Embedded Systems

Giuseppe Anastasi
g.anastasi@iet.unipi.it
Pervasive Computing & Networking Lab. (PerLab)
Dept. of Information Engineering, University of Pisa

Aknowledgments: Antonio Prete, Mario Di Francesco

Objectives

- Introduce the definition and characteristics of embedded systems
- Discuss the main requirements of an embedded system
- Analyze two classes of embedded systems
  - *Real Time* Embedded Systems
  - *Distributed* Embedded Systems

Overview

- Basic Concepts
- Main Requirements
- *Real Time* Embedded Systems
- *Distributed* Embedded Systems

Definizione

- Sistema informatico *dedicato*, progettato cioè per svolgere un compito *preciso* e *determinato*
- *Embedded:* tali sistemi sono parte integrante di sistemi più grandi
  - compiti di controllo, elaborazione, memorizzazione …
- Il PC è un sistema general purpose
  - Può essere utilizzato per realizzare sistemi embedded.
  - Include diversi sistemi embedded (disco, CD, scheda video, …).

Aree di applicazione

- Acquisizione dati ed elaborazione
- Comunicazioni
- Sistemi di controllo digitale
- Robotica
- Interfacce
- Unità ausiliari:
  - Display
  - Dischi
  - Monitoraggio e protezione di sistemi
  - Test e diagnosi di sistemi

Solo alcuni esempi

Caratteristiche

- Specializzazione e ottimizzazione
  - Con riferimento all'applicazione svolta
- Inclusione di elettronica ed altri dispositivi dedicati per
  - Interagire con il mondo esterno
  - Svolgere alcune funzioni elaborative
- Possono avere vincoli Real-time
- Possono essere sistemi distribuiti
  - O parte di un sistema distribuito

Specializzazione

- Un sistema embedded è progettato per eseguire una sola applicazione (o poche applicazioni)
  - Le applicazioni da svolgere sono note a priori, prima che il processo di progettazione inizi
- Futuri aggiornamenti
  - flessibilità per i futuri aggiornamenti o per un eventuale riutilizzo del componente.
  - si raggiunge questo scopo rendendo il sistema riprogrammabile

Hardware dedicato

- Interazione col mondo esterno analogico
  - Necessità di campionare, memorizzare, e trasmettere segnali.
    - *Sensori*
    - *Attuatori*
    - *Convertitori A/D e D/A*
- Interazione con l'utente
  - Avviene con mezzi spesso limitati
    - *Display di dimensione ridotta*
    - *Dispositivi di input limitati*
    - *Dispositivi di I/O specializzati rispetto alle competenze o al modo di operare dell'utente*

Applicazioni Real Time

- Un sistema real-time esegue dei task con vincoli temporali
- Sistemi Hard real-time vs. Soft real-time
- I sistemi Hard real-time molto spesso sono dei sistemi embedded

Sistemi Embedded Distribuiti

PerLab

- Composti da più sottosistemi che cooperano nello svolgimento di un servizio
- Vantaggi:
  - Localizzazione dell'elaborazione
    - il lavoro è fatto dove serve
  - Specializzazione dell'elaborazione
    - il lavoro è fatto da chi lo sa fare meglio
  - Ridondanza
    - possibilità di supplire a guasti parziali
- Svantaggi
  - Necessità di coordinamento e comunicazione fra i vari sottosistemi
  - Aumento della complessità

Overview

PerLab

- Basic Concepts
- **Main Requirements**
- *Real Time* Embedded Systems
- *Distributed* Embedded Systems

Principali requisiti

- Requisiti funzionali
- Requisiti temporali
- Requisiti di affidabilità
- Consumo
- Prestazioni
- Costo

Spesso i requisiti risultano in contrasto tra loro!

Requisiti funzionali

- Contengono la specifica della parte elaborativa del sistema
  - Definiscono quali sono i dati di ingresso e da dove provengono (sensori, utente umano, …)
  - Definiscono quali sono i dati di uscita e a chi devono essere inviati (attuatori, utente umano, …)
  - Definiscono le relazioni che esistono fra dati di ingresso e di uscita
    - *quali dati di uscita devono essere prodotti dal sistema in funzione dei dati di ingresso*

Requisiti temporali

- I task possono avere deadline
- Requisiti temporali derivanti dall'esecuzione di task periodici
- Latenza nella risposta
- Latenza nella rilevazione degli errori
- Interazione con l'uomo

Requisiti di affidabilità

- Affidabilità
  - Il sistema deve presentare un *MTTF (Mean-Time-To-Failure)* sufficientemente alto
- Sicurezza (Safety)
  - Gestione di particolari casi critici
  - Certificazione (obbligatoria in alcuni settori quali il settore spazio, aviazione ed il settore militare)
- Riparabilità
  - *MTTR (Mean-Time-To-Repair)* sufficientemente basso
- Disponibilità
  - D = MTTF / (MTTF + MTTR)

Consumo

- Il consumo è uno dei *requirements* principali
  - Influisce sulla complessità dell'hardware
    - alimentatori, batterie, sistemi di raffreddamento
  - E anche sul costo complessivo
- In sistemi alimentati a batteria è fondamentale
  - Maggiore autonomia (batterie ricaricabili)
  - Maggiore tempo di vita (batterie non ricaricabili)
- Il consumo influisce sulla dissipazione del calore
  - e del rumore (es. ventole)

Prestazioni

- Migliori prestazioni ➔
  - ☺ si soddisfano più facilmente i requisiti temporali
  - ☺ si aumenta l'usabilità del sistema stesso
  - ☹ si aumenta il consumo del sistema
- È possibile variare dinamicamente le prestazioni per controllare il consumo
- Le prestazioni influenzano il costo finale del dispositivo

Costo

- Per sistemi prodotti in larga scala il costo finale è un aspetto fondamentale.
- È profondamente legato alle scelte di progetto,
  - Scelta dell'architettura (distribuita, centralizzata, …)
  - Hardware (tipo di CPU, tipo e quantità di memoria, periferiche di I/O)
  - Software (costi di progettazione e di sviluppo)
  - Licenze e diritti per HW e SW (librerie, ambienti di sviluppo, compilatori, ambienti di testing)
  - Numero di pezzi prodotti

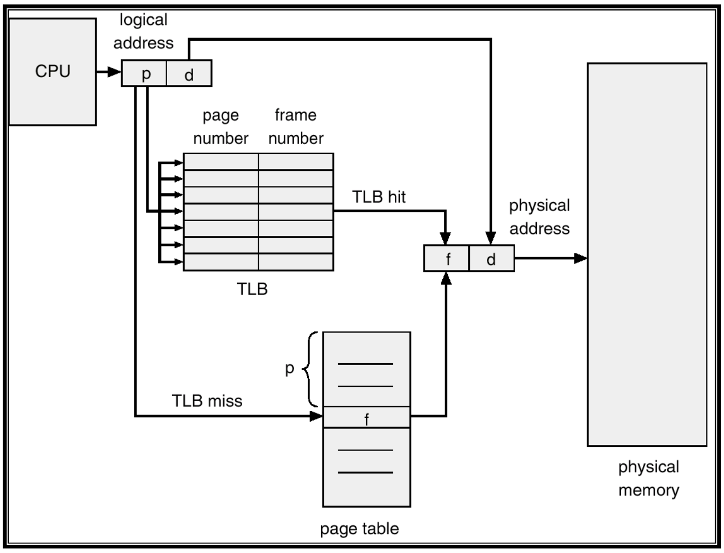
Overview

- Basic Concepts
- Main Requirements
- ***Real Time* (Embedded) Systems**
- *Networked* Embedded Systems

Features of Real-Time Kernels

- Most real-time systems do not provide the features found in a standard desktop system
- Reasons include
  - Real-time systems are typically single-purpose
  - Real-time systems often do not require interfacing with a user
  - Features found in a desktop PC require more substantial hardware than what is typically available in a real-time system

Virtual Memory and Address Translation



Virtual Memory in Real-Time Systems

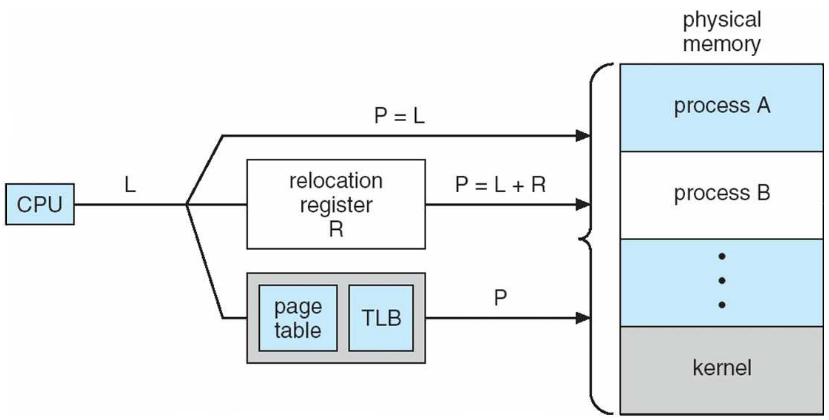
- Address translation may occur via:

  (1) Real-addressing mode where programs generate actual addresses

  (2) Relocation register mode

  (3) Implementing full virtual memory

Address Translation



Implementing Real-Time Systems

- In general, real-time operating systems must provide:

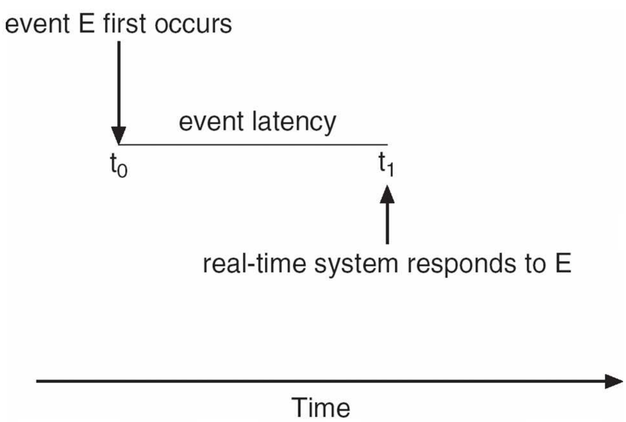
  (1) Preemptive, priority-based scheduling

  (2) Preemptive kernels

  (3) Latency must be minimized

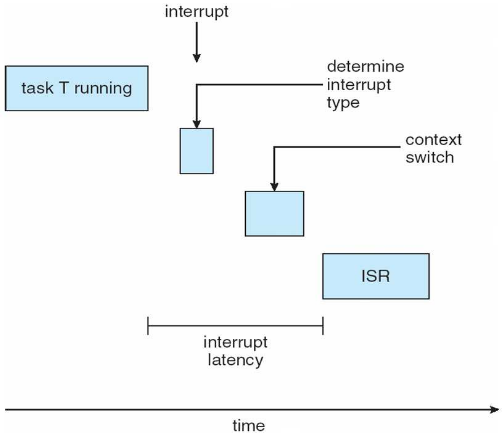
Minimizing Latency

- **Event latency** is the amount of time from when an event occurs to when it is serviced.



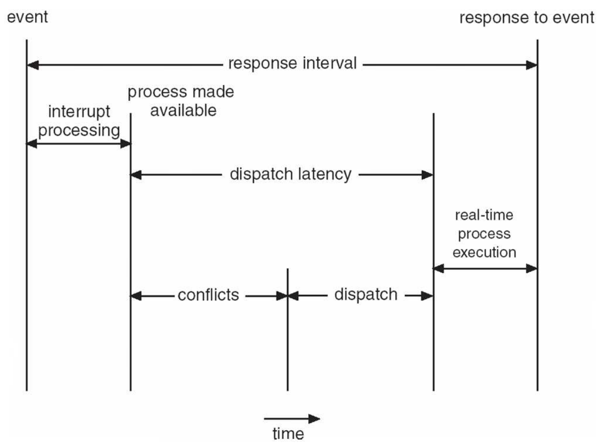
Interrupt Latency

- Period of time from when an interrupt arrives at the CPU to when it is serviced



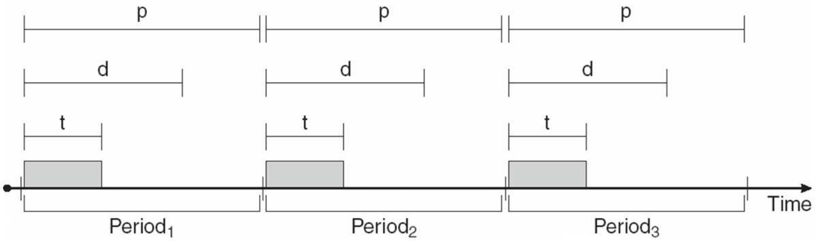
Dispatch Latency

- Amount of time required for the scheduler to stop one process and start another



Real-Time CPU Scheduling

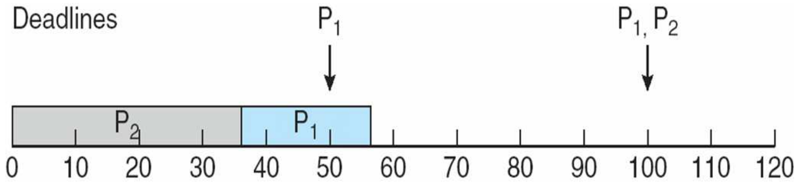
- Periodic processes require the CPU at specified intervals (periods)
- ***p*** is the duration of the period
- ***d*** is the deadline by when the process must be serviced
- ***t*** is the processing time



Real-Time CPU Scheduling

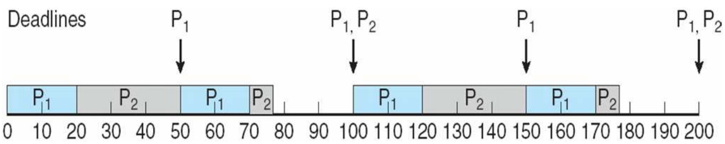
P1: p=50 ms, t=20 ms t/p=0.4
P2: p=100 ms t=35 ms t/p=0.35

Scheduling of tasks when P2 has a higher priority than P1



Rate Monotonic Scheduling

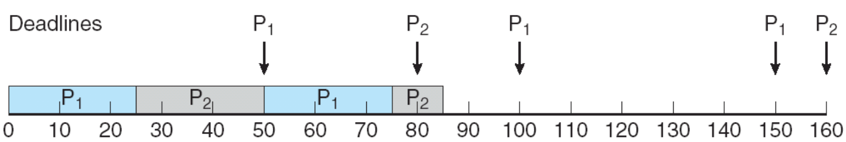
- A priority is assigned based on the inverse of its period
- Shorter periods = higher priority;
- Longer periods = lower priority
- P_1 is assigned a higher priority than P_2.



Rate Monotonic Scheduling

Missed Deadlines with Rate Monotonic Scheduling

P1:	p=50 ms,	t=25 ms	t/p=0.50
P2:	p=80 ms	t=35 ms	t/p=0.44

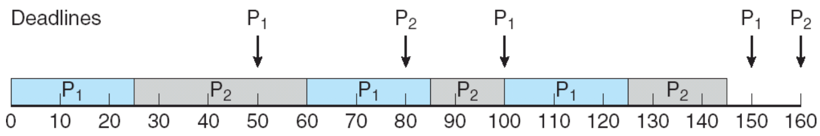


Maximum utilization: $2(2^{1/n}-1)$
- n=2: 0.83

Earliest Deadline First Scheduling

- Priorities are assigned according to deadlines:
  - the earlier the deadline, the higher the priority;
  - the later the deadline, the lower the priority

P1:	p=50 ms,	t=25 ms	t/p=0.50
P2:	p=80 ms	t=35 ms	t/p=0.44



Proportional Share Scheduling

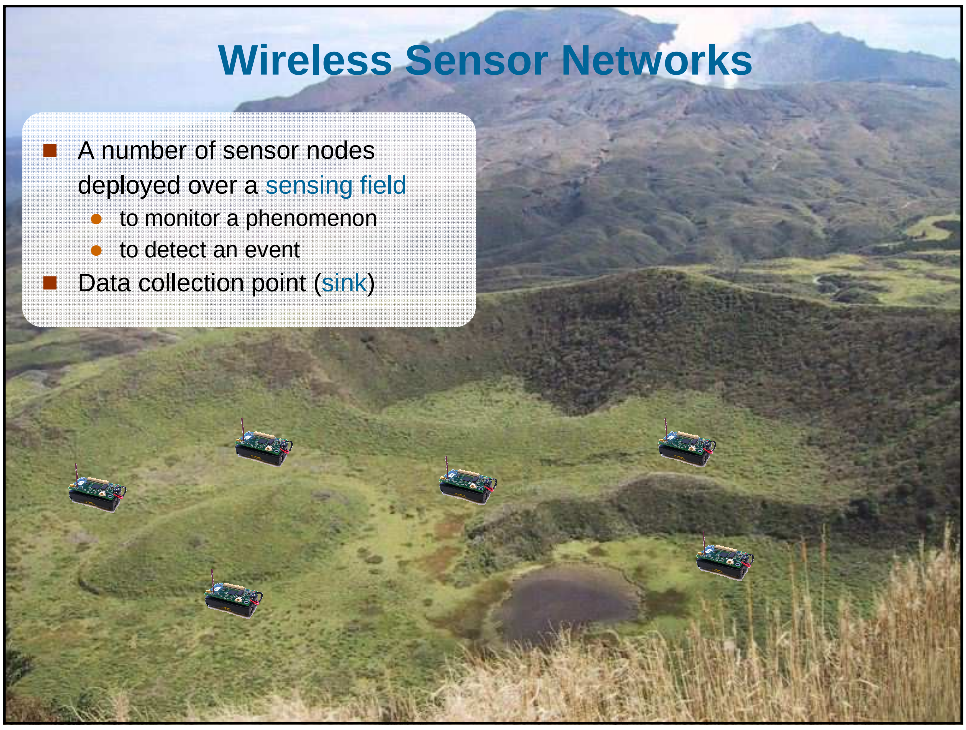
- ***T*** shares are allocated among all processes in the system
- An application receives ***N*** shares where ***N < T***
- This ensures each application will receive ***N / T*** of the total processor time

Overview

- Basic Concepts
- Main Requirements
- *Real Time* Embedded Systems
- ***Networked* Embedded Systems**
  - **Wireless Sensor Networks**

Distributed Embedded System

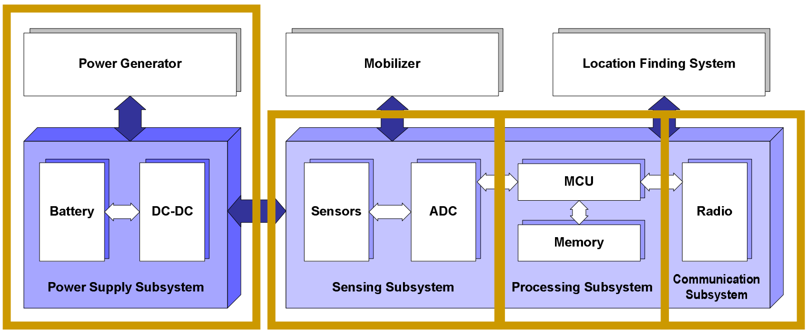
- Several subsystems connected through communication links
- Data acquisition, processing and storage are carried out cooperatively
- Classification based on communication
  - Wired
    - Automotive, intelligent home, …
  - Wireless
    - Environmental monitoring, health care,
    - Wireless Sensor Networks (Networked Embedded Systems)



Wireless Sensor Networks

- A number of sensor nodes deployed over a sensing field
  - to monitor a phenomenon
  - to detect an event
- Data collection point (sink)

Sensor Node Architecture



Battery powered devices
Batteries cannot be changed nor recharged

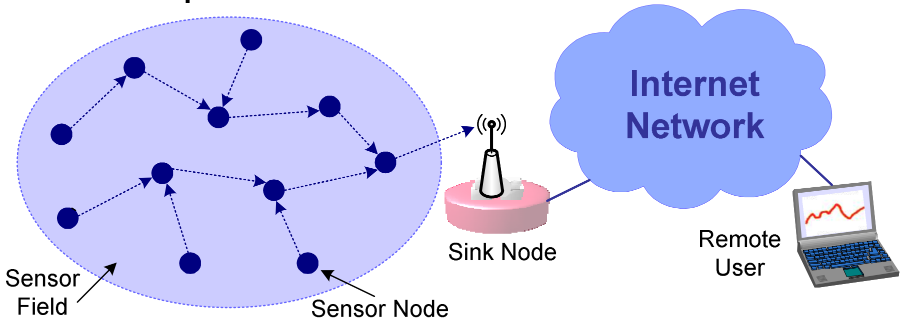
Often negligible power consumption

Short range wireless communication
Radio is the most power hungry component

Local data processing and data storage

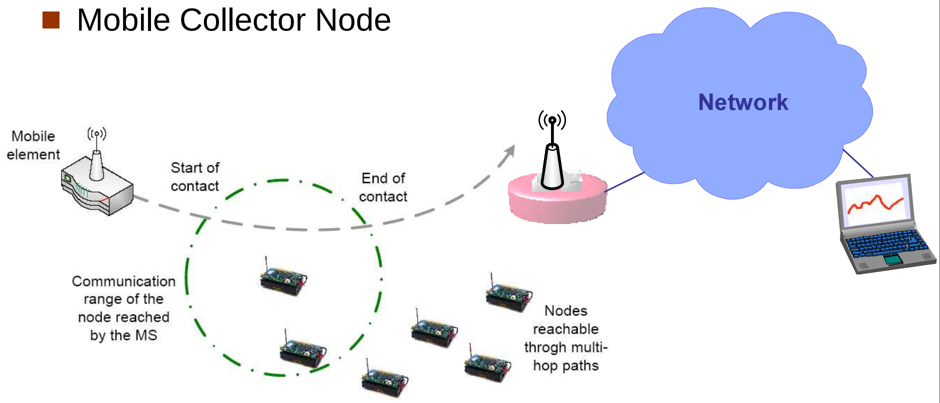
Wireless Sensor Networks

Multi-hop Sensor Network



Wireless Sensor Networks

- Mobile Collector Node



Sensors

- Sensor types
  - seismic
  - magnetic
  - thermal
  - visual
  - infrared
  - acoustic
  - radar...
- Sensor tasks
  - temperature
  - humidity
  - vehicular movement
  - lightning condition
  - pressure
  - soil makeup
  - noise levels
  - mechanical stress levels
  - current characteristics (speed, direction, size) of an object
  - ...

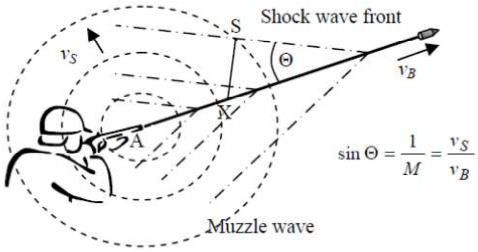
Potential Application Areas

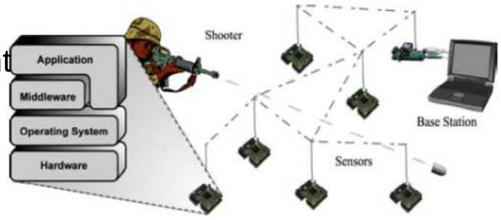
- Military Applications
- Environmental Monitoring
- Precision Agriculture
- Location/Tracking
- Industrial applications
- Health Monitoring
- Smart Buildings
- Smart Grid
- Smart Cities
- Smart *
- …

Military Applications

PerLab

- Monitoring
  - friendly forces, equipment, ammunition
- Battlefield surveillance
- Reconnaissance of opposite forces
  - sniper detection
- Targeting
- Battle damage assessment
- Attack detection
  - nuclear, biological, chemical

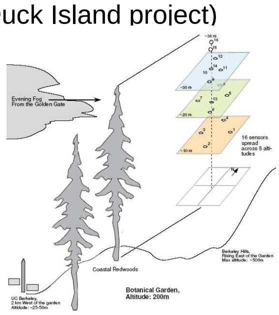




Environmental Monitoring

PerLab

- Environmental Monitoring
  - Temperature, humidity, pollution level
  - Habitat monitoring
    - monitoring of petrel habitat (Great Duck Island project)
  - Microclimate monitoring
    - Berkeley botanical garden
- Alert systems
  - fire detection
  - flood detection
  - seismic events



Precision Agriculture

- Temperature
- Humidity
- Wind Speed and Direction
- Soil moisture

Location/Tracking

- Location/Tracking of moving objects
  - Surveillance
  - Presence assessment
  - Animals' movements
- Inventory Control
  - easy localization of items
  - smart management of items
- Vehicles
  - tracking and detection
  - car theft detection
  - remote monitoring of parking places

Industrial Applications

- Distributed Intelligent Sensing System for
  - Factory automation
  - Process Control
  - Real-time monitoring of machinery’s health
  - Detection of liquid/gas leakage
  - Remote monitoring of contaminated areas
  - Real time inventory management
  - …

E-Health

- Remote monitoring
  - chronicle patients
    - physiological data monitoring
  - elderly people
    - fall detection
- Hospital
  - monitoring of patients
  - tracking of doctors and attendants
  - drug administration
    - minimize adverse drug events (e.g., allergies to a specific medicine)

Smart Buildings

- Building Automation
  - temperature and air flow control
  - light level control
  - energy efficiency
- Smart Home
  - Smart appliances
    - sensors and actuators inside appliances
  - easy management of home devices
    - both local and remote

Smart Cities

- Environmental Monitoring
  - Temperature, Noise, Pollution
- Parks and Gardens Irrigation
- Parking Area Management
- Guidance to free Parking Slots
- Traffic Intensity Monitoring
  - Intelligent traffic management
- Mobile Environmental Monitoring
  - Better coverage than static monitoring
- …

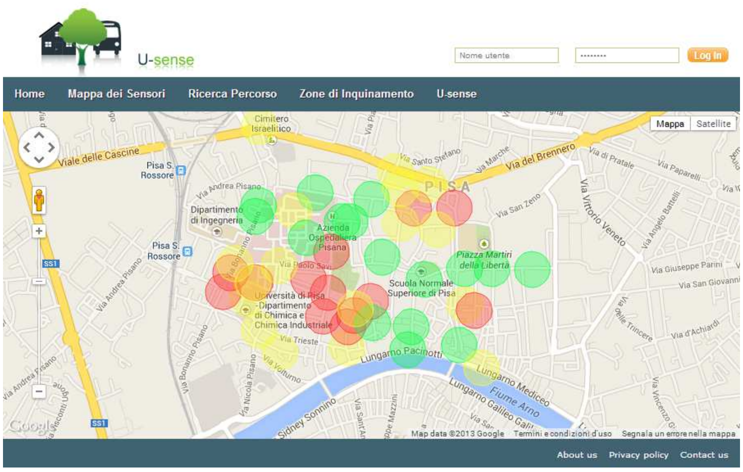
Environmental Monitoring

- Measured Parameters
  - Temperature
  - Light
  - Noise
  - CO
  - …

Light intensity can be used for *smart lighting*

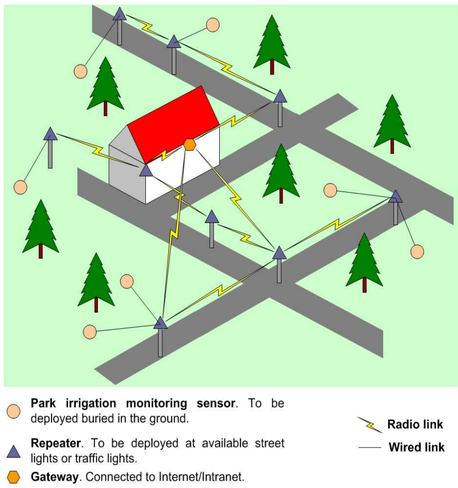
If some critical parameters goes above a threshold the system sends an alarm.

Pollution Monitoring



Parks and Gardens Irrigation

- Goal
  - to control and make more efficient the irrigation in certain parks and gardens
- Sensors
  - Anemometer, pluviometer.
  - Atmospheric pressure, solar radiation, air humidity and temperature sensors.
  - Soil temperature and humidity sensors.
  - Evaluation of water consumption sensor

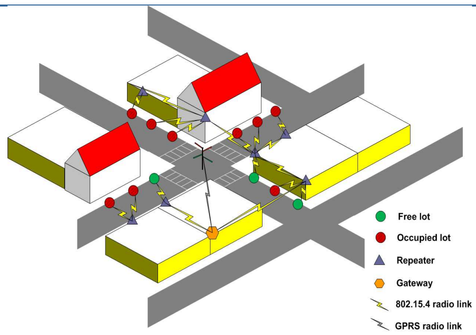


Parking Area Management

Parking areas equipped with sensors

- based on ferromagnetic technology
- buried under the asphalt in the main parking areas.
- provided with one transceiver
- send their parking state (free or occupied), to a gateway through the repeaters.

Guidance to Free Parking Slots



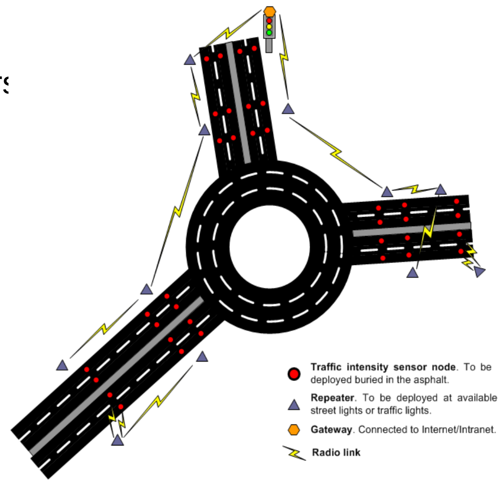
Data collected from parking sensors are used to guide drivers towards free slots, using an architecture divides in two parts:

- **Panel**: Receives information from the Central Station and shows the number of places available in a determined parking zone.
- **Central Station**: It receives, from the Portal Server, all data retrieved by the sensors already deployed to detect parking lots availability.

Traffic Intensity Monitoring

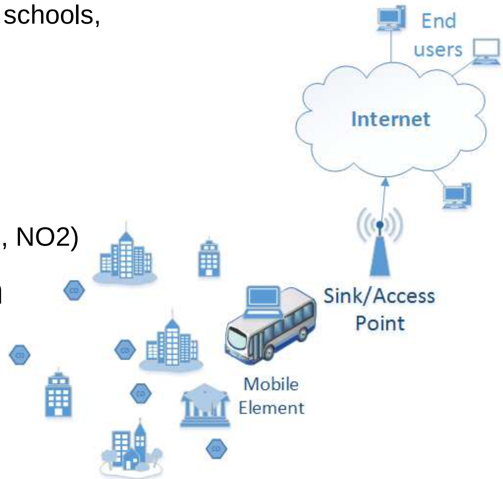
- Sensors buried under the asphalt at the main entrance of the city
- Measure traffic parameters
  - Traffic volumes
  - Vehicle Speed
  - Queue length

Architecture:
- Traffic Sensors
- Repeaters
- Gateway



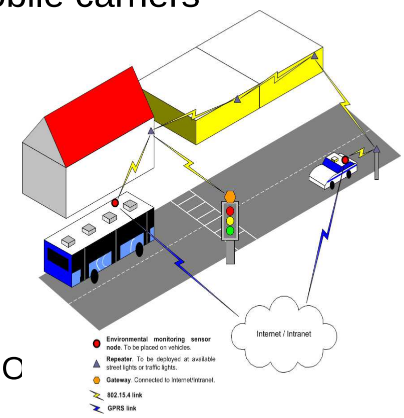
Mobile Environmental Monitoring

- Sensors deployed in strategic (fixed) points
  - Crosses, bus stops, homes, schools, …
- Measured Parameters
  - Temperature
  - Humidity
  - Air Pollution (PM10, CO, O3, NO2)
- Data collection through mobile elements
  - public buses, taxis
  - people



Mobile Environmental Monitoring

- Sensors deployed also on mobile carriers
  - public buses, police cars, taxis
  - people
  - robots
- Measured Parameters
  - Temperature
  - Humidity
  - Air Pollution (PM10, CO, O3, NC



Better coverage in a more efficient way

Platforms and Programming Issues

Sensor Platforms (Motes)

	Btnode 3	mica2	mica2dot	micaz	telos A	tmote sky	EYES
Manufacturer	Art of Technology	Crossbow	Crossbow	Crossbow	Imote iv	Imote iv	Univ. of Twente
Microcontroller	Atmel Atmega 128L	Atmel Atmega 128L	Atmel Atmega 128L	Atmel Atmega 128L	Texas Instruments MSP430	Texas Instruments MSP430	Texas Instruments MSP430
Clock	7.37 MHz	7.37 MHz	4 MHz	7.37 MHz	8 MHz	7.37 MHz	5 MHz
RAM (KB)	64 + 180	4	4	4	2	10	2
ROM (KB)	128	128	128	128	60	48	60
Storage (KB)	4	512	512	512	256	1024	4
Radio	Chipcon CC1000 315/433/868/916 MHz 38.4 Kbauds	Chipcon CC1000 315/433/868/916 MHz 38.4 Kbauds	Chipcon CC1000 315/433/868/916 MHz 38.4 Kbauds	Chipcon CC2420 2.4 GHz 250 Kbps IEEE 802.15.4	Chipcon CC2420 2.4 GHz 250 Kbps IEEE 802.15.4	Chipcon CC2420 2.4 GHz 250 Kbps IEEE 802.15.4	RFM TR1001868 MHz 57.6 Kbps
Max Range	150–300 m	150–300 m	150–300 m	75–100 m	75–100 m	75–100 m	75–100 m
Power	2 AA batteries	2 AA batteries	Coin cell	2 AA batteries	2 AA batteries	2 AA batteries	2 AA batteries
PC connector	PC-connected programming board	PC-connected programming board	PC-connected programming board	PC-connected programming board	USB	USB	Serial Port
OS	Nut/OS	TinyOS	TinyOS	TinyOS	TinyOS	TinyOS	PEEROS
Transducers	On acquisition board	On acquisition board	On acquisition board	On acquisition board	On board	On board	On acquisition board
Extras	+ Bluetooth						

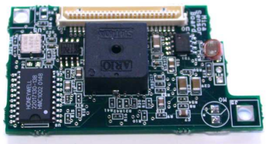
Paolo Baronti, P. Pillai, V. Chook, S. Chessa, A. Gotta, Y. Hu, **Wireless Sensor Networks: A Survey on the State of the Art and the 802.15.4 and ZigBee Standards**, *Computer Communications*, Vol. 30, 2007.

Mica Motes

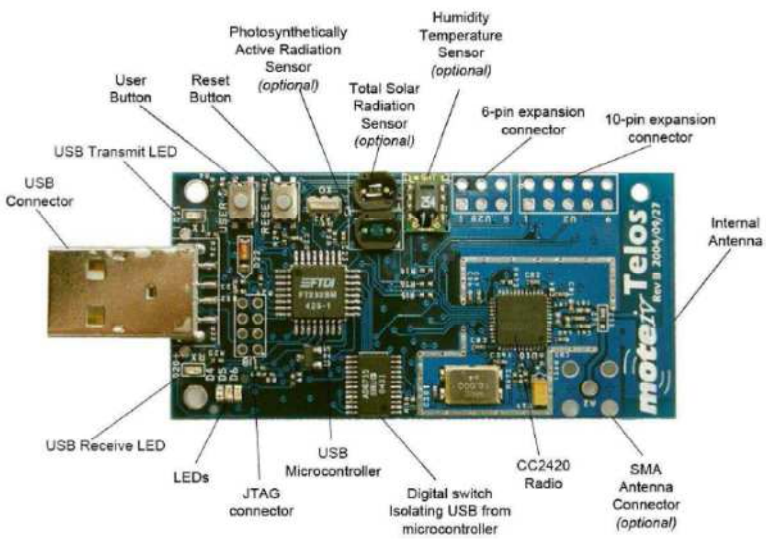
Mica2

Mica2dot

Sensor Board for Mica
light, temperature, accelerometer, magnetometer, microphone, tone detector, 4.5 Khz sounder



Telos/Tmote Sky Mote



Challenges to be addressed

- Driven by interaction with environment
  - Message arrival, sensor acquisition
  - Concurrency Management
    - Event arrival and data processing are concurrent activities
    - Potential bugs must be managed (e.g., race conditions)
- Limited Resource
  - Due to small size, low cost and low power consumption
- Reliability
  - Although single node may fail, we need long-lived applications
  - No recovery mechanism in field, except automatic reboot
- Soft real-time requirements

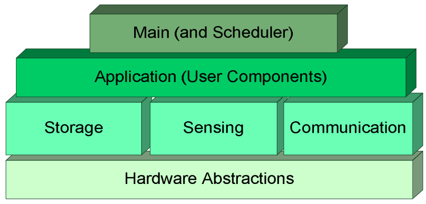
The TinyOS Operating System

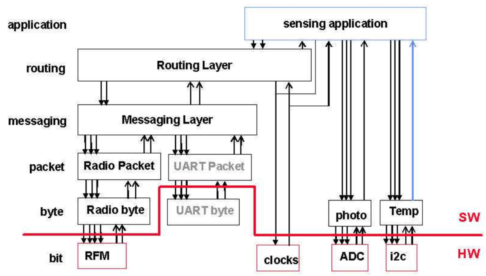
- Specifically targeted to wireless sensor networks
  - Component-based architecture
  - Event-based concurrency
  - Split-phase operation

http://www.tinyos.net/

TinyOS Architecture

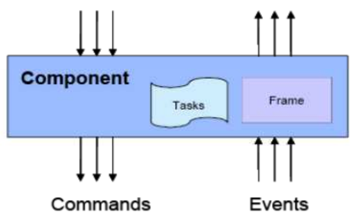
- Component-based architecture
- Components
  - Software modules
  - Hardware modules
  - The distinction is invisible to developers
- Unused Services
  - Excluded from the application





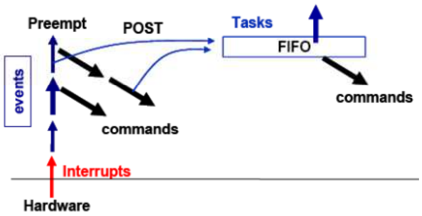
Component-based computation model

- Component
  - computing entity
    - interface (commands and events)
    - frame (private variables)
- Computing abstractions
  - command
    - service request to a component
    - non-blocking
  - event
    - command completion, message or interrupt
  - task
    - context of execution (~function)
    - run to completion, preemption by event



Concurrency Management

- Two sources of concurrency
  - Tasks and Events (interrupts)
  - Components can post a task
  - The post operation returns immediately
- Task scheduling
  - FIFO (non pre-emptive)
- Tasks run to completion
  - Can be pre-empted only by events
- Events run to completion
  - Signify either an external event (message) or completion of a split-phase operation
  - May pre-empt tasks



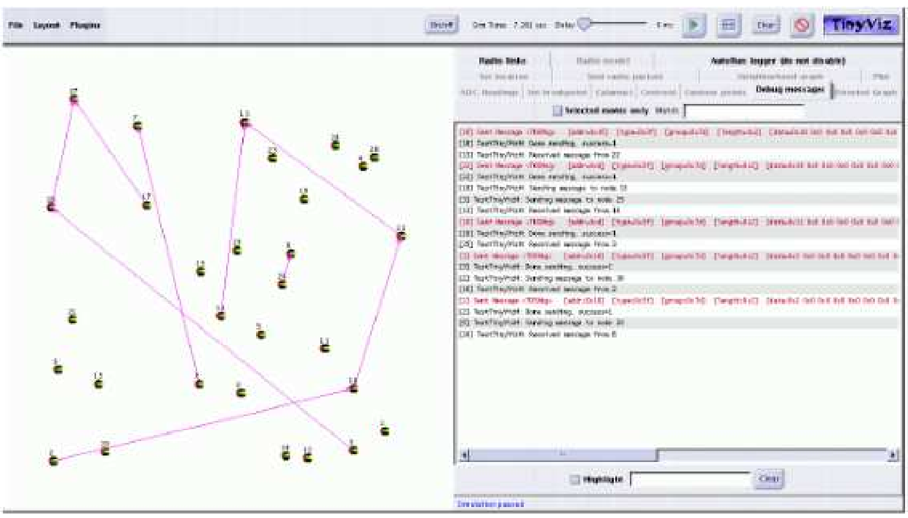
Split-phase Operations

- Due to task execution model (non pre-emptive)
- Split-phase
  - Operation request/completion are separate functions
  - If an operation is split phase
    - the command returns immediately
    - Completion is signaled with an event
- Example: message transmission
  - Application component: send command
  - communication component: sendDone event

TinyOS development environment

- nesC language
  - extension to the C language
  - definition of interfaces
  - abstraction between definition and composition of components
- nesC compiler and OS source
  - composition of the component graph (at compilation time)
  - TinyOS computational model (additional checks)
- TOSSIM simulator
  - same code runs in actual nodes and simulator
  - flexible models for radio and sensors
  - scripting (Tython), graphical interface (TinyViz)

TinyViz



Tutorials on TinyOS Programming Environment

- TinyOS/TOSSIM Tutorial, http://docs.tinyos.net/index.php/TinyOS_Tutorials
- TinyOS Reference Manual, http://www.tinyos.net/tinyos-2.x/doc/pdf/tinyos-programming.pdf
- D. Gay et al., "The nesC Language: A Holistic Approach to Networked Embedded Systems", 2002.
- nesC Reference Manual, http://www.tinyos.net/dist-2.0.0/tinyos-2.0.0beta1/doc/nesc/ref.pdf

Contiki Operating System

- Limited Memory Footprint
- Event-driven Kernel
- Portability
  - Many different platform supported
    - Tmote Sky, Zolertia, RedBee, etc
- C Programming
- Academic and Industrial support
  - Cisco and Atmel are part of the *Contiki project*

Contiki Operating System

- Protothread (optional multi-threading)
- Dynamic Memory Allocation
- TCP/IP stack (uIP)
  - Both IPV4 and IPv6
- Power profiling
- Dynamic loading and over-the-air programming
- IPsec
- On-node database Antelope
- Coffee file system
- …

Contiki Operating System

- Prototread example
  - Stop-and-wait sender

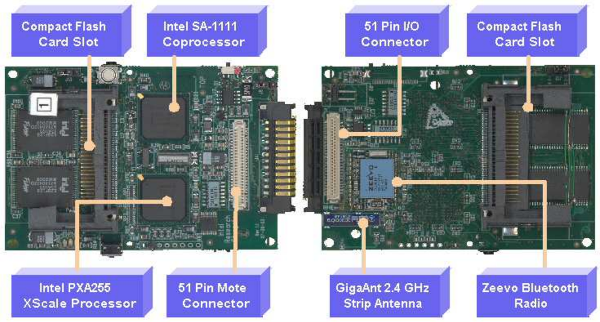
```
PROCESS_THREAD(reliable_sender, …) {
  PROCESS_THREAD_BEGIN();

  do {
    PROCESS_WAIT_UNTIL(data_to_send());
    send(pkt);
    timer_start();
    PROCESS_WAIT_UNTIL((ack_received() || timer_expired());
  } while (!ack_received());

  PROCESS_THREAD_END();
}
```

References

- A. Dunkels, B. Gronvall, T. Voigt, "Contiki - a lightweight and flexible operating system for tiny networked sensors", IEEE International Conference on Local Computer Networks, 16-18 November 2004
- Contiki: The Open Source OS for the Internet of Things, http://www.contiki-os.org/
  http://en.wikipedia.org/wiki/Contiki
- Contiki, Processes. Available Online at: https://github.com/contiki-os/contiki/wiki/Processes

Microserver-class Node: Stargate

- Embedded platform from Intel
- Compute engine: PXA255 (32bit, 2.3 nJ/instruction, 200 MHz, 1.5V), several power management options
- Communication: built-on Bluetooth, 802.11 via PC or CF connector, and Mica2 or MicaZ mote via mote connector
- Software: Compaq bootloader, Linux 2.4 series kernel

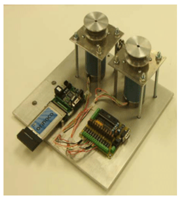


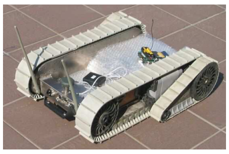
Applications of Stargate-class nodes

- Seismic monitoring
- Personal exploration rover
- Mobile micro-servers
- Networked info-mechanical systems
- Hierarchical wireless sensor networks

[Robotics, CMU]

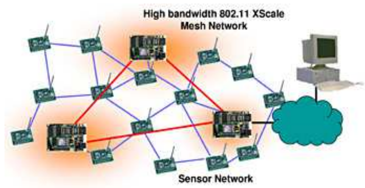
[NESL, UCLA]





[NIMS, UCLA]

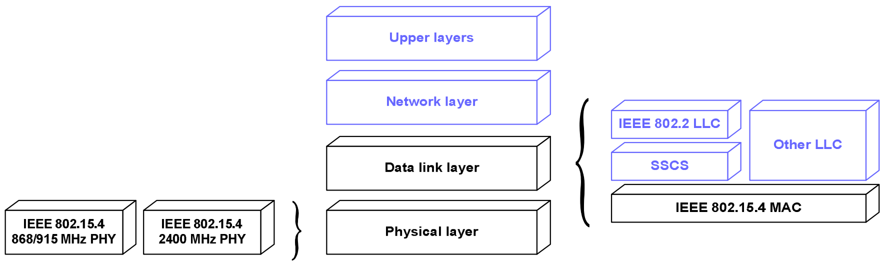
[CENS, UCLA]



[Intel + UCLA]

Networking Issues

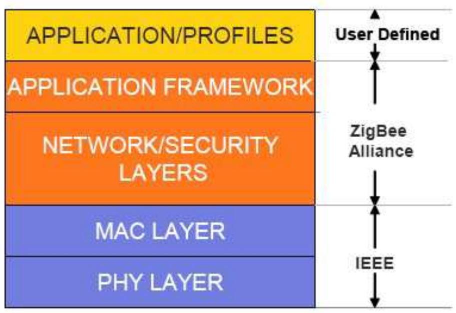
IEEE 802.15.4/ZigBee standard



- Standard for Personal Area Networks (PANs)
  - low-rate and low-power
  - PHY and MAC layers
- Main features
  - transceiver management
  - channel access
  - PAN management

IEEE 802.15.4 and ZigBee

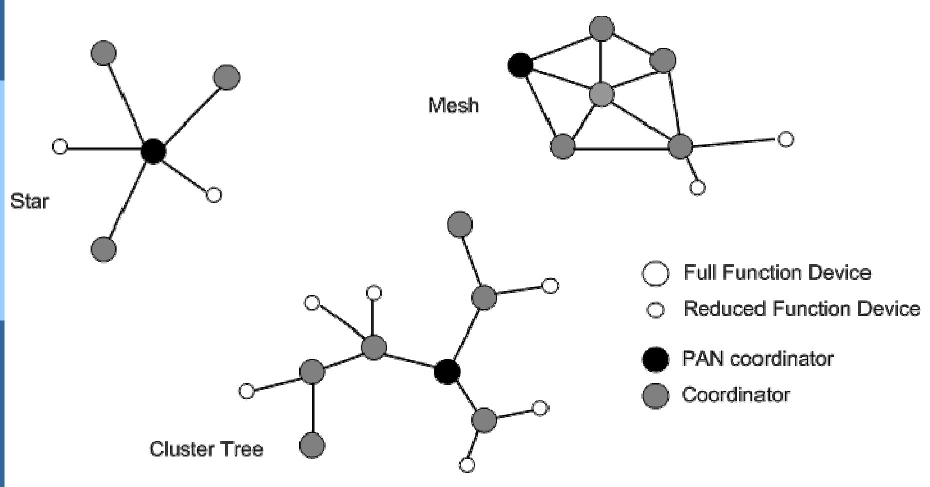
- IEEE 802.15.4 standard
  - Low-rate, Low-power, Low-cost Personal Area Networks (PANs)
  - PHY and MAC layers
- ZigBee Specifications
  - Upper Layers



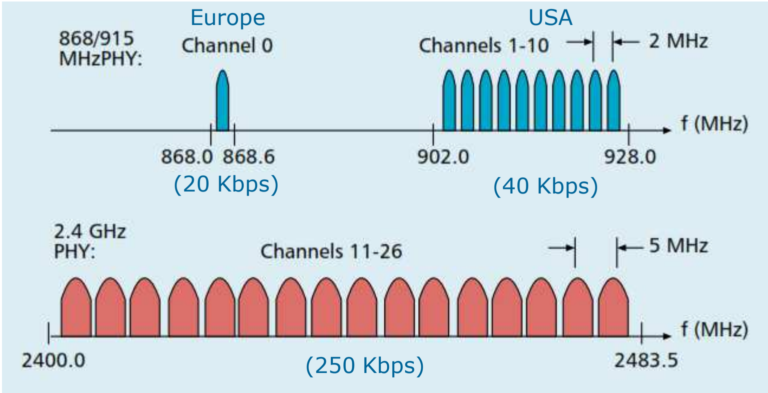
IEEE 802.15.4 components

- Full Function Device (FFD)
  - implements the full set of standard functionalities
  - PAN coordinator
  - coordinator
    - broadcasts beacons
    - clock synchronization
- Reduced Function Device (RFD)
  - implements a minimal set of standard functionalities
  - cannot be a (PAN) coordinator
  - can only communicate with a FFD

IEEE 802.15.4/ZigBee Network Topologies



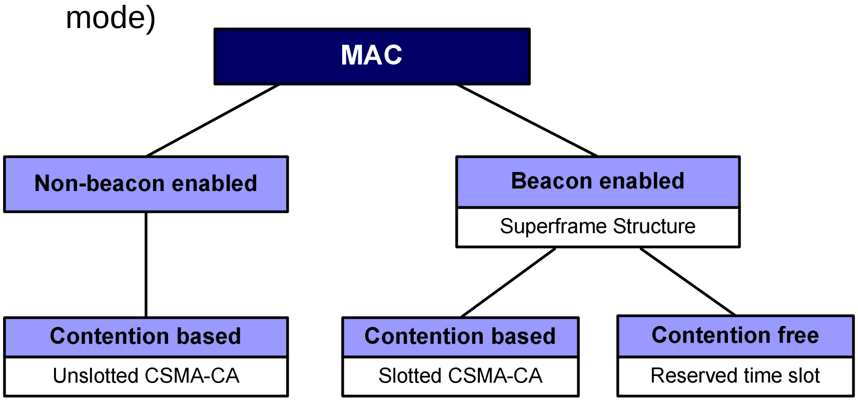
Channel frequencies



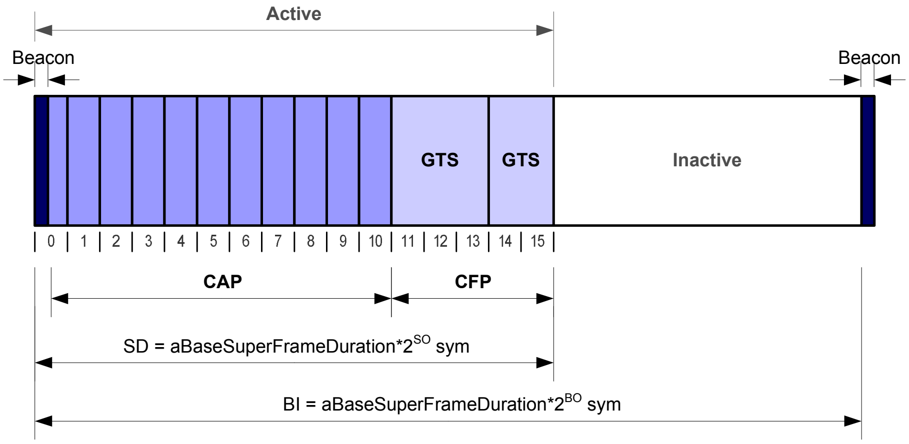
Channel number	Channel center frequency (MHz)
$k = 0$	868.3
$k = 1, 2, \ldots, 10$	$906 + 2(k - 1)$
$k = 11, 12, \ldots, 26$	$2405 + 5(k - 11)$

IEEE 802.15.4: MAC protocol

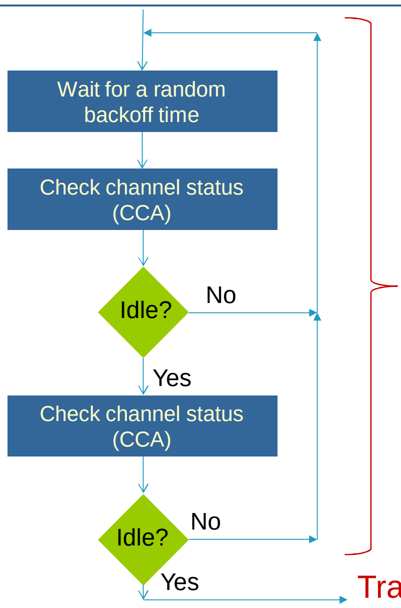
- Two different channel access methods
  - Beacon-Enabled duty-cycled mode
  - Non-Beacon Enabled mode (aka Beacon Disabled mode)



IEEE 802.15.4: Beacon Enabled mode



IEEE 802.15.4: Beacon Enabled mode



At each trial the backoff-window size is doubled

Only a limited number of attempts is permitted (***macMaxCSMABackoffs***)

Transmission

Acknowledgement Mechanism

- Optional mechanism
- Destination Side
  - ACK sent upon successful reception of a data frame
- Sender side
  - Retransmission if ACK not (correctly) received within the timeout
  - At each retransmission attempt the backoff window size is re-initialized
  - Only a maximum number of retransmissions allowed (*macMaxFrameRetries)*

Limits of 802.15.4

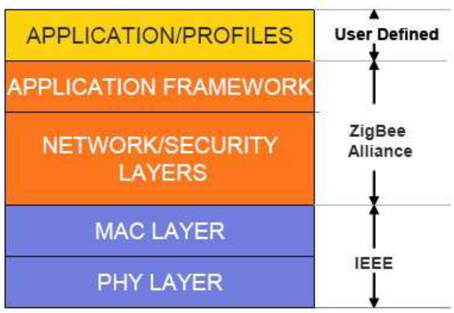
- Unbounded latency
  - Due to contention-based CSMA/CA algorithm
- Limited Reliability
  - Due to default CSMA/CA parameter values
- No guaranteed bandwidth
  - Unless GTS is used
  - GTS only provides a limited service (7 slots)
- No built-in frequency hopping technique
  - Prone to failure due to interferences and multi-path fading

IEEE 802.15.4e

- IEEE 802.15 Task Group 4e
  - chartered to define a MAC amendment to the existing standard 802.15.4-2006.
  - The intent of this amendment was to enhance and add functionality to the 802.15.4-2006 MAC
    - better support the industrial markets
    - permit compatibility with modifications being proposed within the Chinese WPAN.
  - On February 6, 2012 the IEEE Standards Association Board approved the IEEE 802.15.4e MAC Enhancement Standard document for publication.
    - http://www.ieee802.org/15/pub/TG4e.htmle WPAN

IEEE 802.15.4 and ZigBee

- IEEE 802.15.4 standard
  - Low-rate, Low-power, Low-cost Personal Area Networks (PANs)
  - PHY and MAC layers
- ZigBee Specifications
  - Upper Layers



ZigBee Alliance

PerLab

- **Independent, neutral, nonprofit corporation**
  - Created in in 2002
- **Open and global**
  - Anyone can join and participate
  - Membership is global
- **Activities**
  - Specification creation
  - Certification and compliance programs
  - Branding, market development, and user education

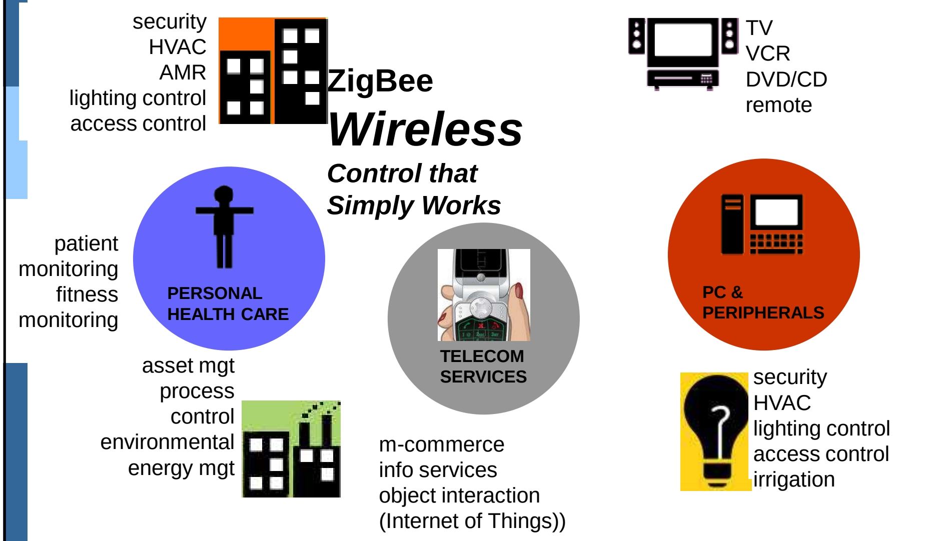
ZigBee™ Alliance

ZigBee Promoters

PerLab

EATON
ember
freescale semiconductor (Launched by Motorola)
Honeywell
HUAWEI
MOTOROLA
MITSUBISHI ELECTRIC Changes for the Better
PHILIPS
SAMSUNG
SIEMENS
Schneider Electric
STMicroelectronics
TEXAS INSTRUMENTS
TENDRIL

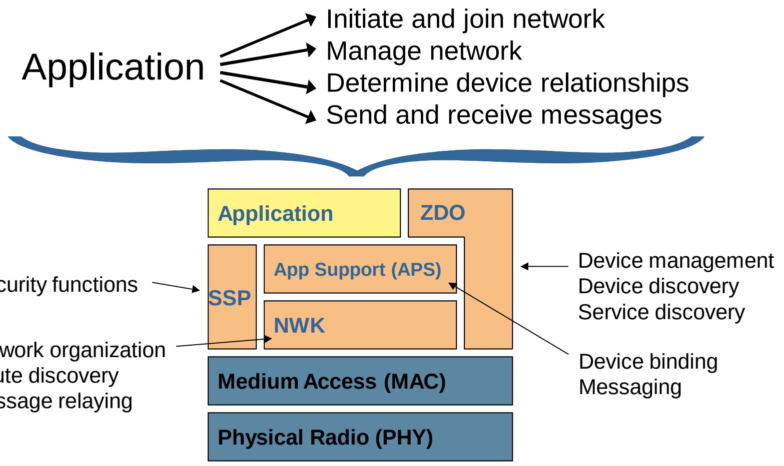
Application Areas



ZigBee Standards

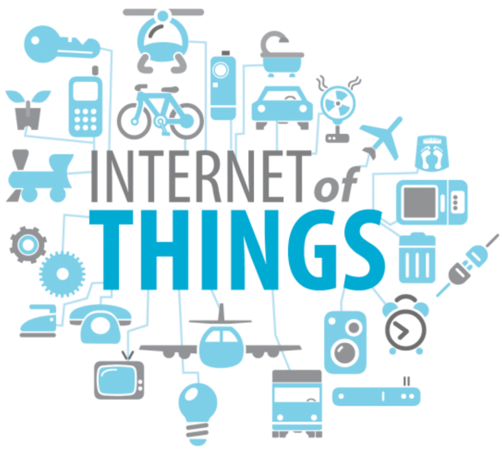
- ZigBee Building Automation (Efficient commercial spaces)
- ZigBee Remote Control (Advanced remote controls)
- ZigBee Smart Energy (Home energy savings)
- ZigBee Health Care (Health and fitness monitoring)
- ZigBee Home Automation (Smart homes)
- ZigBee Input Device (Easy-to-use touchpads, mice, keyboards, wands)
- ZigBee Light Link (LED lighting control)
- ZigBee Retail Services (Smarter shopping)
- ZigBee Telecom Services (Value-added services)

ZigBee Stack



Internet of Things

"The next logical step in the technological revolution connecting people anytime, anywhere is to connect inanimate objects.
This is the vision underlying the **Internet of things: anytime, anywhere, by anyone and anything"** (ITU, Nov. 2005)



Internet of Things

- The Internet penetrates in embedded computing.
- The Internet of Things envisions a network of objects, where all things are uniquely and universally addressable, identified and managed by computers in the same way humans can.

Questions?