

Design principle for wsn:-

The design method aims at the definition of simple algorithms that are easily implemented on resources constrained sensor nodes. These algorithms allows the network to meet the reliability and latency required by the control application while minimizing for energy consumption

principle for wsn's:-

→ Distributed organization:-

1) The wsn's nodes should cooperatively organize the network, using distributed algorithms and protocols, self-organization is a commonly used term for this principle.

2) when organizing a network in a distributed fashion, it is necessary to be aware of potential shortcomings of this approach

→ In Network processing (Aggregation):-

1) The simplest-network processing technique is aggregation

2) The name aggregation stems from the fact that in nodes intermediate b/w source and sinks, informati.

is aggregated into a condensed form out of information provides by nodes further away from the sink

→ Adaptive fidelity & accuracy

→ Data centrality

→ Exploit location information

→ Exploit activity patterns

2) Gate way concept need for gateway :-

Gate way :-

All networks have a boundary that limits communication to devices that are directly connected to it

Due to this, if a network wants to communicate with devices, nodes or networks outside of that boundary, they require the functionality of a gateway. A gateway is often characterized as being

the combination of a router and a modem

Need for gateway :-

The gateway acts as a bridge between the wsn and the other network. This enables data to be

stored and processed by devices with more

resources, for example, in a remotely located

server. A wireless wide area network used

primarily for low-power devices is known as a

low

* The link between two computers to connect to internet or another network is called gateway.

The gateway works like a portal among two programs by means of communication b/w protocol and permit them to share data on same computer or among different computers.

* Gateways provides full protocol conversion from one proprietary LAN Technology to another, i.e. Ethernet to token ring or FDDI or any other standard or protocol rather than encapsulation

* It uses high layers of the OSI model, perhaps, through layer 7, the application layer.

3) WSN to Internet Communication:-

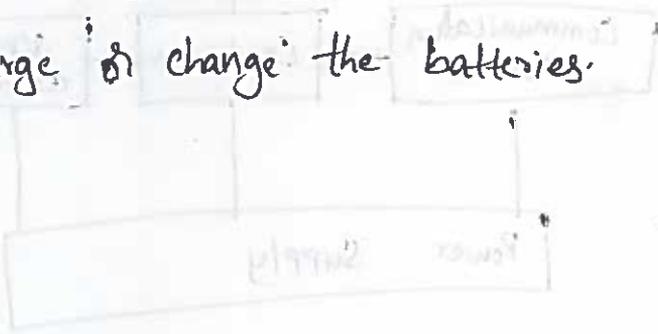
* The wireless sensor Internet of things fall

* WSN outline introduction mote Revolution wireless sensor network - Tiered application architectures Dynamic Cluster Formation

Power-Aware MAC protocol S-MAC, T-MAC, CPL, X-MAC

The internet of things & internet of things wsn

* WSN a distributed connection of nodes that coordinate to perform a common task. In many applications, the nodes are battery powered and it is often very difficult to recharge or change the batteries.



* Environment monitoring great duck Island 150 Sensing nodes deployed throughout the island relay data temp, pressure & humidity to a central device.

* Power measurements \rightarrow Internet of things wireless sensor network

* Tiered WSN architectures 33 Internet of things wireless sensor network

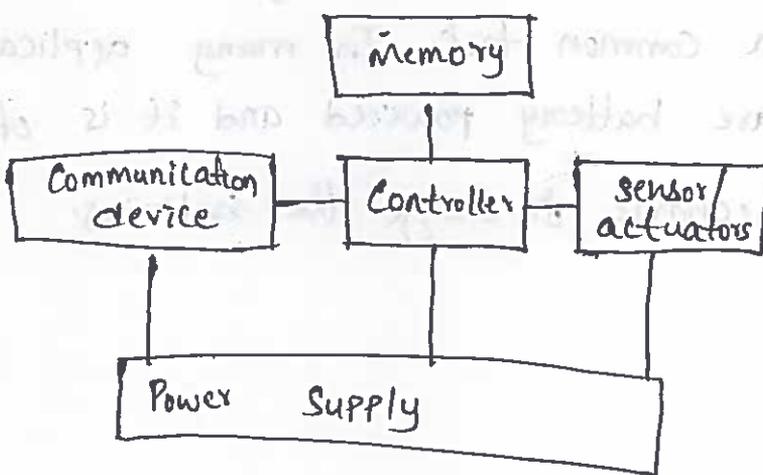
* WSN Summary introduction Note Revolution wireless sensor network Application WSN details Types of wireless sensor network

e) Single Node Architecture

Hardware Components

\rightarrow Choosing the hardware components for wireless sensor node has to consider size, cost and energy consumption of the nodes

\rightarrow A basic sensor node contains five main components such as Controller, memory, sensor and actuators, communication device



Sensor node hardware components

Controller:-

* An field programmable gate array (FPGA) can be reprogrammed "in the field" to adapt to a changing set of requirements, this can take time and energy

* An application specific integrated circuit (ASIC) is a specialized processor, custom designed for a given application only.

Memory:-

* Memory is required to store programs and intermediate data; usually, different types of memory are used for programs and data.

* In WSN there is a need for random access memory to store intermediate sensor readings, packets from other nodes and so on.

* RAM is fast, its main disadvantage is that it loses its content if power supply is interrupted.

Sensors and actuators:-

The actual interfaces for the physical world: The devices that can observe or control physical parameter of the environment.

Sensor:-

Sensor can be roughly categorized into three

Categories as

- Passive quantity at the point of the sensor node
- Passive omnidirectional sensors: These sensor can measure a physical quantity at the point of the sensor node without actually manipulating the environment by active probing
- Passive narrow-beam sensors: These sensor are passive as well but have a well-defined notion/area idea of direction of measurement
- Active sensors: These actively analyses the environment. For example, a radar sensor or some type of seismic sensors.

Actuators

→ Actuators are just about as diverse as sensors, yet for the purposes of designing a system that converts electrical signal into physical phenomenon.

Communication device:

To turn node into a network a device is required for sending and receiving information over a wireless channel.

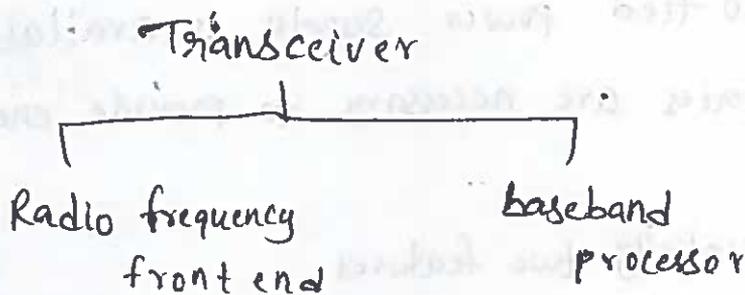
Transmission medium selection:-

The communication device is used to exchange data b/w individual nodes

In some cases, wired communication can actually be the method of choice & is frequently applied in many sensor networks.

Transceivers:

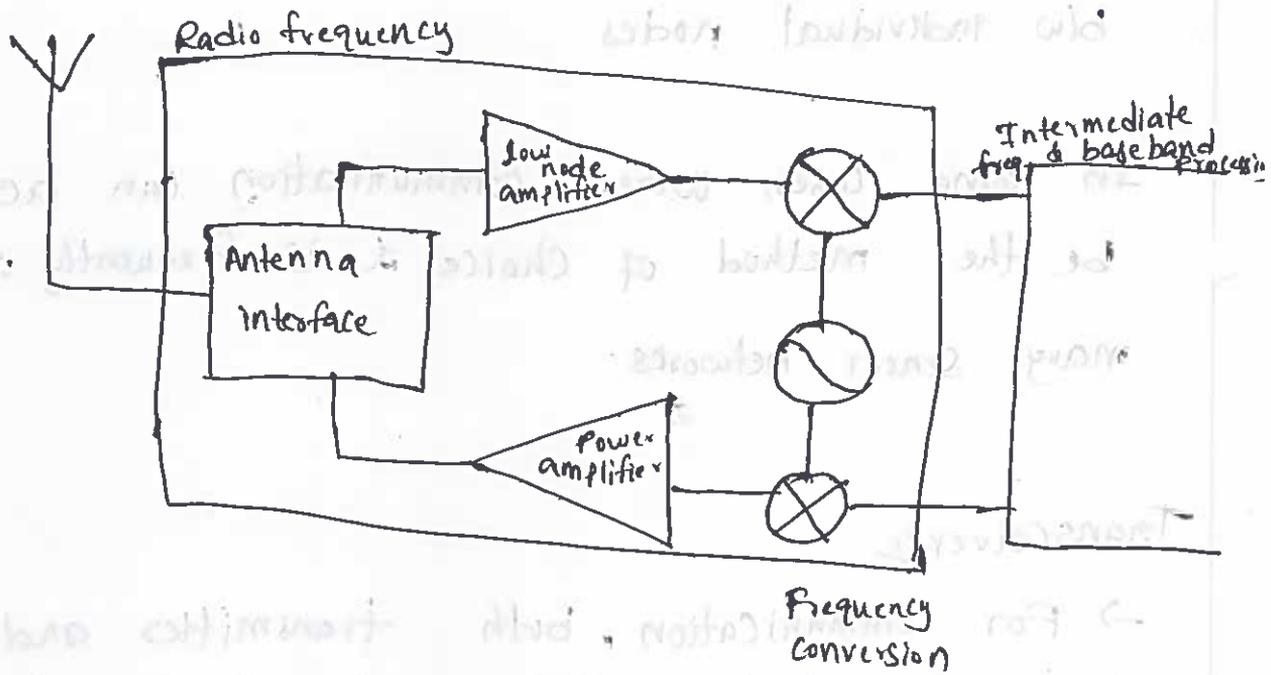
→ For communication, both transmitter and receiver are required in a sensor node to convert a bit stream coming from a microcontroller & convert them to and from radio waves.



* RF Front end structure

→ The power Amplifier (PA) accepts signals from the RF or base band part and amplifies them for transmission over the antenna

→ Low noise amplifier (LNA) amplifies incoming signals up to levels suitable for further processing without significantly reducing the SNR



RF Front end structure

Power Supply:-

As usually no tied power supply is available, some form of batteries are necessary to provide energy.

These are essentially two features

1. Storing energy
2. Energy Scavenging

* Micro Controller energy Consumption

For Controller, typical states are "active", "idle" and "Sleep"

Energy saving in microcontroller

$$E_{\text{saved}} = (t_{\text{event}} - t_1) P_{\text{active}} - (N_{\text{down}} (P_{\text{active}} + P_{\text{sleep}}) / 2 + (t_{\text{event}} - t_1 - N_{\text{down}}) P_{\text{sleep}})$$

* The energy overhead is denoted by

$$E_{\text{overhead}} = T_{\text{up}} (P_{\text{active}} + P_{\text{sleep}}) / 2$$

Memory Energy Consumption

→ The most relevant kinds of memory are on-chip memory and flash memory.

→ off-chip RAM is rarely used. In fact, the power needed to drive on-chip memory is usually included in the power

consumption number given for the controllers.

→ Hence, writing to flash memory can be a time & energy consuming task that is best avoided if somehow possible.

Storing energy: Batteries

Traditional batteries: The power source of a sensor node is a battery, either non-rechargeable or rechargeable.

Batteries requirements are

- Capacity;
- Capacity under load;
- Self-discharge;
- efficient recharging;
- Relaxation;

Energy Scavenging:

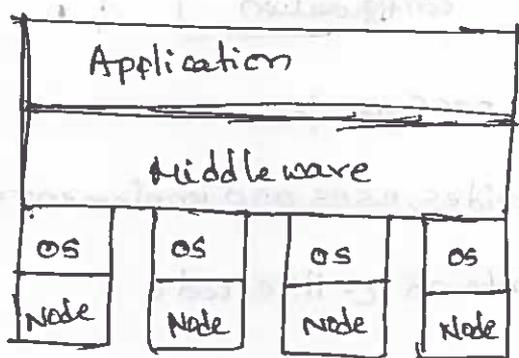
- * Depending on application, high capacity batteries that last for long times with negligible self-discharge rate.
- * For this energy scavenging is used is the process of recharging the battery with energy gathered from the environment like solar cells or vibration-based power generation.

Energy Consumption of Sensor nodes

- * Energy is supplied to a sensor node through battery and recharging by energy scavenging. Hence, the energy consumption of a sensor node must be tightly controlled.

Operating System and execution environment

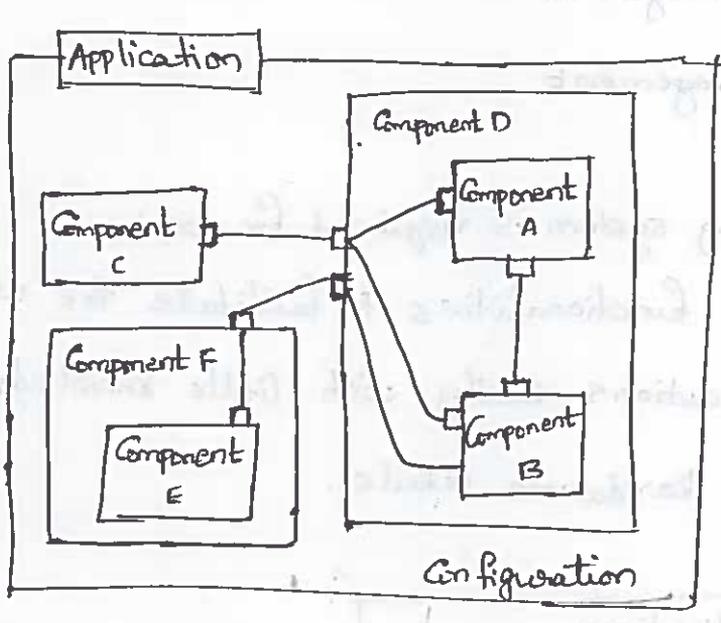
- The basic functionality of an operating system is to hide the low-level details of the sensor node by providing a clear interface to the external world.
- The low-level services provided by an operating system:
 - processor management
 - Memory Management
 - Device management
- A suitable operating system is required for WSN to provide these functionalities to facilitate the user in writing applications easily with little knowledge of the low-level hardware details.



- WSN operates at two levels!
 - Network level
 - Node level

- Application: writing Components using Configurations.
- Whole program analysis & optimization.
 - detect race conditions.
- static language.
 - No dynamic memory allocation.
- No multiprocessing.

nesC Model :



Components types of nesC are :-

Modules → provides, uses and implements

- Contains C-like code.

Configurations → • A collection of other Components put together by "wiring"

Syntax :

- Does not use C-like code.

• Network level intersects are:

- Connectivity
- Routing
- Communication channel characteristics
- Protocols.

• Node level intersects are:

- Hardware, Radio
- CPU
- Sensors.

At a higher level OS for WSN can also be classified as:

- ✓ Node-level
- ✓ Network-level

Design characteristics of OS for WSNs :-

- Flexible Architecture
- Efficient execution model
- Clear Application Programming Interface
- Reprogramming
- Resource management.

OS classification :-

Monolithic	Modular	VM
TinyOS	SOS	VMSTAR
MagnetOS	Contiki MantisOS GORMOS Beatha	Nate MagnetOS Contiki VM

Design issues of OS :-

1. Should be compact and small in size.
2. The information should be collected and reported as quickly as possible.
3. The CPU time and limited memory must be scheduled and allocated for processes carefully to guarantee fairness.
4. Should support power management resource.

VM	Virtualization	Virtualization
System	OS	OS
Hardware	Hardware	Hardware
Software	Software	Software

Introduction to TinyOS and nesC

- TinyOS: operating system for sensor networks
- nesC: programming language for sensor networks.

Why TinyOS?

- Traditional OSes are not suitable for networked sensors.
- Characteristics of networked sensors:
 - small physical size & low power consumption
 - software must make efficient use of processor & memory, enable low power communication
- Concurrency intensive.
 - simultaneous sensor readings, incoming data from other nodes.
 - Many low-level events, interleaved w/high-level processing.
- limited physical parallelism (few controllers, limited capability)
- Diversity in design & usage
 - software modularity - application specific

TinyOS solution

- Support Concurrency
 - event-driven architecture
- software modularity
 - application = scheduler + graph of Components
 - A Component contains Commands, event handlers,
- Efficiency: get done quickly and then sleep
- static memory allocation

TinyOS Computational Concepts.

- 1) Events - Time Critical, short duration
- 2) Commands - Can call lower-level Commands
- 3) Tasks - Can be preempted by events.

nesC

Main features

- Support and reflect TinyOS's design
- Support Components, event-based concurrency model
- Extending C with support for Components
 - Components provide and use interfaces.