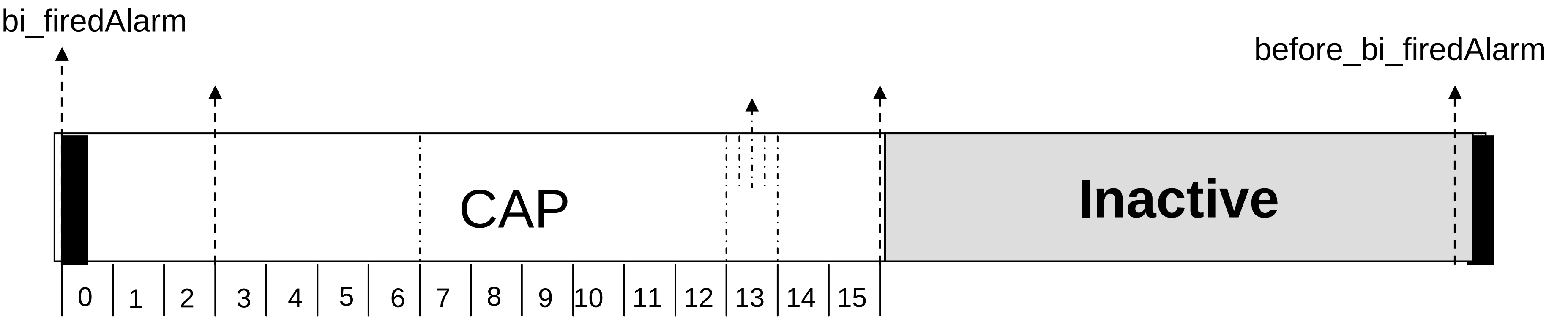
# ERIKA + IEEE802.15.4 Functions

- **Operating System (ERIKA)**
  - Providing the needed alarms for the MAC mechanisms
    - Superframe Definition
    - CSMA/CA
  - CC2420 Driver
  - Mapping the protocol services into tasks
  - Memory management i.e. buffer queues statically allocated in the global scope to accommodate message payloads (MPDU) used for send and receive;
  - UART Console - Debug
- **Communication Stack (open-ZB)**
  - Beacon transmission
  - Static Network formation (without negotiation)
  - IEEE 802.15.4 Superframe operation;
  - Coordinator/End device time synchronization using periodic beacon broadcasting;
  - Data transmission and reception in:
    - Unslotted mode;
    - Slotted mode (using the Open-ZB mechanisms for CSMA/CA).

# Task Definition – Superframe Management



# Task Definition – Superframe Management

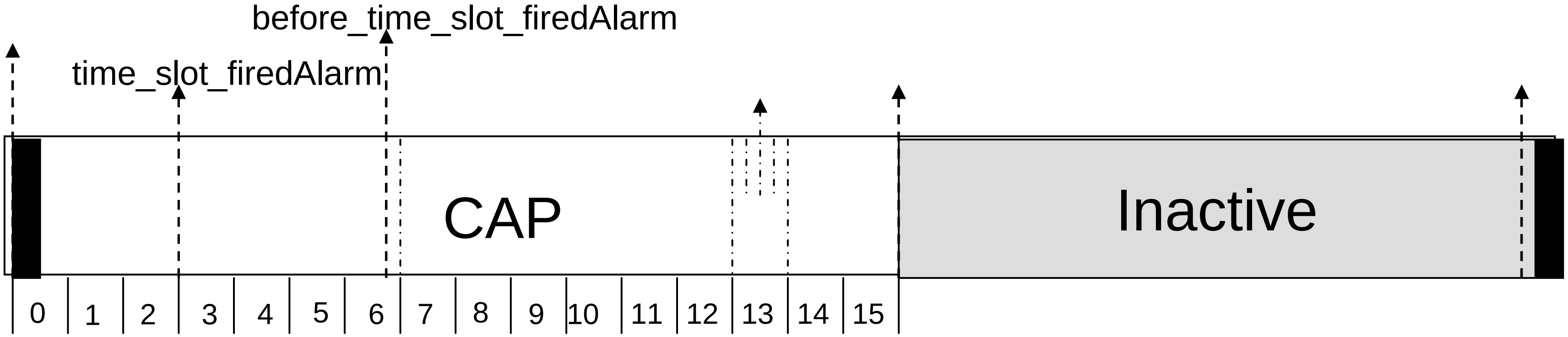



BI_Fired TASK:

- Highest Priority Task.
- Fired on every beacon interval
- If Co-ordinator; then send beacon frame.

| 2 | variable | variable | variable |
|---|---|---|---|
| Superframe specification | GTS fields | Pending address | Beacon payload |

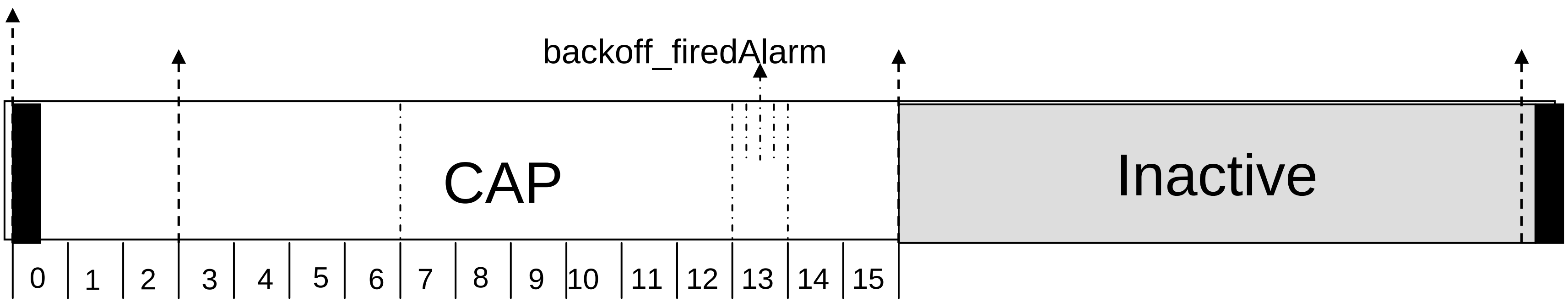
# Task Definition – Superframe Management




TimeSlot_Fired TASK:

- Fired every time slot = Superframe Duration / 16 (during the Superframe duration);
- Used during GTS to send or receive data;
- Before TimeSlot to change the transceiver status.

# Task Definition – Superframe Management

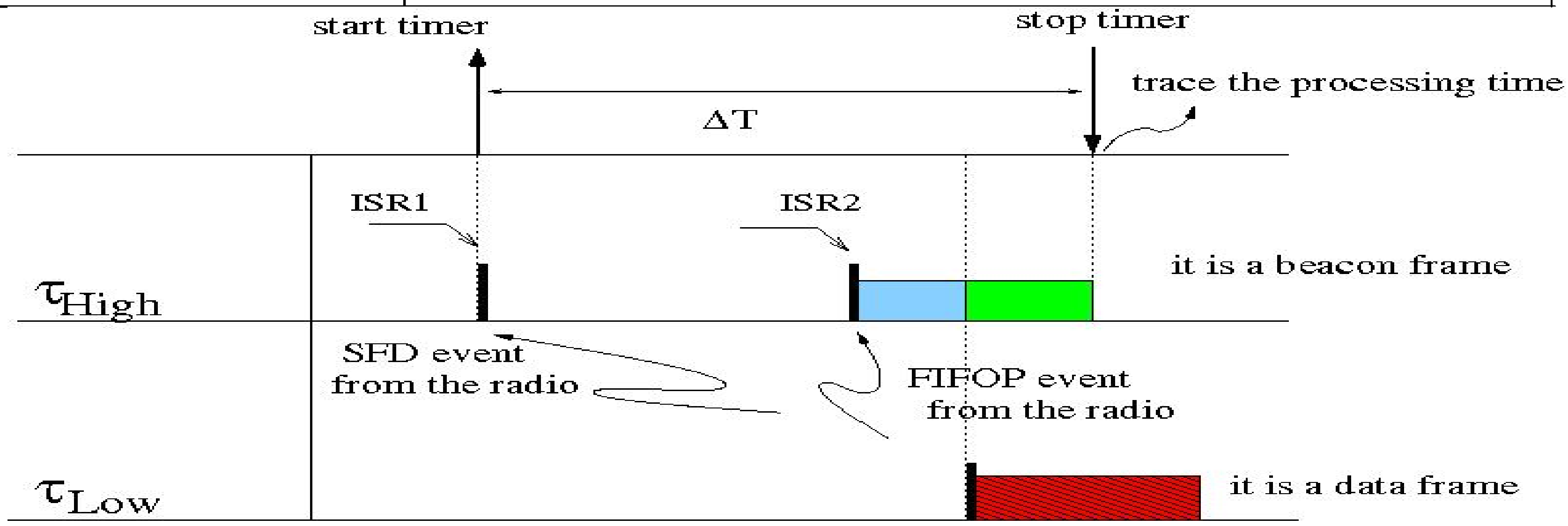



Backoff_fired TASK:

- Fired every backoff period = 20 symbols = 320 us
- Used by slotted csma to send the data.

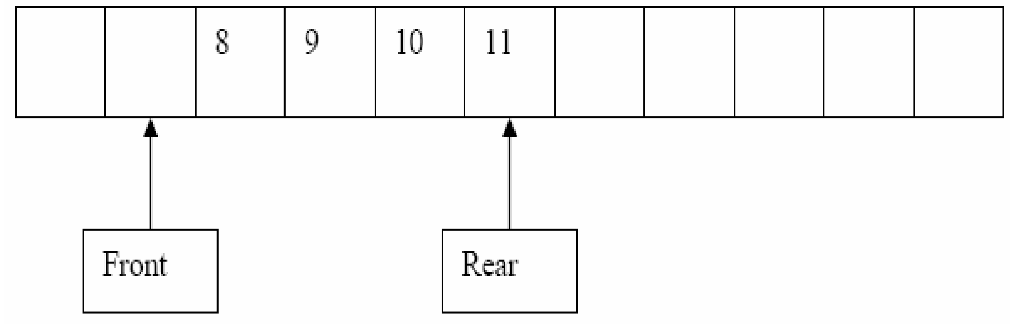
# Task Definition – Frame Reception

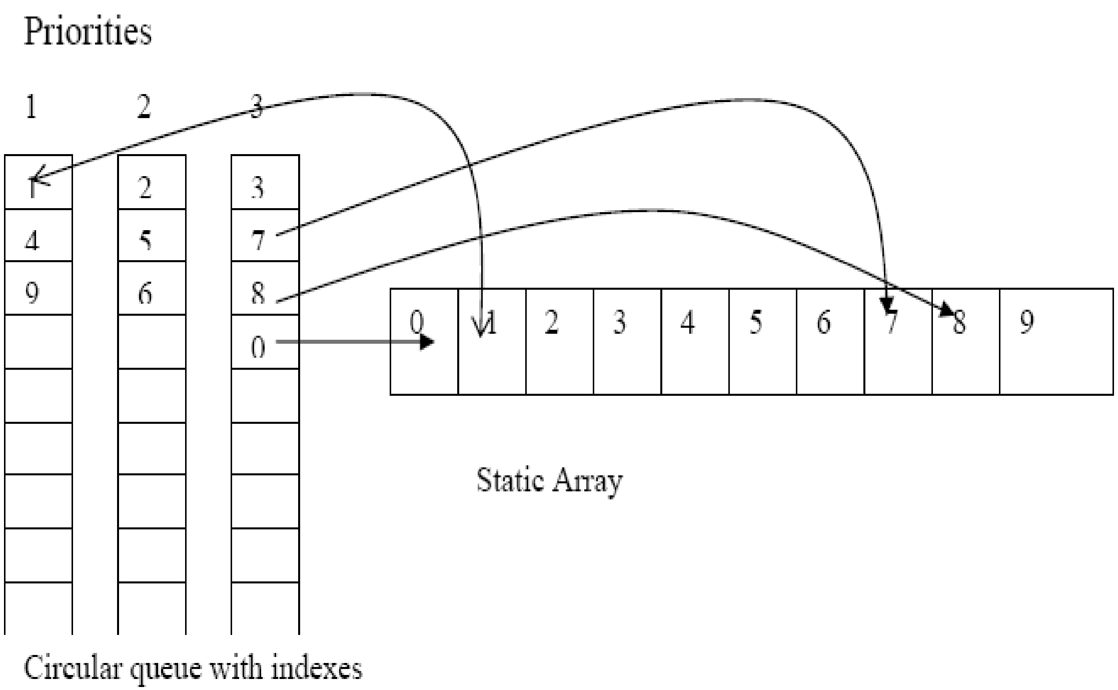
| Task Name | Description |
|---|---|
| ReadDispatcher | Task posted upon the reception of the FIFO ISR |
| DataFrameDispatcher | Task posted in the ReadDispatcher to process data frames |
| AckFrameDispatcher | Task posted in the ReadDispatcher to process acknowledgment frames |
| CmdFrameDispatcher | Task posted in the ReadDispatcher to process command frames |



# Memory Management - Buffers

1. Circular queue for receive;
2. Priority based indexed queues for send;
3. Indexed structure for GTS data.



# Phy Layer / Mac Layer Functions

- Phy Layer
  - Activation and deactivation of the radio transceiver;
  - Turnaround of the transceiver;
  - Energy Detection (ED) within the current channel;
  - Link quality indicator (LQI) for received packets;
  - Clear Channel Assessment (CCA);
  - Channel frequency selection;
  - Data transmission and reception.
- Mac Layer
  - Generating beacon frames if the device is a coordinator;
  - Synchronizing to the beacons;
  - Employing the CSMA-CA mechanism for channel access;
  - Handling and maintaining the GTS mechanism;
  - Supporting PAN association and disassociation.

# How to install Open-ZB stack

- Go to Mangesh home page:

http://feanor.sssup.it/~mangesh/IEEE802154.tar.gz

- Place the folder ieee802154 in ee/contrib;
- Place the folder pic30_ieee802154_app/ in ee/examples;
- When you will instantiate a new RT-Druid project you wil find a template named “ieee802154_app”.

# Demo 1

- We will show you a working demo where:
  - 2 device nodes (1 is coordinator, 1 is end device) establish a network;
  - the coordinator periodically sends a greeting message to the end device;
  - the device receives the packet and sends a reply;
  - both the nodes display the content of the received messages.

# Assignment 1

- During the lab session you are required to implement the following application:
  - measure some sensor variables (acceleration, illumination, temperature) in the End Device and communicate it to the coordinator (message #1);
  - the coordinator acknowledges the receipt sending the value back (message #2);
  - At the device the value in the payload of message #2 is compared with that in memory. In case of mismatch, some action is taken (the buzzer sounds, the Leds switch on, etc.).