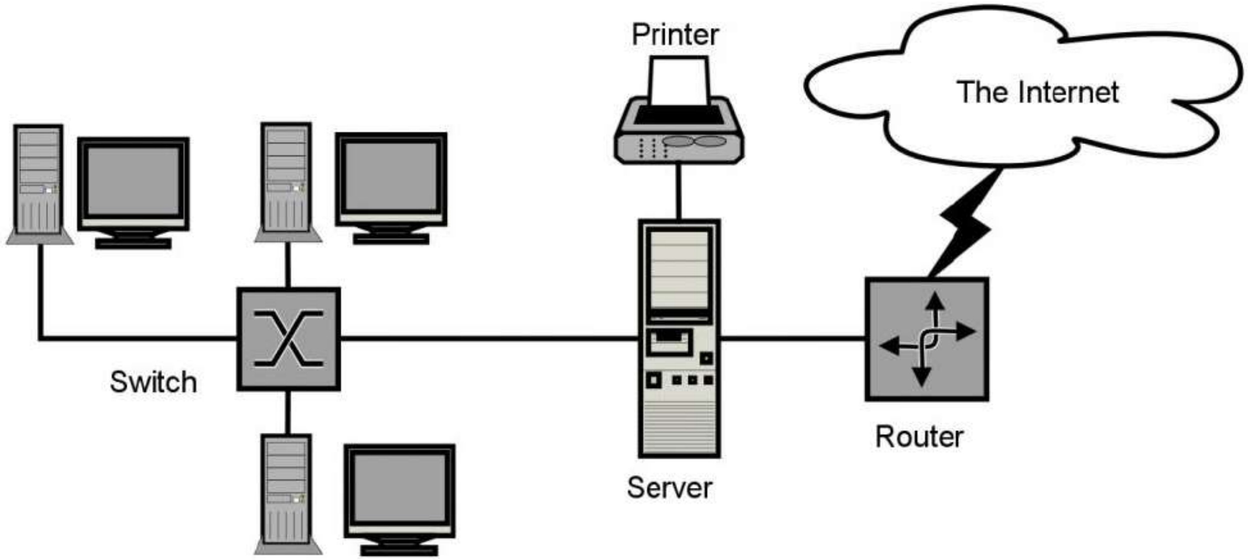
WIRELESS SENSOR NETWORKS

COMPUTER NETWORK

- A **computer network** is any group of interconnected computing devices capable of sending or receiving data. A **computing device** isn't just a computer—it's any device that can run a program, such as a tablet, phone, or smart sensor.

- The computing devices are connected to each other through communication media like Co-axial Cable, CAT 6 Cable, Fiber Optic Cable, Radio Waves, Microwaves etc.

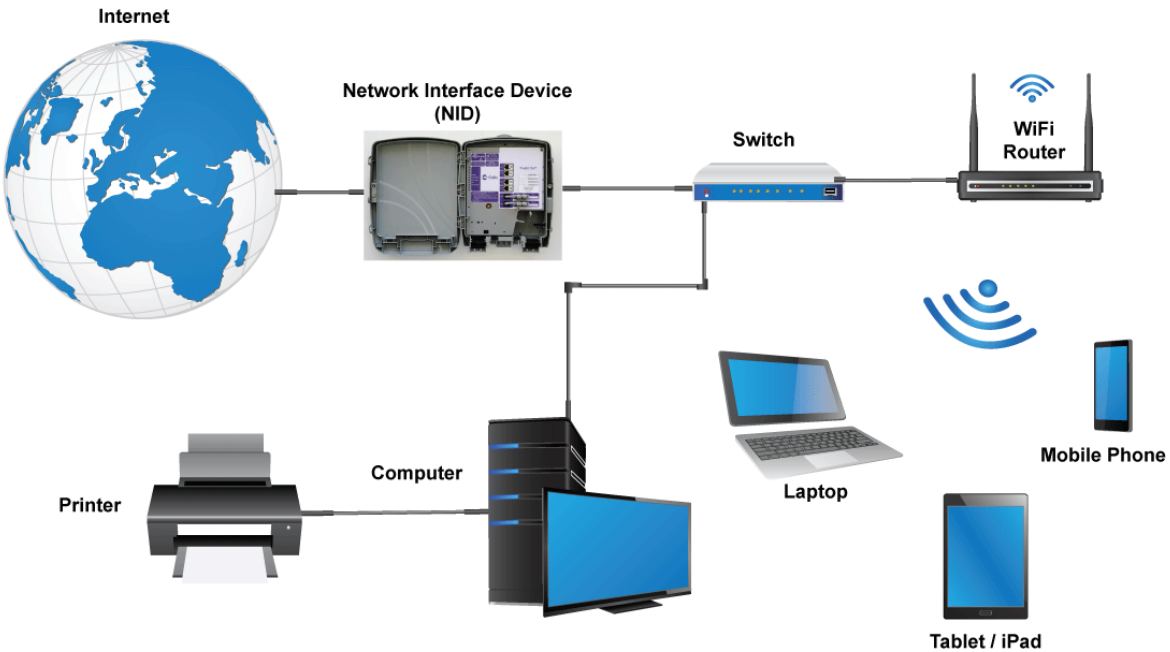
- Switch, Router, Hub and NICs (Network Interface Cards) are used to form computer network.



WIRED Vs WIRELESS NETWORK

- A **wireless local-area network (LAN)** uses radio waves to connect devices such as laptops and mobile phones to the Internet and to your business network and its applications.

- A **wired network** uses cables to connect devices, such as laptop or desktop computers, to the Internet or another network. A wired network has some disadvantages when compared to a wireless network. The biggest disadvantage is that your device is tethered to a router. The most common wired networks use cables connected at one end to an Ethernet port on the network router and at the other end to a computer or other device.



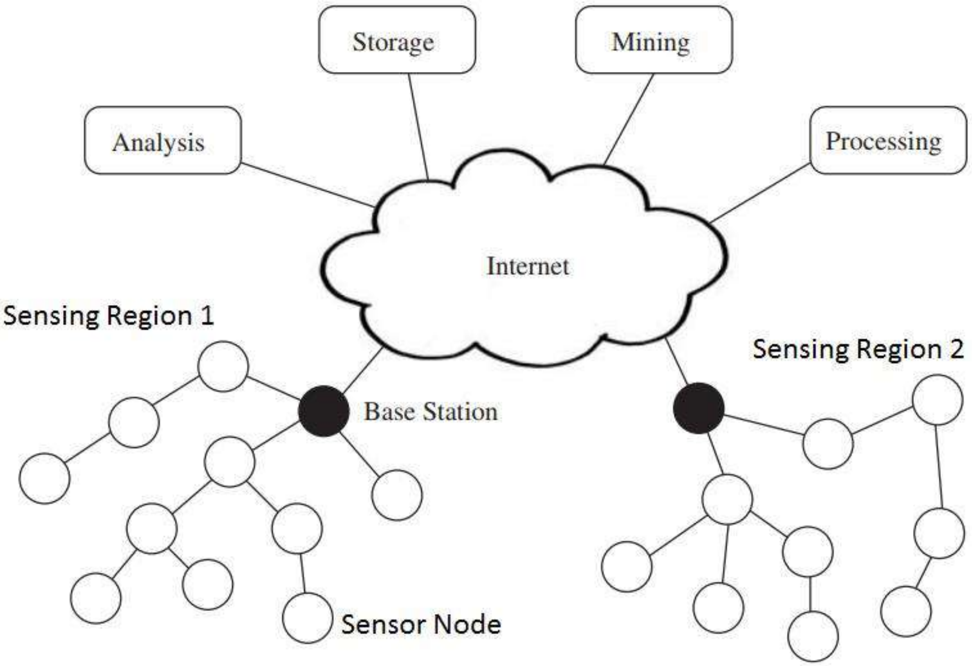
Wireless Sensor Networks

- A **wireless sensor network** is an infrastructure-less wireless network that is deployed in a large number of wireless sensors in an ad-hoc manner that is used to monitor the system, physical or environmental conditions.

- Sensor nodes are used in WSN with the onboard processor that manages and monitors the environment in a particular area. They are connected to the Base Station which acts as a processing unit in the WSN System.

- **Base Station** in a WSN System is connected through the Internet to share data.

- WSN can be used for processing, analysis, storage, and mining of the data.



Wireless Sensor Networks

- A **Wireless sensor network (WSN)** is a collection of sensor nodes that distributed in an arbitrary manner to solve a particular problem. The position of the node is predefined and based on random nature. Each node directly or indirectly connected with the base station (BS). BS is used to control and manages all sensor nodes.

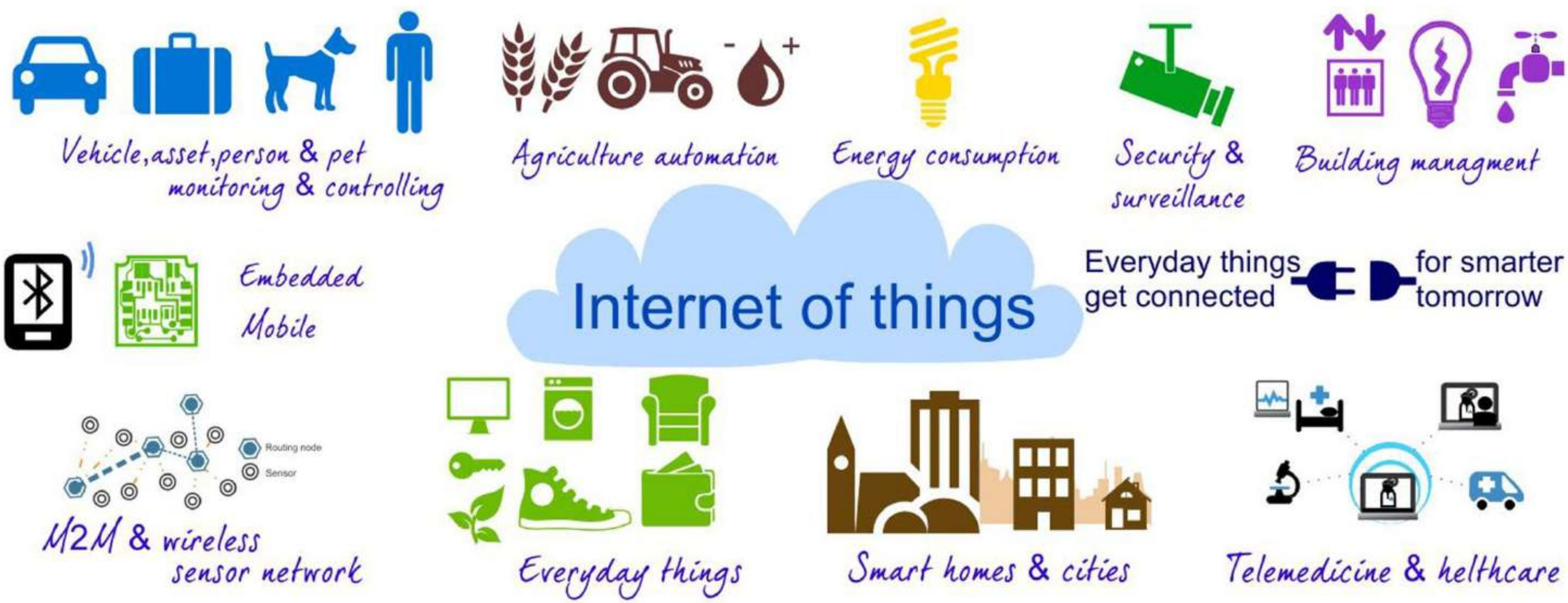
- A Wireless Sensor Network consists of Sensor Nodes that are deployed in high density and often in large quantities and support sensing, data processing, embedded computing and connectivity.

- WSN is a wireless network that consists of base stations and numbers of nodes (wireless sensors). These networks are used to monitor physical or environmental conditions like sound, pressure, temperature, and co-operatively pass data through the network to the main location.

IoT (Internet of Things)

- The **Internet of Things (IoT)** describes the network of physical objects—"things"—that are embedded with sensors, software, and other technologies for the purpose of connecting and exchanging data with other devices and systems over the internet.

- The **Internet of Things** is the concept of connecting any device (so long as it has an on/off switch) to the Internet and to other connected devices. The IoT is a giant network of connected things and people – all of which collect and share data about the way they are used and about the environment around them.



IoT (Internet of Things) Vs Wireless Sensor Networks (WSN)

IoT	WSN
In an IoT system, all of the sensors directly send their information to the internet. For example, a sensor may be used to monitor the temperature of a body of water. In this case, the data will be immediately or periodically sent directly to the internet, where a server can process the data and it can be interpreted on a front-end interface.	Conversely, in a WSN, there is no direct connection to the internet. Instead, the various sensors connect to some kind of router or central node. A person may then route the data from the router or central node as they see fit. That being said, an IoT system can utilize a wireless sensor network by communicating with its router to gather data.
IoT exists at a higher level then WSN. In other words, WSN is often a technology used within an IoT system.	A large collection of sensors, as in a mesh network, can be used to individually gather data and send data through a router to the internet in an IoT system.
A fridge with the capability of sending temperature reading to the internet is unlikely to use a wireless sensor network but it is an IoT device.	A large collection of sensors used to monitor precipitation on an acre of land is likely to be considered a "wireless sensor network" if in fact all the sensors are wireless. This system may or may not be connected to an IoT system.

Wireless Sensor Network

Why Sensors?

- A sensor is a connection between the physical environment to the digital world for collecting and finding the real world information and convert them into a digital form or signal form to process, store and perform.

- Sensors can be integrated with the devices, machines and environment for various befits.

- Sensors are manufactured and developed by the advancement of the new technologies such as VLSI (Very Large Scale Integration), MEMS (Microelectromechanical System) and Wireless Communication.

- Sensors and Microcontrollers are tiny, low powered and inexpensive miniature devices.

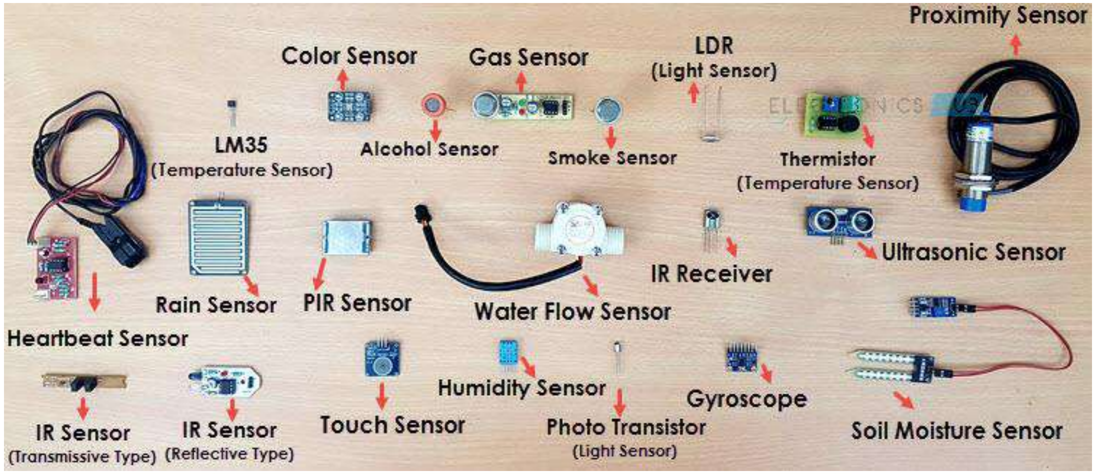
- Sensors are used in applications of numerous areas such as
  - Structural Monitoring
  - Forecasting of Environmental Pollution and Flooding
  - Weather Monitoring
  - Precision Agriculture
  - Active Volcano Monitoring
  - Smart Transportation
  - Health Care Monitoring
  - Underground Mining
  - Tracking Chemical Plumes

What is Sensing?

- Sensing is a technique to collect the information about the object, process and any event from the surrounding environment and convert them into readable form for the further processing.

What is Sensor?

- An object or device which is used to perform the sensing task called Sensor.
- A sensor is a device that measures physical input from its environment and converts it into data that can be interpreted by either a human or a machine.
- Most sensors are electronic (the data is converted into electronic data), but some are more simple, such as a glass thermometer, which presents visual data. People use sensors to measure temperature, gauge distance, detect smoke, regulate pressure and a myriad of other uses.



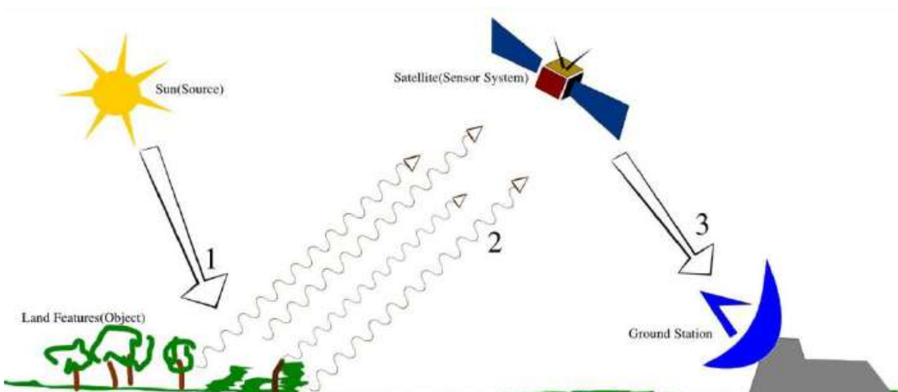
Wireless Sensor Network

Remote Sensor and Remote Sensing

- A Remote Sensor is a device that don’t touch the monitored object to gather the information and converts it into data that can be interpreted by either a human or a machine.

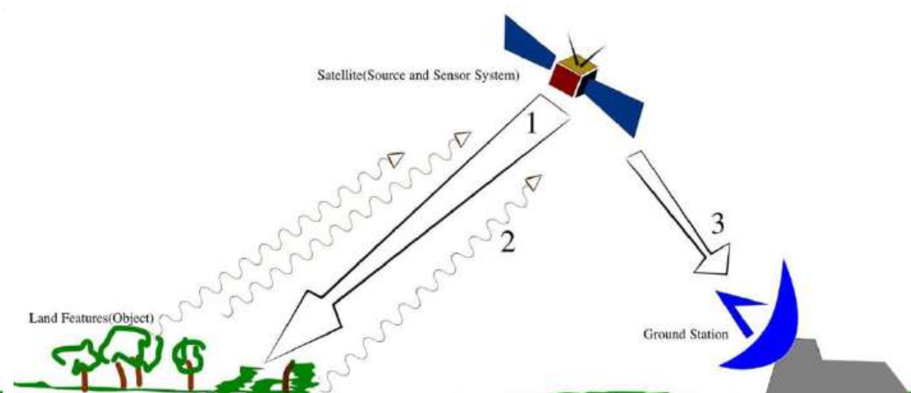
- The sensing units of human body like Eye, Ear and Nose are example of Remote Sensors as they don’t need to touch the monitored object to collect the information.

- Remote sensing is the process of detecting and monitoring the physical characteristics of an area by measuring its reflected and emitted radiation at a distance (typically from satellite or aircraft). Special cameras collect remotely sensed images, which help researchers "sense" things about the Earth.



Remote sensing using passive sensor system

Passive Remote Sensing



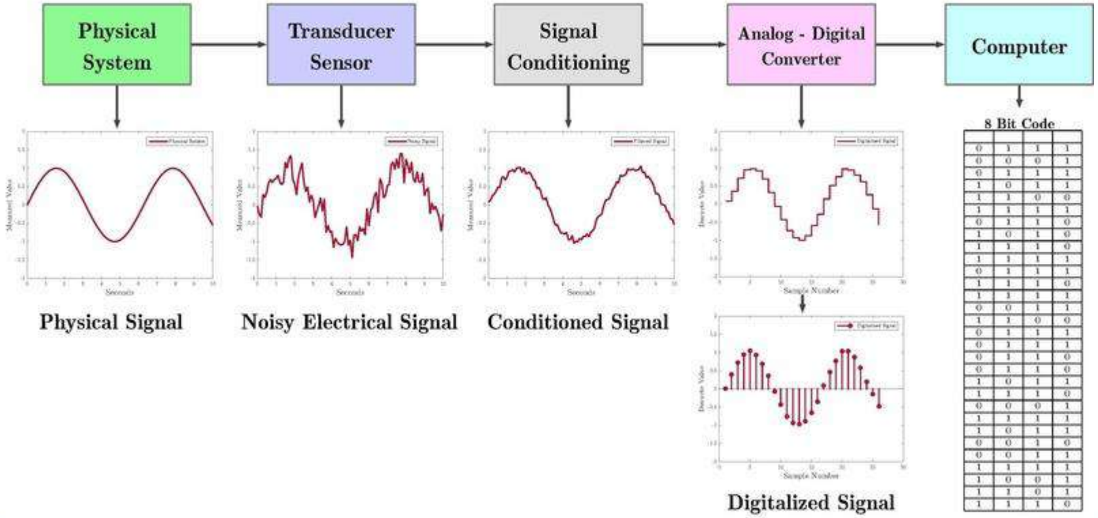
Remote sensing using active sensor system

Active Remote Sensing

Wireless Sensor Network

Data Acquisition

- A **transducer** is a device that converts energy from one form to another. Usually a **transducer** converts a signal in one form of energy to a signal in another.
- A sensor is a type of Transducer that converts different forms of energy into electrical energy and pass it to the controller and system for processing.
- The Sensing process is also known as Data Acquisition. **Data acquisition** is the process of sampling signals that measure real world physical conditions and converting the resulting samples into digital numeric values that can be manipulated by a computer.
- The components of data acquisition systems include:
  - Sensors, to convert physical parameters to electrical signals.
  - Signal conditioning circuitry, to convert sensor signals into a form that can be converted to digital values.
  - Analog-to-digital converters, to convert conditioned sensor signals to digital values.



Data Actuation

- An actuator is a device that activates process control equipment by using pneumatic, hydraulic, or electrical power. For example, a valve actuator opens and closes a valve to control fluid rate.

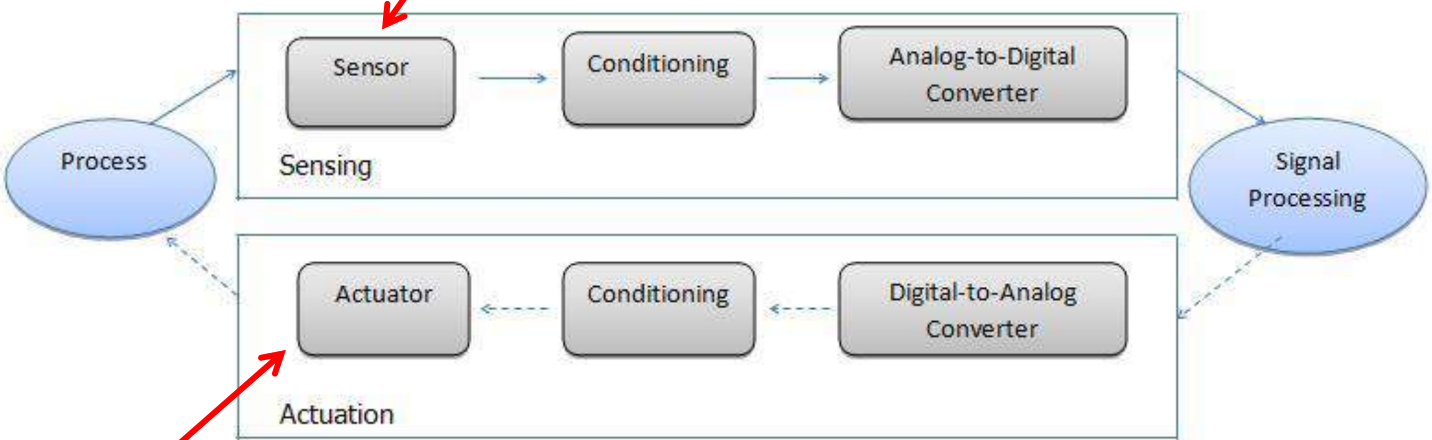
- An **actuator** is a component of a machine that is responsible for moving and controlling a mechanism or system, for example by opening a valve. In simple terms, it is a "mover".

- An actuator requires a control signal and a source of energy. The control signal is relatively low energy and may be electric voltage or current, pneumatic, or hydraulic fluid pressure, or even human power. Its main energy source may be an electric current, hydraulic pressure, or pneumatic pressure. When it receives a control signal, an actuator responds by converting the source's energy into mechanical motion. In the *electric*, *hydraulic*, and *pneumatic* sense, it is a form of automation or automatic control.

- An actuator is a mechanism by which a control system acts upon to perform an operation or task. The control system can be simple (a fixed mechanical or electronic system), software-based (e.g. a printer driver, robot control system), a human, or any other input.

Wireless Sensor Network

Data Acquisition and Actuation

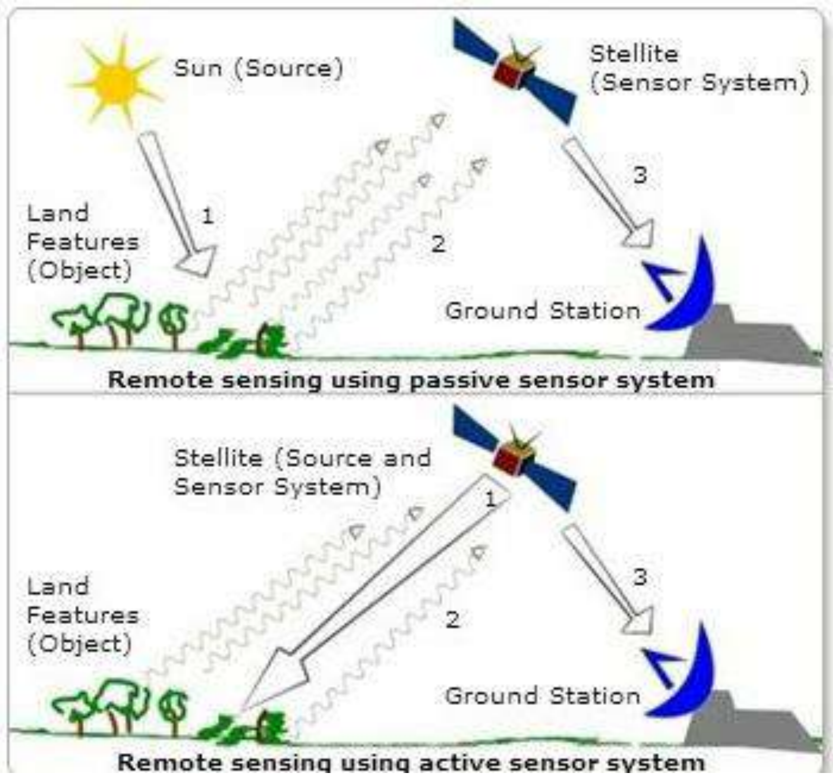


Wireless Sensor Network

Classification of Sensors

- **First Classification: Based on external power supply**
    1. Active Sensor: Active Sensors are those which require an external excitation signal or a power signal. e.g. Solar Cell, Piezoelectric Material and Thermocouple, etc.

    2. Passive Sensor: Passive Sensors, on the other hand, do not require any external power signal and directly generates output response. e. g. Photodiode, Strain Gauge, Thermistor etc.



Remote sensing using passive sensor system

Remote sensing using active sensor system

- **Passive sensors** detect natural energy (radiation) that is emitted or reflected by the object or scene being observed. Reflected sunlight is the most common source of radiation measured by passive sensors.

- **Active sensors**, on the other hand, provide their own source of energy to illuminate the objects they observe. An active sensor emits radiation in the direction of the target to be investigated. The sensor then detects and measures the radiation that is reflected or backscattered from the target.

Wireless Sensor Network

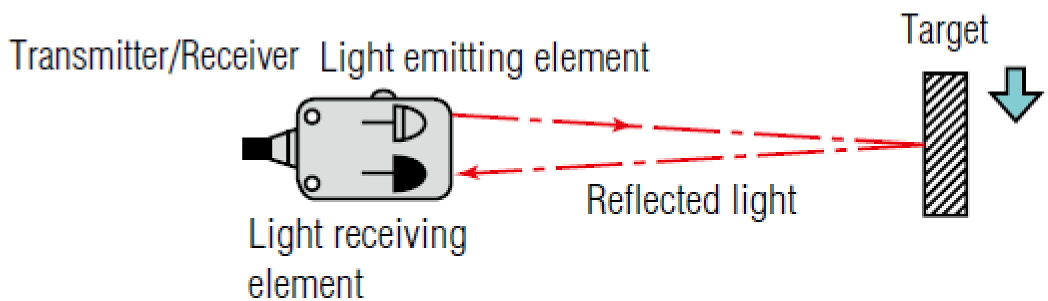
Classification of Sensors

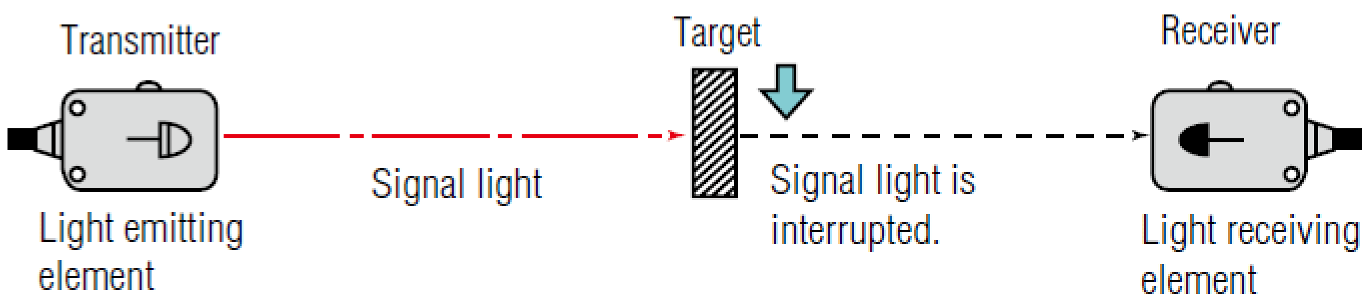
- **Second Classification:** Based on the physical property to be monitored such as Temperature, Flow, Pressure, Sound, Light etc.

Type	Examples
Temperature	Thermistors, thermocouples
Pressure	Pressure gauges, barometers, ionization gauges
Optical	Photodiodes, phototransistors, infrared sensors, CCD sensors
Acoustic	Piezoelectric resonators, microphones
Mechanical	Strain gauges, tactile sensors, capacitive diaphragms, piezoresistive cells
Motion, vibration	Accelerometers, mass air flow sensors
Position	GPS, ultrasound-based sensors, infrared-based sensors, inclinometers
Electromagnetic	Hall-effect sensors, magnetometers
Chemical	pH sensors, electrochemical sensors, infrared gas sensors
Humidity	Capacitive and resistive sensors, hygrometers, MEMS-based humidity sensors
Radiation	Ionization detectors, Geiger-Mueller counters

Classification of Sensors

- **Third Classification:** The next classification is based on conversion phenomenon i.e. the input and the output. Some of the common conversion phenomena are Photoelectric, Thermoelectric, Electrochemical, Electromagnetic, Thermooptic, etc.
  - A **photoelectric sensor** emits a light beam (visible or infrared) from its light-emitting element. A reflective-type photoelectric sensor is used to detect the light beam reflected from the target. A thrubeam type sensor is used to measure the change in light quantity caused by the target crossing the optical axis.



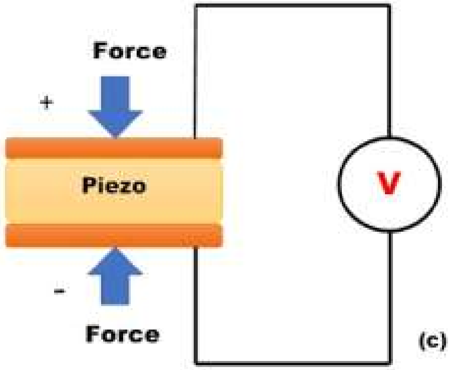


- **Piezoelectric effect:** Piezoelectric effect is a property of certain material to generate electrical energy by applying mechanical stress. It is an irreversible type of effect means inverse piezoelectric effect can deform the material when an electrical field is applied on it.

- The stress and strain may be applied from the different source like motion, noise, air flow and vibration. When mechanical stress is applied on a piezoelectric material, there is a shifting of positive charge and negative charge internally. It results an electrical field outside the material.

- There are two types of piezoelectric materials.
    1. Some Natural materials are Quartz, Topaz, Tourmaline, Cane Sugar, berlinite, Bone and Rochelle salt. Examples of artificial piezoelectric materials are lithium niobate, barium titanate and lead zirconate titanate etc.

    2. The electrical charge of piezoelectric material depends upon the various parameters and relationship.

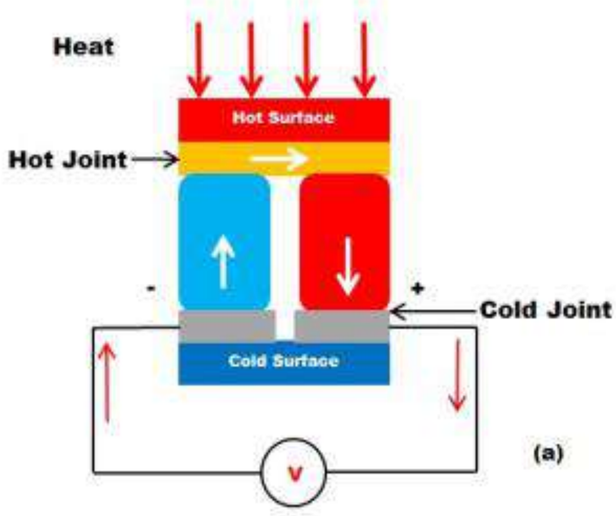


Wireless Sensor Network

- **Thermoelectric Effect/Method:** It is an irreversible property of Thermoelectric Materials for converting Thermal Energy into Electrical Energy and vice-versa. These devices basically have three thermoelectric effects (Peltier, Seebeck, Thomson).

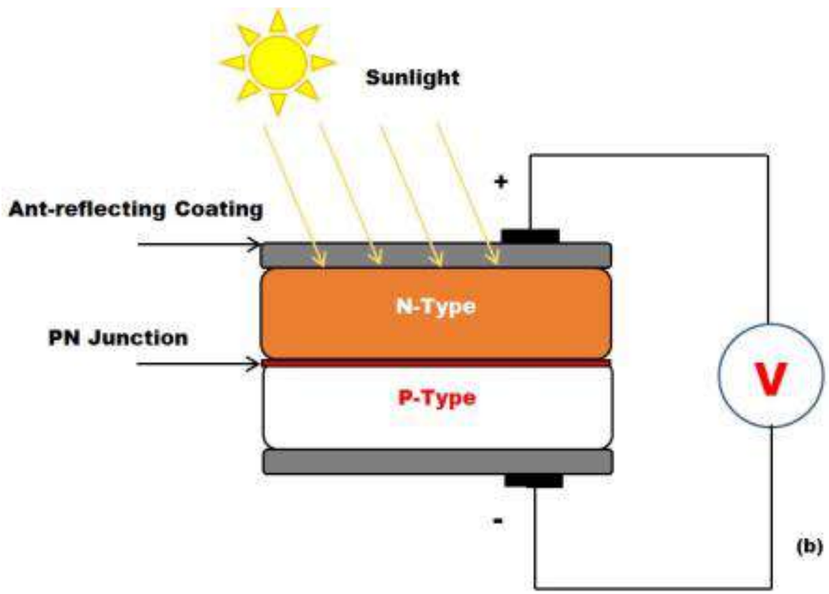
- Thermoelectric energy is produced by the Seebeck Effect due to the voltage generated by the difference of temperature at the joint of two different conductors.

- The thermoelectric energy harvester consists of TEG (Thermoelectric Generator) consists of a group of thermocouples connected in series to the main source of heat

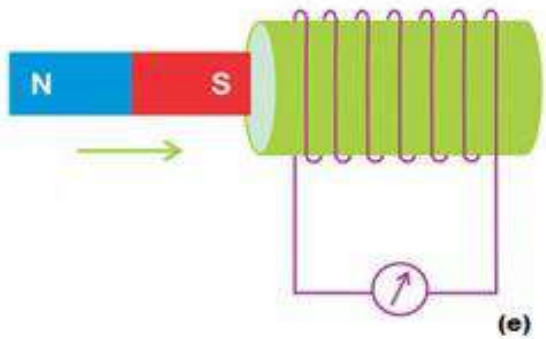


Wireless Sensor Network

- **Photovoltaic Effect/Method:** The Solar/Light energy is converted into electric energy using the Photovoltaic Cell termed as Photovoltaic Effect. When a PV Cell is exposed by Sunlight/Light, it releases electrons and generate the electric current internally without any external device or source. The intensity of Energy depends upon the size of cell, efficiency and the light intensity .



- **Electromagnetic Effect/Method:** Electromagnetic energy is harvested by electromagnetic induction effect. When a wire/coil moves inside the magnetic field, electric current is induced in the wire.
- Electricity can be generated by rotating a magnet inside a coil of wire. This induces a current in the wire. As the magnetic field is changing direction repeatedly (as it turns), the direction of the induced current also changes. Therefore, alternating Current (AC) is produced. This type of generator is called an alternator. Energy can also be produced by electromagnetic effect due to vibration and fluid flow source of energy.



Resistive Sensors

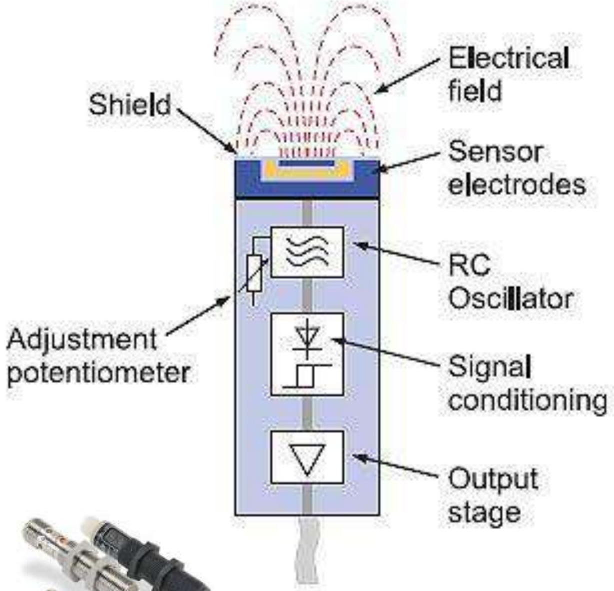
- The most commonly used type of transducer is variable resistance transducer. It is otherwise called as resistive sensors. It measures temperature, pressure, displacement, force, vibrations, etc. to understand the working principle, consider a conductor rod.
- Resistive sensors works on the principle that, the conductor length is directly proportional to resistance of the conductor and it is inversely related with area of the conductor. Therefore, L is denominated for conductor length, A for area of the conductor and R for resistance of conductor. ρ is the resistivity and it is constant for all the materials used for conductor construction.

$$R = \frac{\rho L}{A}$$

- The resistance of the transducer varies due to external environmental factors and physical properties of the conductor. AC or DC devices are used to measure the resistance change. This transducer acts as both primary and secondary transducer. As a primary transducer, it converts physical quantity into mechanical signal. As a secondary transducer, the obtained mechanical signal is converted into electrical signal.
- There are different sensors available in the market which consider different factors in detecting resistance. Examples of some basic resistive sensors:
  Thermistors, Flex sensors, Force sensing resistors, Light dependent resistors (photoresistors)

Capacitive Sensors

- It is used to measure motion, proximity, acceleration, pressure, electric fields, chemical composition and liquid depth.
- A typical capacitor consists of two conductive elements (sometimes called plates) separated by some kind of insulating material that can be one of many different types including ceramic, plastic, paper, or other materials. The capacitance is determined as:
 $C = \varepsilon A/d$, Where A is the Plate Area and d is the distance between two plates.
- Capacitive Proximity Sensors detect changes in the capacitance between the sensing object and the Sensor. As per the name, capacitive proximity sensors operate by noting a change in the capacitance read by the sensor.
- The amount of capacitance varies depending on the size and distance of the sensing object. An ordinary Capacitive Proximity Sensor is similar to a capacitor with two parallel plates, where the capacity of the two plates detected.
- One of the plates is the object being measured (with an imaginary ground), and the other is the Sensor's sensing surface. It detects the changes in the capacity generated between these two poles. The detection of the object depends on their dielectric constant, but they include resin and water in addition to metals.



Wireless Sensor Network

Sensors are classified based on

Based on measured value	Temperature, pressure, force/ torspeed/ position, light sensors...etc
Based on power source	Active: RTD, Thermistor Passive: Thermocouple, photodiode
Based on output signal	Analog Sesnor: thermocouple, RTD, strain gauge Digital sensors: DHT11,DHT11...
Application based	Agriculture, automotive, civil engineering and construction, domestic appliances, commerce, finance environment, meteorology, security, energy, telecommunication, health&medicine, marine, military and space
Specifications based	Accuracy, sensitivity, stability, response time, hysteresis, frequency response, input, resolution, linearity, hardness, cost, size, weight....
Physical law based	Photoelectric, magnetoelectric, thermoelectric, photoconductive, photomagnetic, thermomagnetic, thermooptic, electrochermical, magnetoresistive, photoelastic
Proprioceptive / exteroceptive	Proprioceptive : motor speed, wheel load, robot angles, and battery voltage Exteroceptive: distance, light intensity, and sound amplitude
Sensor material	Organic and inorganic sensors
Stimuli	Acoustic, electric, magnetic, optical, mechanical, radiation, thermal sensors

Wireless Sensor Network: History

- DARPA:
  - Distributed Sensor Nets Workshop (1978)
  - Distributed Sensor Networks (DSN) program (early 1980s)
  - Sensor Information Technology (SensIT) program
- UCLA and Rockwell Science Center
  - Wireless Integrated Network Sensors (WINS)
  - Low Power Wireless Integrated Microsensor (LWIM) (1996)
- UC-Berkeley
  - Smart Dust project (1999)
  - concept of "motes": extremely small sensor nodes
- Berkeley Wireless Research Center (BWRC)
  - PicoRadio project (2000)
- MIT
  - µAMPS (micro-Adaptive Multidomain Power-aware Sensors) (2005)

WSN Standards

The 802.11 standard is defined through several specifications of WLANs. It defines an over-the-air interface between a wireless client and a base station or between two wireless clients.

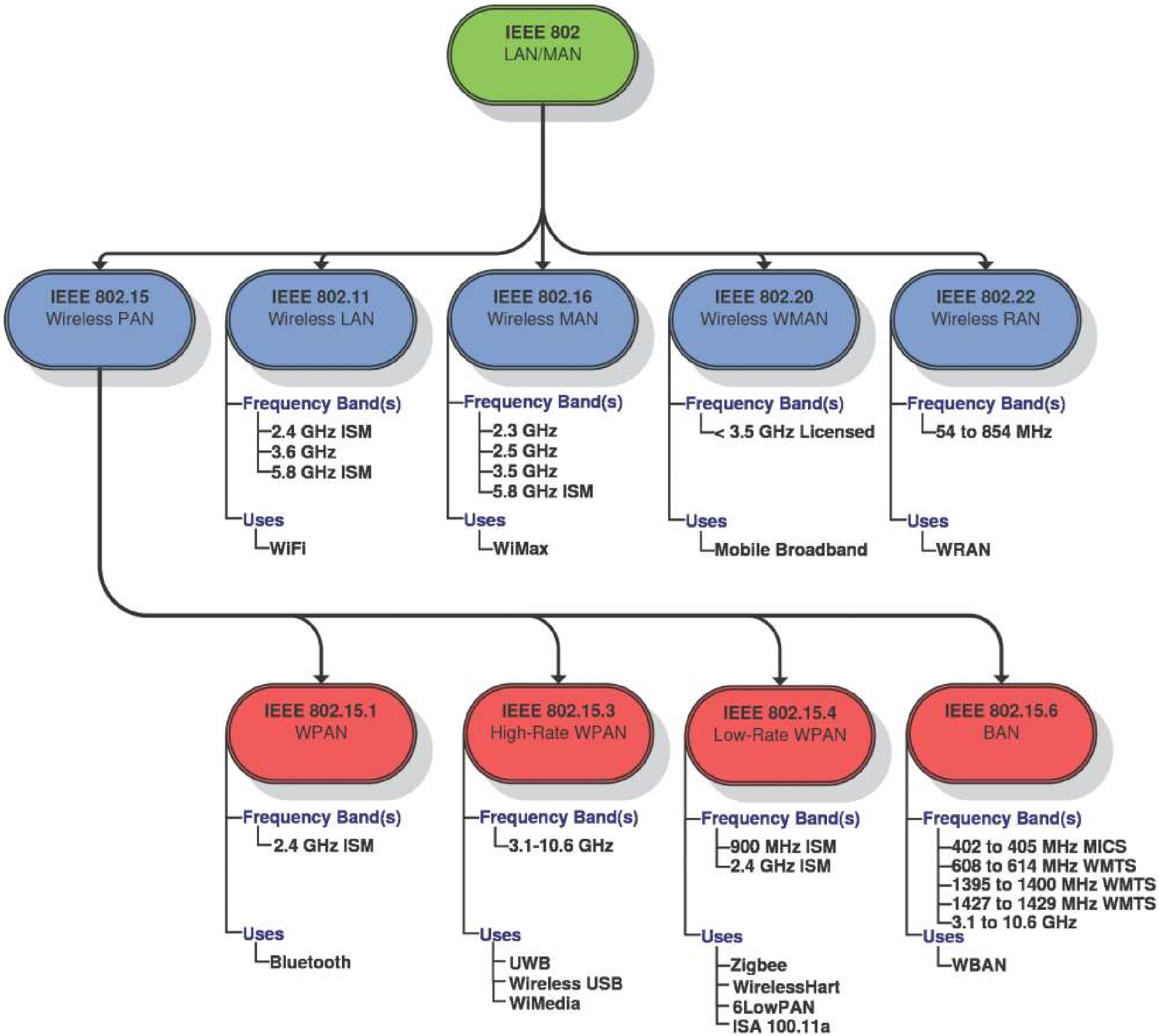
IEEE 802.11 Standards

802.11 protocol	Release	Freq. (GHz)	Band-width (MHz)	Data rate per stream (Mbit/s)	Allowable MIMO streams	Approximate indoor range (m)	Approximate indoor range (ft)	Approximate outdoor range (m)	Approximate outdoor range (ft)
-	Jun 1997	2.4	22	1, 2	N/A	20	66	100	330
a	Sep 1999	5	20	6, 9, 12, 18, 24, 36, 48, 54	N/A	35	115	120	390
		3.7				—	—	5,000	16,000
b	Sep 1999	2.4	22	1, 2, 5.5, 11	N/A	35	115	140	460
g	Jun 2003	2.4	20	6, 9, 12, 18, 24, 36, 48, 54	N/A	38	125	140	460
n	Oct 2009	2.4/5	20	7.2, 14.4, 21.7, 28.9, 43.3, 57.8, 65, 72.2	4	70	230	250	820
			40	15, 30, 45, 60, 90, 120, 135, 150		70	230	250	820
ac	Dec 2013	5	20	up to 87.6	8				
			40	up to 200					
			80	up to 433.3					
			160	up to 866.7					

IEEE802.11 was frequently used in early WSN and can still be used in current networks where the bandwidth demands are high (e.g. for Multimedia Sensors).

However, the high-energy overheads of IEEE802.11 is unsuitable for the low-power sensor networks. So, the variety of protocol have been developed for the need of low-power consumption and low data rates. e.g. 802.15.4 protocol has been designed specifically for short-range communication and low-power sensor networks.

WSN Standards



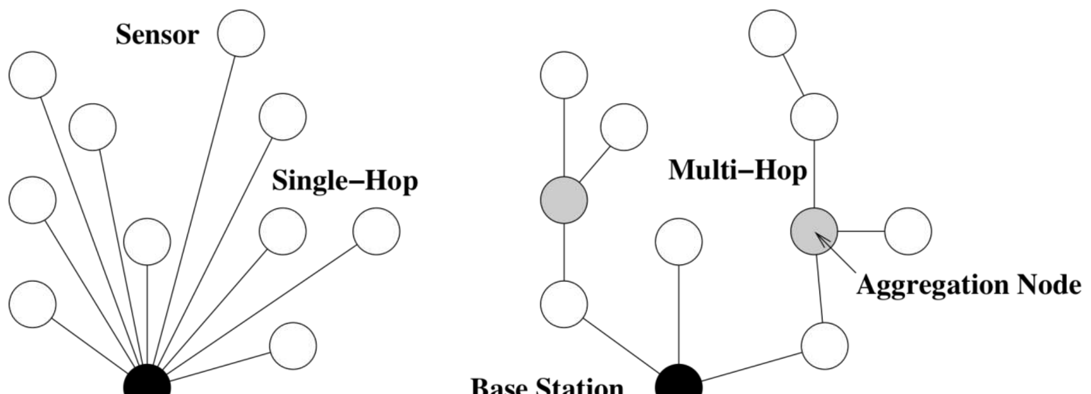
Wireless Sensor Network Communication

Star Topology (Single-hop Communication)

- Every sensor directly communicate to the base station (single hop).
- May require large transmit power
- May be infeasible for large geographical area

Mesh Topology (Multi-hop Communication)

- Sensor serves as relay (forwarder) to the others sensor nodes.
- May reduce power consumption and allows for larger coverage area.
- Introduces the problems of routing (the task of finding a multi-hop path from a sensor node to the base station).
- Elimination of redundant data or aggregation of data that may be smaller than original data.



Challenges in WSN: Energy

Energy

- Sensors typically powered through batteries
  - replace battery when depleted
  - recharge battery, e.g., using solar power
  - discard sensor node when battery depleted

- For batteries that cannot be recharged, sensor node should be able to operate during its entire **mission time** or until battery can be replaced

- Energy efficiency is affected by various aspects of sensor node/network design.

- Physical layer:
  - switching and leakage energy of CMOS-based processors.

$$E_{CPU} = E_{switch} + E_{leakage} = C_{total} * V_{dd}^{2} + V_{dd} * I_{leak} * \Delta t$$

Where

Ctotal	is the total capacitance switched by the computation
Vdd	is supply voltage
Ileak	is the leakage current
Δt	is the duration of computation

Challenges in WSN: Energy

Energy

- Medium access control layer:
  - contention-based strategies lead to energy-costly collisions
  - problem of idle listening

- Network layer:
  - responsible for finding energy-efficient routes

- Operating system:
  - small memory footprint and efficient task switching

- Security:
  - fast and simple algorithms for encryption, authentication, etc.

- Middleware:
  - in-network processing of sensor data can eliminate redundant data or aggregate sensor readings

Challenges in WSN: Self Management

Self Management

- Ad-hoc deployment
  - many sensor networks are deployed “without design”
    - sensors dropped from airplanes (battlefield assessment)
    - sensors placed wherever currently needed (tracking patients in disaster zone)
    - moving sensors (robot teams exploring unknown terrain)
  - sensor node must have some or all of the following abilities
    - determine its location
    - determine identity of neighboring nodes
    - configure node parameters
    - discover route(s) to base station
    - initiate sensing responsibility
- Unattended operation
  - once deployed, WSN must operate without human intervention
  - device adapts to changes in topology, density, and traffic load
  - device adapts in response to failures

Challenges in WSN: Self Management

- A Self-managing device
  - Monitor its surroundings,
  - Adapt to changes in the environment,
  - Cooperate with neighboring devices to form topologies or agree on sensing, processing, and communication strategies.

- Various Forms of Self-management
  - **self-organization** is the ability to adapt configuration parameters based on system and environmental state. (e.g. increase of transmission power in order to connect with the neighbor node).
  - **self-optimization** is the ability to monitor and optimize the use of the limited system resources.
  - **self-protection** is the ability recognize and protect from intrusions and attacks.
  - **self-healing** is the ability to discover, identify, and react to network disruptions.

Challenges in WSN: Wireless Networking

Wireless communication faces a variety of challenges

- Attenuation:
  - limits radio range, i.e. a RF signal fades (i.e. decreases its power) while it propagates through a medium and it passes through obstacles. It can be represented by **Inverse-Square Law**

$$P_r \propto \frac{P_t}{d^2}$$

Where,

Received Power	: Pr
Transmitted Power	: Pt
Distance from the source of signal	: d

  - As a Consequences, an increase in distance require more transmission power. Therefore, distance can be divided into shorter distance (i.e. Multi-hop communication can be adopted).

Challenges in WSN: Wireless Networking

- Multi-hop communication:
  - Nodes in a network cooperate with each other to identify efficient routes and serves as relay.
  - increased latency.
  - increased failure/error probability.
  - complicated by use of **duty cycles**,
    - ✓ Power Conservation Policies are used
    - ✓ Radios are switched off when they are not in use.
    - ✓ Neither Sensor Nodes receive signals from its neighbors nor serve as relay for other nodes.

1. Wake-up and demand Duty Cycling.
   - ❑ Node can wakeup whenever needed
   - ❑ Devices use tow radios, a lower power radio is used to receive wakeup calls, a high power radio is activated in response of wakeup call.
2. Adaptive Duty Cycling
   - ❑ Not all the nodes are allowed to sleep at the same time.
   - ❑ A subset of nodes remain active to form a network backbone.

Challenges in WSN: Decentralization

- **Centralized** management (e.g., at the base station) of the network often not feasible to due large scale of network and energy constraints.

- Therefore, **decentralized** (or distributed) solutions often preferred, though they may perform worse than their centralized counterparts, but they may be more energy-efficient than centralized solutions.

- Example: routing
- Centralized:
  - BS collects information from all sensor nodes
  - BS establishes “optimal” routes (e.g., in terms of energy)
  - BS informs all sensor nodes of routes
  - can be expensive, especially when the topology changes frequently

- Decentralized:
  - each sensors makes routing decisions based on limited local information
  - routes may be nonoptimal, but route establishment/management can be much cheaper

Challenges in WSN: Design Constraints

- Many hardware and software limitations affect the overall system design.

- Examples include:
  - Low processing speeds (to save energy)
  - Low storage capacities (to allow for small form factor and to save energy)
  - Lack of I/O components such as GPS receivers (reduce cost, size, energy)
  - Lack of software features such as multi-threading (reduce software complexity)

Challenges in WSN: Security

- Sensor networks often monitor critical infrastructure or carry sensitive information, making them desirable targets for attacks.

- Attacks may be facilitated by:
  - remote and unattended operation
  - wireless communication
  - lack of advanced security features due to cost, form factor, or energy

- Conventional security techniques often not feasible due to their computational, communication, and storage requirements.

- As a consequence, sensor networks require new solutions for intrusion detection, encryption, key establishment and distribution, node authentication, and secrecy.

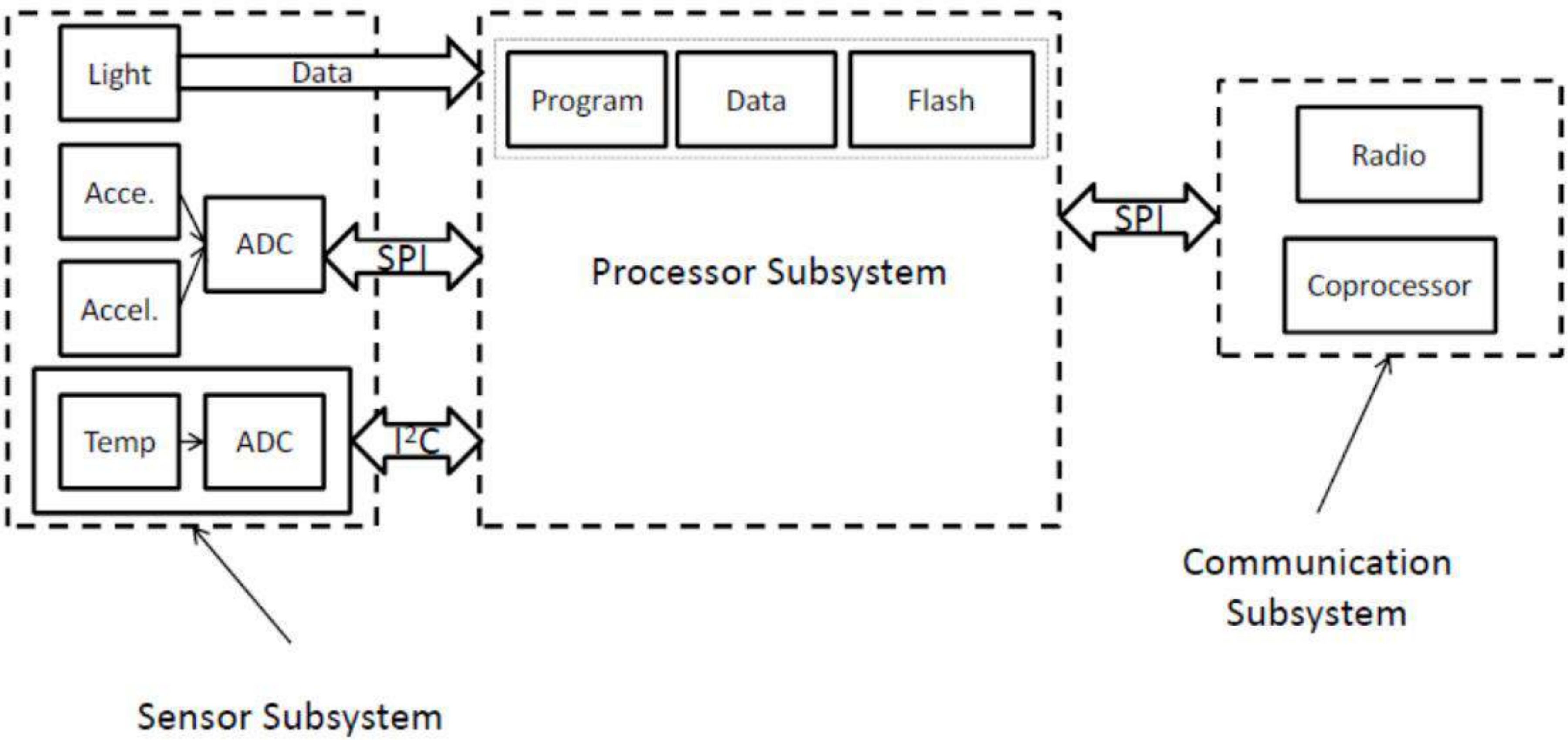
Comparison: Traditional Networks Vs Wireless Sensor Networks

Traditional Networks	Wireless Sensor Networks
General-purpose design; serving many applications	Single-purpose design; serving one specific application
Typical primary design concerns are network performance and latencies; energy is not a primary concern	Energy is the main constraint in the design of all node and network components
Networks are designed and engineered according to plans	Deployment, network structure, and resource use are often ad-hoc (without planning)
Devices and networks operate in controlled and mild environments	Sensor networks often operate in environments with harsh conditions
Maintenance and repair are common and networks are typically easy to access	Physical access to sensor nodes is often difficult or even impossible
Component failure is addressed through maintenance and repair	Component failure is expected and addressed in the design of the network
Obtaining global network knowledge is typically feasible and centralized management is possible	Most decisions are made localized without the support of a central manager

Node Architecture

- Wireless sensor nodes are the essential building blocks in a wireless sensor network
  - Sensing, Processing, and Communication
  - Stores and executes the communication protocols as well as data processing algorithms

- The node consists of sensing, processing, communication, and power subsystems.



Node Architecture

- Sensing units are usually composed of two subunits:
  - Sensors and
  - Analogue to Digital Converters (ADCs).

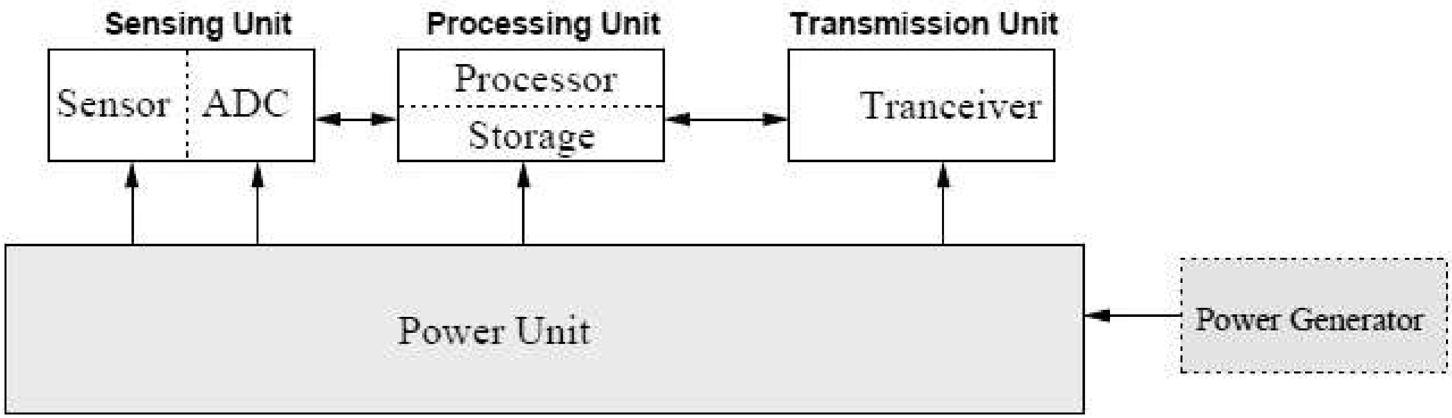
    The analogue signals produced by the sensors are converted to digital signals by the ADC, and then fed into the processing unit.

- The processing unit is generally associated with a small storage unit and it can manage the procedures that make the sensor node collaborate with the other nodes to carry out the assigned sensing tasks. The processor subsystem is the central element of the node and the choice of a processor determines the tradeoff between flexibility and efficiency – in terms of both energy and performance. There are several processors as options:
  - Microcontroller
  - Digital Signal Processor (DSP)
  - Application-specific Integrated Circuit (ASIC)
  - Field Programmable Gate Array (FPGA)

- A transceiver unit connects the node to the network.
  - Serial Peripheral Interface (SPI)
  - Inter-Integrated Circuit (I^2C)

Node Architecture

- The power subsystem provides DC power to all the other subsystems to bias their active components such as crystal oscillators, amplifiers, registers, and counters. Moreover, it provides DC–DC converters so that each subsystem can obtain the right amount of bias voltage.



Node Architecture: The Sensing Subsystem

- The Sensing Subsystem integrates the sensors as per the requirement of the application. The following table shows the details of sensors used in applications.

Sensor	Application Area	Sensed Event	Explanation
Accelerometer	AVM	2D and 3D acceleration of movements of people and objects	Volcano activities
	SHM		Stiffness of a structure
	Health care		Stiffness of bones, limbs, joints; Motor fluctuation in Parkinson's disease
	Transportation		Irregularities in rail, axle box or wheels of a train system
	SCM		Defect of fragile objects during transportation
Acoustic emission sensor	SHM	Elastic waves generated by the energy released during crack propagation	Measures micro-structural changes or displacements
Acoustic sensor	Transportation & Pipelines	Acoustic pressure vibration	Vehicle detection; Measure structural irregularities; Gas contamination
Capacitance sensor	PA	Solute concentration	Measure the water content of a soil

Node Architecture: The Sensing Subsystem

Sensor	Application Area	Sensed Event	Explanation
ECG	Health care	Heart rate	
EEG		Brain electrical activity	
EMG		Muscle activity	
Electrical sensors	PA	Electrical capacitance or inductance affected by the composition of tested soil	Measure of nutrient contents and distribution
Gyroscope	Health care	Angular velocity	Detection of gait phases
Humidity sensor	PA & HM	Relative and absolute humidity	
Infrasonic sensor	AVM	Concussive acoustic waves – earth quake or volcanic eruption	
Magnetic sensor	Transportation	Presence, intensity, direction, rotation and variation of a magnetic field	Presence, speed and density of a vehicle on a street; congestion
Oximeter	Health care	Blood oxygenation of patient's hemoglobin	Cardiovascular exertion and trending of exertion relative to activity
pH sensor	Pipeline (water)	Concentration of hydrogen ions	Indicates the acid and alkaline content of a water measure of cleanliness

Node Architecture: The Sensing Subsystem

Sensor	Application Area	Sensed Event	Explanation
Photo acoustic spectroscopy	Pipeline	Gas sensing	Detects gas leak in a pipeline
Piezoelectric cylinder	Pipeline	Gas velocity	A leak produces a high frequency noise that produces a high frequency noise that produces vibration
Soil moisture sensor	PA	Soil moisture	Fertilizer and water management
Temperature sensor	PA & HM	Pressure exerted on a fluid	
Passive infrared sensor	Health care & HM	Infrared radiation from objects	Motion detection
Seismic sensor	AVM	Measure primary and secondary seismic waves (Body wave, ambient vibration)	Detection of earth quake
Oxygen sensor	Health care	Amount and proportion of oxygen in the blood	
Blood flow sensor	Health care	The Doppler shift of a reflected ultrasonic wave in the blood	

Node Architecture: The Sensing Subsystem

ADC converts the output of a sensor - which is a continuous, analog signal - into a digital signal. It requires two steps:

1. The analog signal has to be quantized
   - allowable discrete values is influenced :
     a) by the frequency and magnitude of the signal
     b) by the available processing and storage resources

2. The sampling frequency
   - Nyquist rate does not suffice because of noise and transmission error
   - Resolution of ADC - an expression of the number of bits that can be used to encode the digital output

$$Q = \frac{E_{pp}}{2^M}$$

   - where Q is the resolution in volts per step (volts per output code); *Epp* is the peak-to-peak analog voltage; *M* is the ADC's resolution in bits

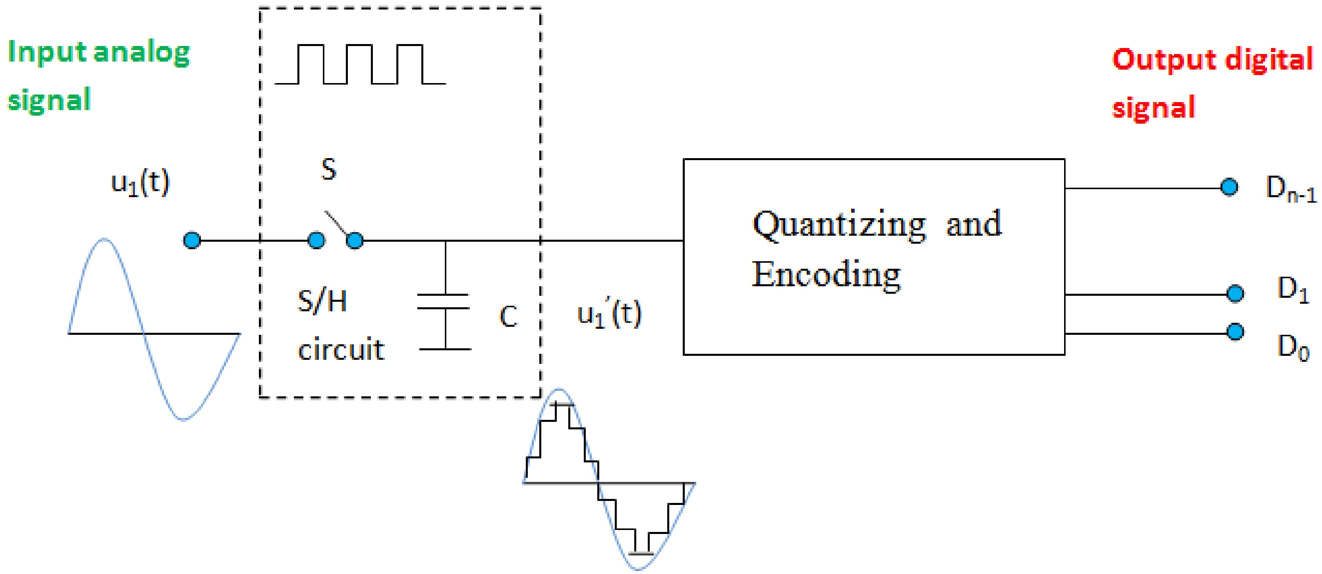
Node Architecture: The Sensing Subsystem

There are mainly two steps involves in the process of conversion. They are

- Sampling and Holding
- Quantizing and Encoding

Sampling and Holding

In the process of Sample and hold (S/H), the continuous signal will gets sampled and freeze (hold) the value at a steady level for a particular least period of time. It is done to remove variations in input signal which can alter the conversion process and thereby increases the accuracy. The minimum sampling rate has to be two times the maximum data frequency of the input signal.



Node Architecture: The Sensing Subsystem

Quantizing and Encoding

For understanding quantizing, we can first go through the term Resolution used in ADC. It is the smallest variation in analog signal that will result in a variation in the digital output. This actually represents the quantization error.

$$Resolution, \triangle V = \frac{V_r}{2^N}$$

V → Reference voltage range
2^N → Number of states
N → Number of bits in digital output

Quantizing: It is the process in which the reference signal is partitioned into several discrete quanta and then the input signal is matched with the correct quantum.

Encoding: Here; for each quantum, a unique digital code will be assigned and after that the input signal is allocated with this digital code. The process of quantizing and encoding is demonstrated in the table below.

Analog signal			Digital o/p
7.5	7 →	7Δ=7V →	111
6.5	6 →	6Δ=6V →	110
5.5	5 →	5Δ=5V →	101
4.5	4 →	4Δ=4V →	100
3.5	3 →	3Δ=3V →	011
2.5	2 →	2Δ=2V →	010
1.5	1 →	1Δ=1V →	001
0.5	0 →	0Δ=0V →	000

From the above table we can observe that only one digital value is used to represent the whole range of voltage in an interval. Thus, an error will occur and it is called quantization error. This is the noise introduced by the process of quantization. Here the maximum quantization error is

$$\pm \frac{1}{2}\triangle V = \pm 0.5V$$

Node Architecture: The Processor Subsystem

- The processor subsystem brings together
  - all the other subsystems and some additional peripherals.
  - Its main purpose is to process (execute) instructions pertaining to sensing, communication, and self-organization.

- It consists of
  - a processor chip,
  - a nonvolatile memory (usually an internal flash memory) for storing program instructions, an active memory for temporarily storing the sensed data, and
  - an internal clock, among other things.

- A wide range of off-the-shelf processors are available for building a wireless sensor node, one has to make a careful choice, as it affects the cost, flexibility, performance, and energy consumption of the node.
  - Microcontroller
  - Digital Signal Processor (DSP)
  - Application-specific Integrated Circuit (ASIC)
  - Field Programmable Gate Array (FPGA)

Node Architecture: The Processor Subsystem

- If the sensing task is well defined from the outset and does not change over time, a designer may choose either a **field programmable gate array or a digital signal processor**. These processors are very efficient in terms of their energy consumption; and for most simple sensing tasks, they are quite adequate.

- The sensing goal changes or a modification may be required. Moreover, the software that runs on the wireless sensor node may require occasional updates or remote debugging. Such tasks require a considerable amount of computation and processing space at runtime. In which case, **special-purpose, energy-efficient processors** are not suitable.

- WSNs are emerging technologies; and the research community is still active with research for developing energy-efficient communication protocols and signal-processing algorithms. As this requires dynamic code installation and update, the **microcontroller** is the best option.

Node Architecture: The Processor Subsystem

- A major concern in resource-constrained processors is the efficient execution of algorithms, since this requires the transferring of information from and to memory.

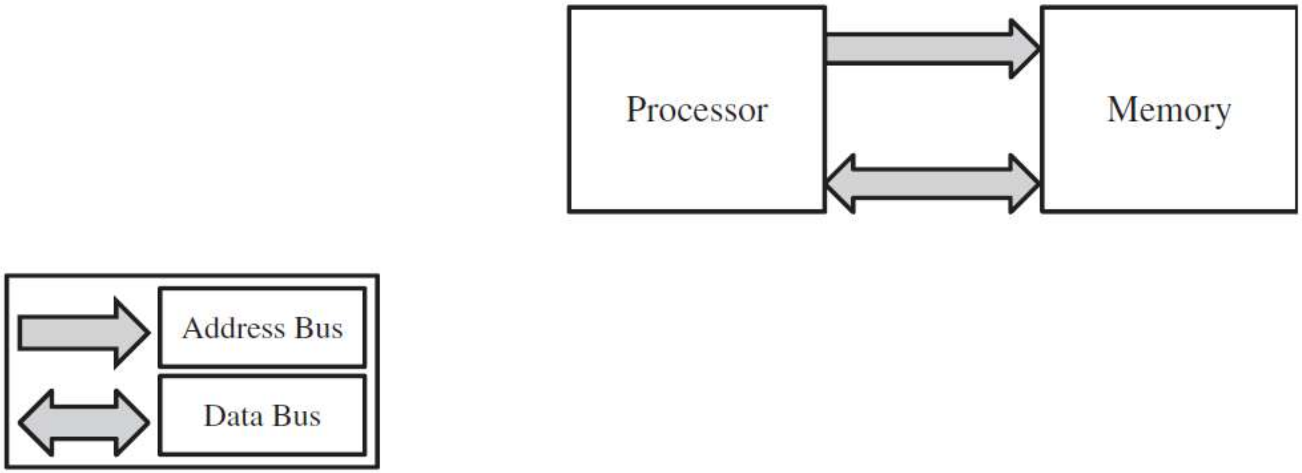
- This includes program instructions and the data to be processed or manipulated. For example, in WSNs, the data stem from the physical sensors and the program instructions relate to communication, self-organization, data compression, and aggregation algorithms.

- The processor subsystem can be designed by employing one of the three basic computer architectures!
  - *Von Neumann architecture*
  - *Harvard architecture*
  - *Super-Harvard (SHARC) architecture*

Node Architecture: The Processor Subsystem

Von Neumann architecture

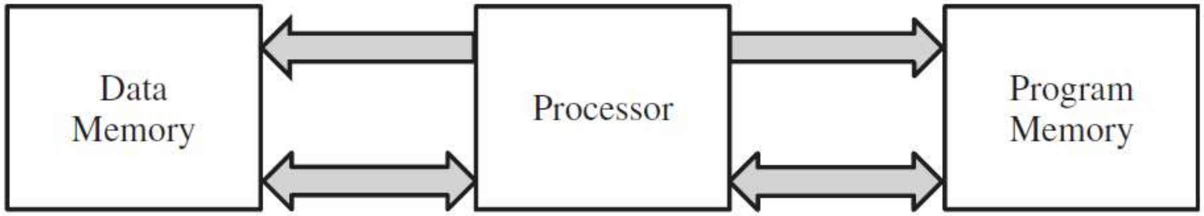
- A single memory space that is used by program instructions and data.
- A single bus to transfer data between the processor and the memory.
- A relatively slow processing speed because each data transfer requires a separate clock.

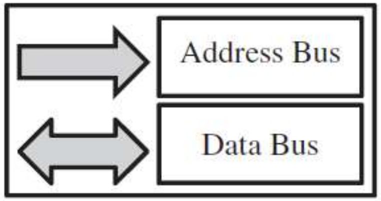


Node Architecture: The Processor Subsystem

Harvard architecture

- provides *separate memory spaces* - storing program instructions and data
- each memory space is interfaced with the processor with a separate data bus
- program instructions and data can be accessed *at the same time*
- a special *single instruction, multiple data (SIMD)* operation, a special arithmetic operation and a bit reverse
- supports multi-tasking operating systems; but does not provide virtual memory protection



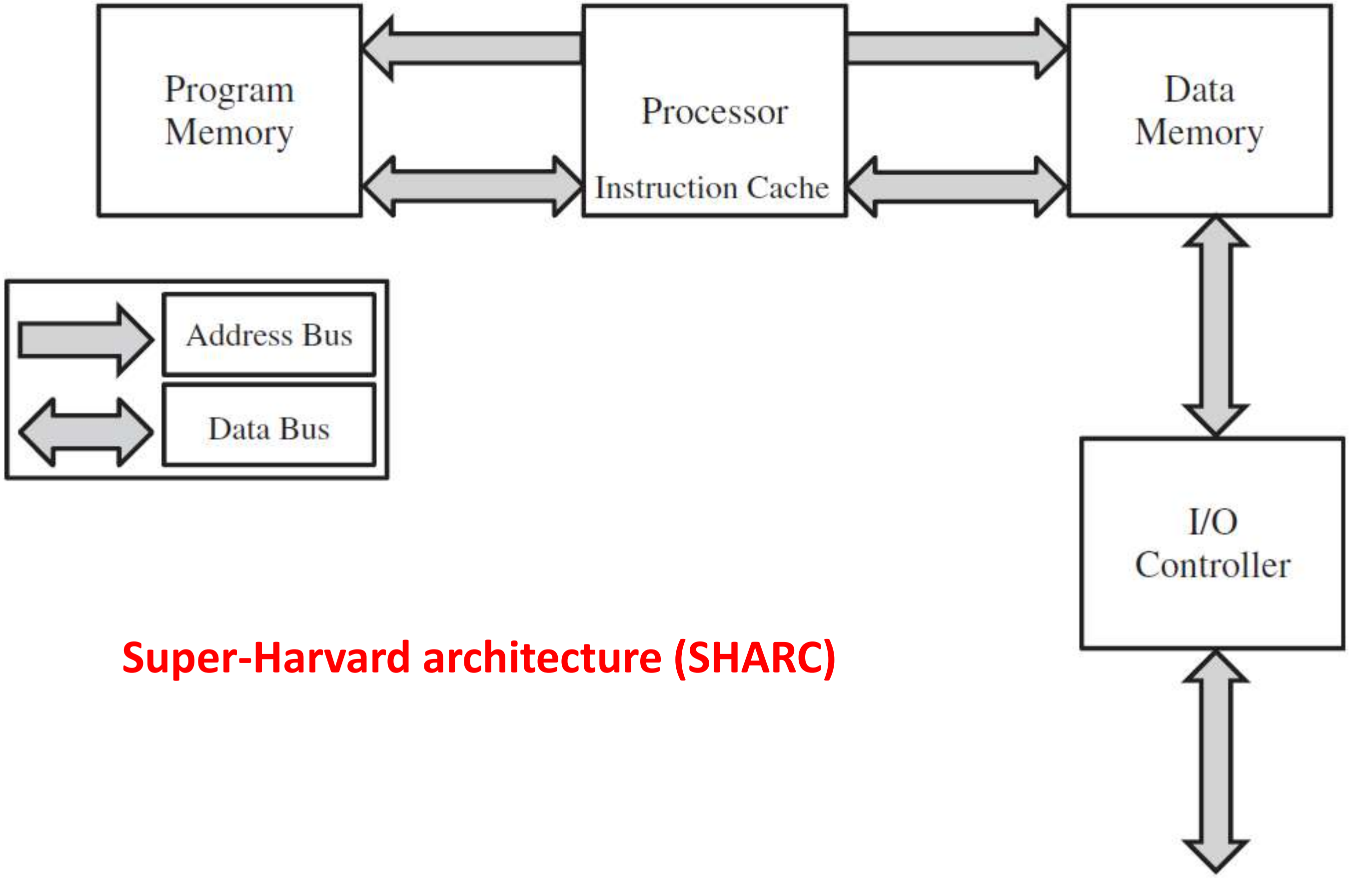


Node Architecture: The Processor Subsystem

Super-Harvard architecture (SHARC)

- An extension of the Harvard architecture
- *Adds* two components to the Harvard architecture
  - Internal instruction cache - temporarily store frequently used instructions - enhances performance
  - An underutilized program memory can be used as a temporary relocation place for data
  - Why Direct Memory Access (DMA) ?
    - costly CPU cycles can be invested in a different task
    - program memory bus and data memory bus accessible from outside the chip

Node Architecture: The Processor Subsystem



Super-Harvard architecture (SHARC)

Node Architecture: The Processor Subsystem

Microcontroller

- It is a computer on a single integrated circuit, consisting of a comparatively simple central processing unit and additional components such as high-speed uses, a memory unit, a watchdog timer, and an external clock.

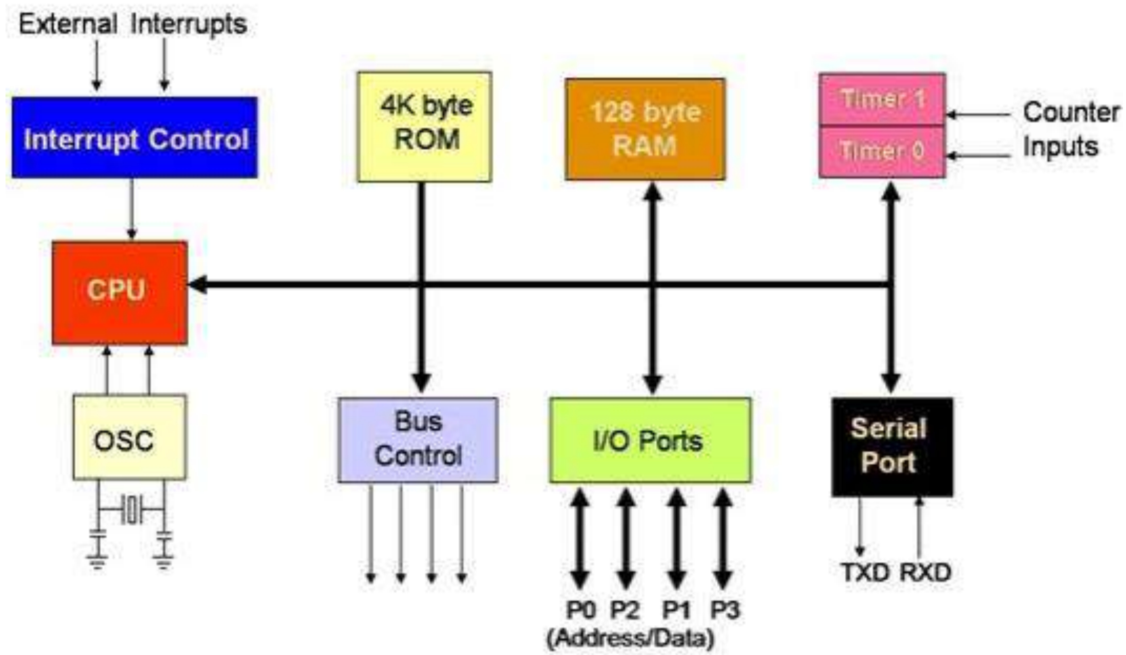
- Microcontrollers are integrated in many products and embedded devices. Today such simple systems as elevators, ventilators, office machines, household appliances, power tools, and toys ubiquitously employ microcontrollers.

Structure of microcontroller

integrates the following components:

- CPU core
- Volatile memory (RAM) for data storage
- ROM, EPROM, EEPROM, or Flash memory
- Parallel I/O interfaces
- Discrete input and output bits
- Clock generator
- One or more internal analog-to-digital converters
- Serial communications interfaces

Node Architecture: The Processor Subsystem



Advantages:

- suitable for building computationally less intensive, standalone applications, because of its *compact construction*, *small size*, *low-power consumption*, and *low cost*
- *high speed* of the programming and eases debugging, because of the use of higher-level programming languages

Disadvantages:

- *not* as *powerful* and as *efficient* as some custom-made processors (such as DSPs and FPGAs)
- some applications (simple sensing tasks but large scale deployments) may prefer to use architecturally simple but energy- and cost-efficient processors

Node Architecture: The Processor Subsystem

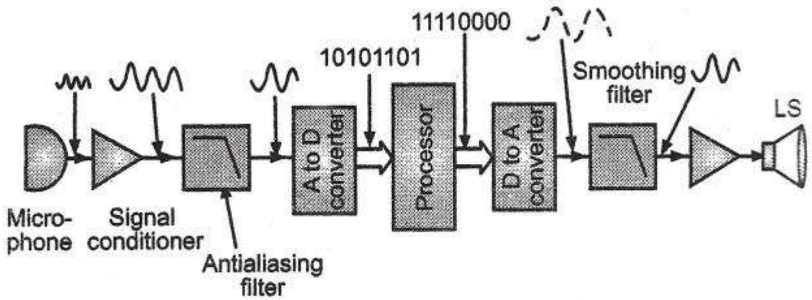
Digital Signal Processor

The main function:

- process discrete signals with digital filters
- filters minimize the effect of noise on a signal or enhance or modify the spectral characteristics of a signal
- while analog signal processing requires complex hardware components, digital signal processors (DSP) requires simple adders, multipliers, and delay circuits
- DSPs are highly efficient
- most DSPs are designed with the Harvard Architecture

Advantages:

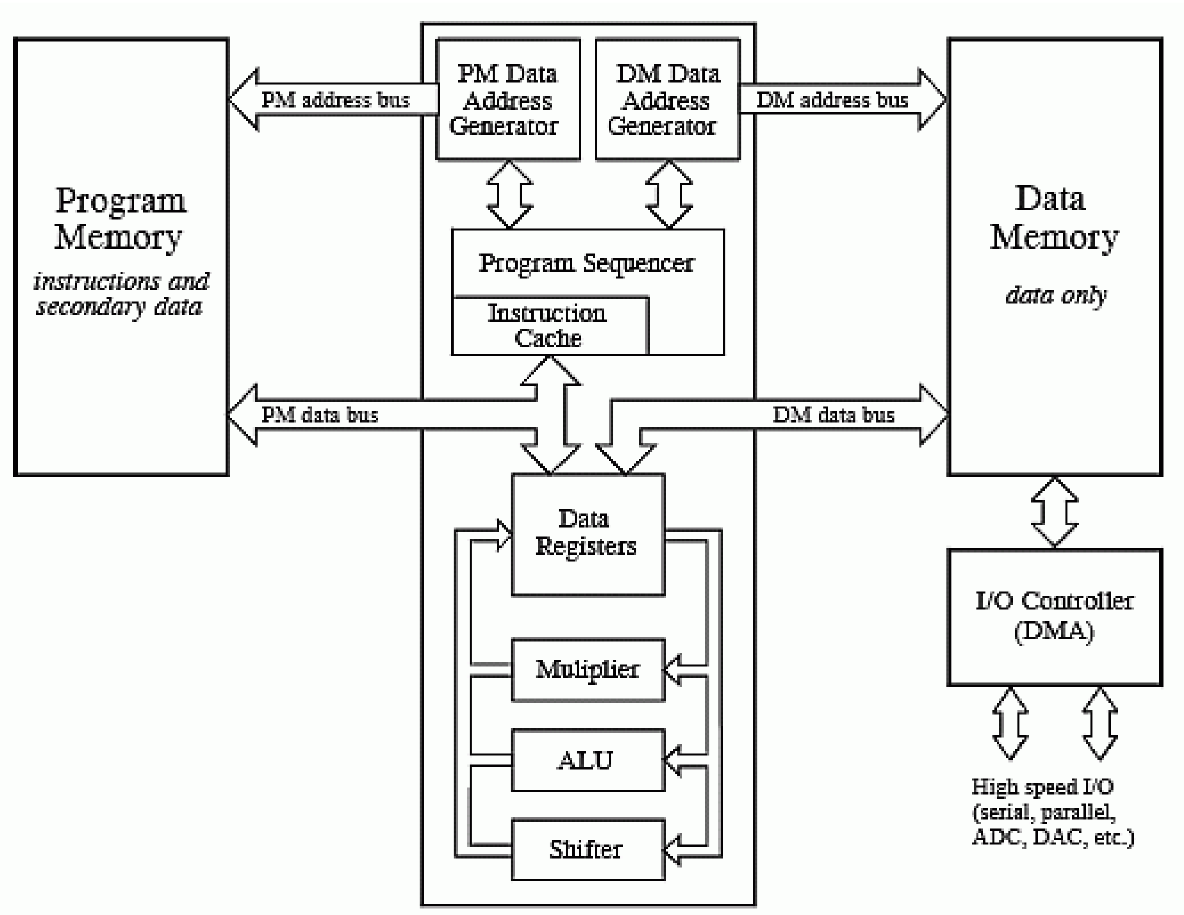
- *powerful* and *complex* digital filters can be realized with *commonplace* DSPs
- *useful for applications* that require the deployment of nodes in harsh physical settings (where the signal transmission suffers corruption due to noise and interference and, hence, requires aggressive signal processing)



Node Architecture: The Processor Subsystem

Disadvantage:

- some tasks require *protocols* (and not numerical operations) that *require* periodical *upgrades* or *modifications* (i.e., the networks should support flexibility in network reprogramming)



Node Architecture: The Processor Subsystem

Application-specific Integrated Circuit

- ASIC is an *IC* that can be customized for a specific application
- Two types of design approaches: *full-customized* and *half-customized*
- Full-customized IC:
  - Some logic cells, circuits, or layout are custom made in order to optimize cell performance
  - Includes features which are not defined by the standard cell library
  - Expensive and long design time
- Half-customized ASICs are built with logic cells that are available in the standard library
- In both cases, the final logic structure is configured by the end user - an ASIC is a *cost efficient solution, flexible,* and *reusable*

Node Architecture: The Processor Subsystem

Advantages:

- Relatively *simple design*; can be optimized to *meet a specific customer demand*
- *Multiple* microprocessor *cores* and *embedded software* can be designed in a *single* cell

Disadvantage:

- *High* development *costs* and *lack of re-configurability*

Application:

- ASICs are not meant to replace microcontrollers or DSPs but to complement them
- Handle rudimentary and low-level tasks
- To decouple these tasks from the main processing subsystem

Node Architecture: The Processor Subsystem

Field Programmable Gate Array (FPGA)

The distinction between ASICs and FPGAs is not always clear

- FPGAs are *more complex* in design and *more flexible* to program
- FPGAs are programmed electrically, by modifying a packaged part
- Programming is done with the support of circuit diagrams and hardware description languages, such as VHDL and Verilog

Typical features of a FPGA are summarized as follows:

- In a FPGA, none of the mask layers is customized;
- A FPGA includes some programmable logic components, or *logic blocks* – these are: a 4-input lookup table (LUT), a flip-flop, and an output block;
- There is a well-defined and formal method for programming the basic logic cells and the interconnect;
- There is a matrix of programmable interconnects surrounding the basic logic cells producing a configuration instance; and
- There are programmable I/O cells that surround the core.

Node Architecture: The Processor Subsystem

Advantages

- *Higher bandwidth* compared to DSPs
- *Flexible* in their application
- Support *parallel processing*
- Work with *floating point representation*
- Greater *flexibility of control*

Disadvantages

- *Complex*
- The design and realization process is *costly*

Node Architecture: The Processor Subsystem

Comparison ASIC, FGPA, Microcontroller & DSP

	ASIC	FGPA	UP/UC	DSP
Flexibility	None	Limited	High	High
Design Time	Long	Medium	Short	Short
Power Consumption	Low	Low-Medium	Medium-High	Low-Medium
Performance	High	High	Low-Medium	Medium-High
Development Cost	High	Medium	Low	Low
Production Cost	Low	Low-Medium	Medium-High	Low-Medium

- Working with a ***micro-controller*** is *preferred* if the design goal is to achieve *flexibility*
- Working with the other mentioned options is preferred if power consumption and computational efficiency is desired.
- ***DSPs*** are expensive, large in size and less flexible; they are *best for signal processing*, with specific algorithms.
- ***FPGAs*** *are faster* than both microcontrollers and digital signal processors and support *parallel computing*; but their production cost and the programming difficulty make them *less suitable.*
- ***ASICs*** have *higher bandwidths*; they are *the smallest* in size, *perform much better*, and *consume less power* than any of the other processing types; but have a *high cost* of production owing to the complex design process.

Node Architecture: Communication Interfaces

COMMUNICATIONS

- Echo Cancellation
- Digital PBXs
- Line Repeaters
- Modems
- Global Positioning
- Sound/Modem/Fax Cards
- Cellular Phones
- Speaker Phones
- Video Conferencing
- ATMs

VOICE/SPEECH

- Speech Recognition
- Speech Processing/Vocoding
- Speech Enhancement
- Text-to-Speech
- Voice Mail

PRO-AUDIO

- AV Editing
- Digital Mixers
- Home Theater
- Pro Audio

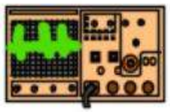
DSP

CONSUMER

- Radar Detectors
- Power Tools
- Digital Audio / TV
- Music Synthesizers
- Toys / Games
- Answering Machines
- Digital Speakers

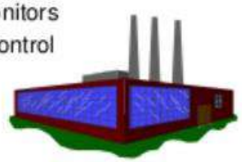
INSTRUMENTATION

- Spectrum Analyzers
- Seismic Processors
- Digital Oscilloscopes
- Mass Spectrometers



INDUSTRIAL/CONTROL

- Robotics
- Numeric Control
- Power Line Monitors
- Motor/Servo Control



MEDICAL

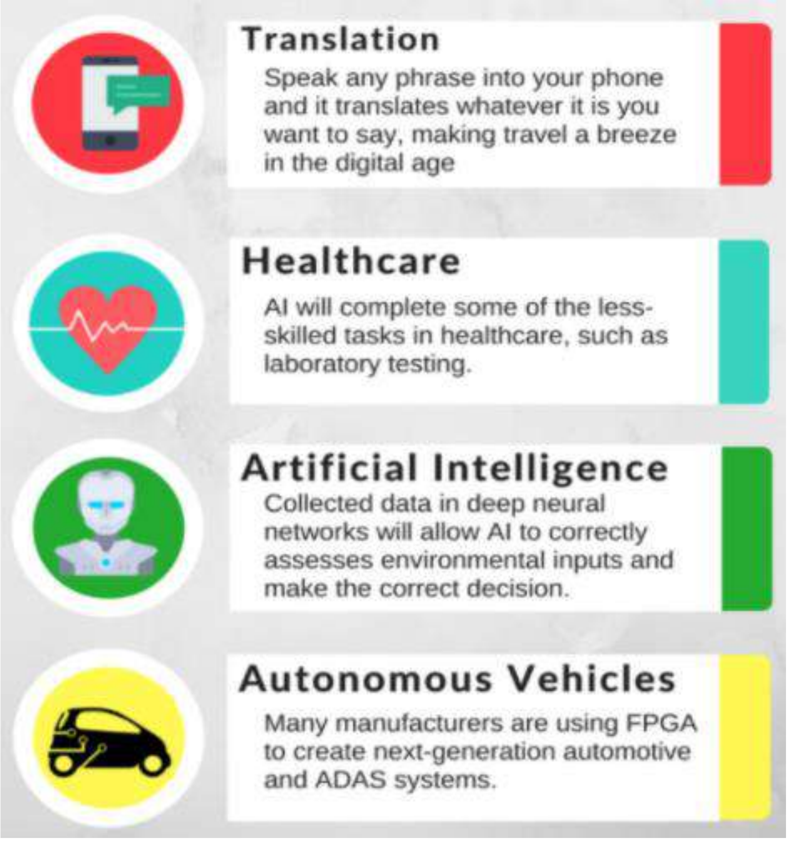
- Patient Monitoring
- Ultrasound Equipment
- Diagnostic Tools
- Fetal Monitors
- Life Support Systems
- Image Enhancement

MILITARY

- Secure Communications
- Sonar Processing
- Image Processing
- Radar Processing
- Navigation, Guidance

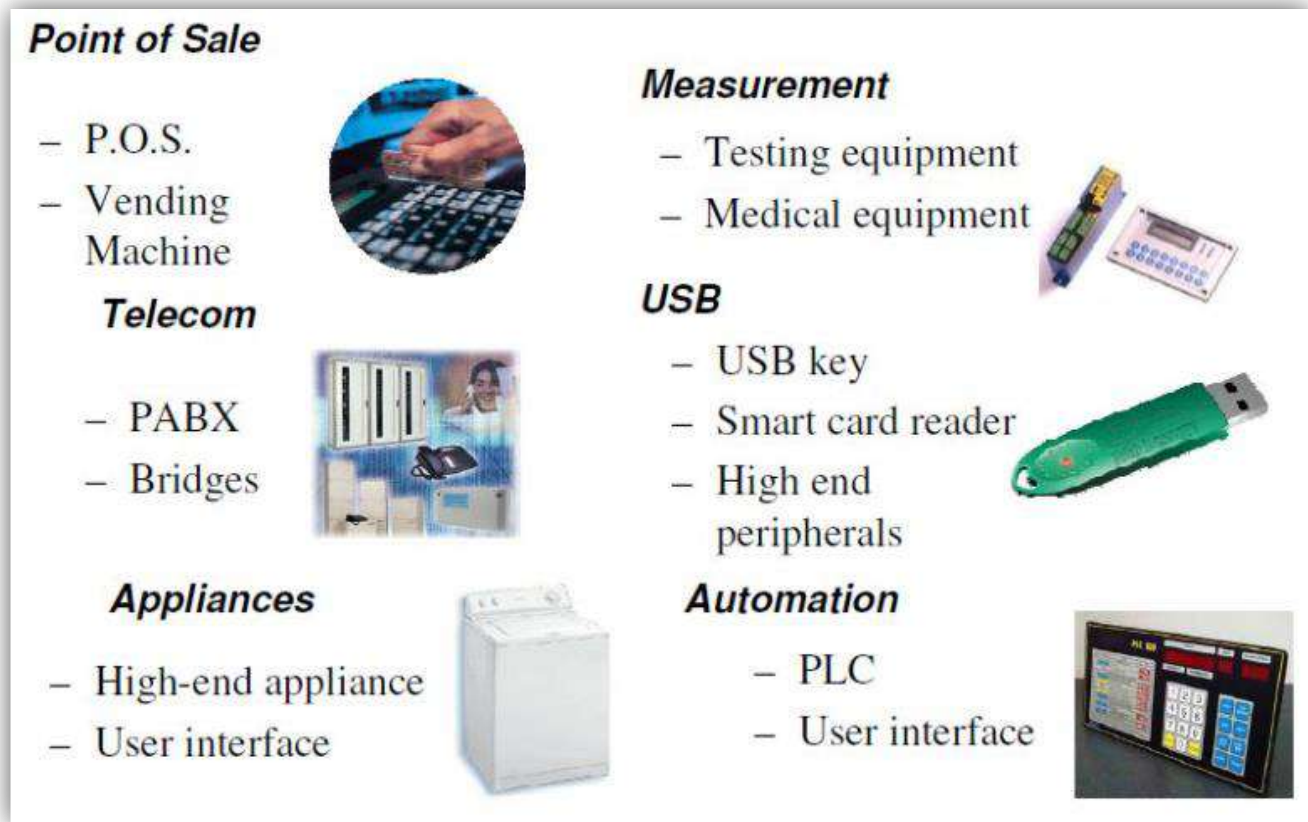
Node Architecture: Communication Interfaces

4 Popular FGPA Application Areas



Node Architecture: Communication Interfaces

Microcontroller Application Areas



Node Architecture: Communication Interfaces

ASIC Application Areas

Apple A6 Processor

Aerospace Engineering

Node Architecture: Communication Interfaces

- *Fast* and *energy efficient data transfer* between the subsystems of a wireless sensor node *is vital.*
  - However, the practical size of the node puts restriction on system buses
  - Communication via a parallel bus is *faster* than a serial transmission
  - A parallel bus needs *more space*

- Therefore, considering *the size of the node*, *parallel buses* are never supported.

- The choice is often between *serial interfaces* :
  - Serial Peripheral Interface (SPI)
  - General Purpose Input/Output (GPIO)
  - Secure Data Input/Output (SDIO)
  - Inter-Integrated Circuit (I2C)

- Among these, the most commonly used buses are *SPI* and *I2C.*

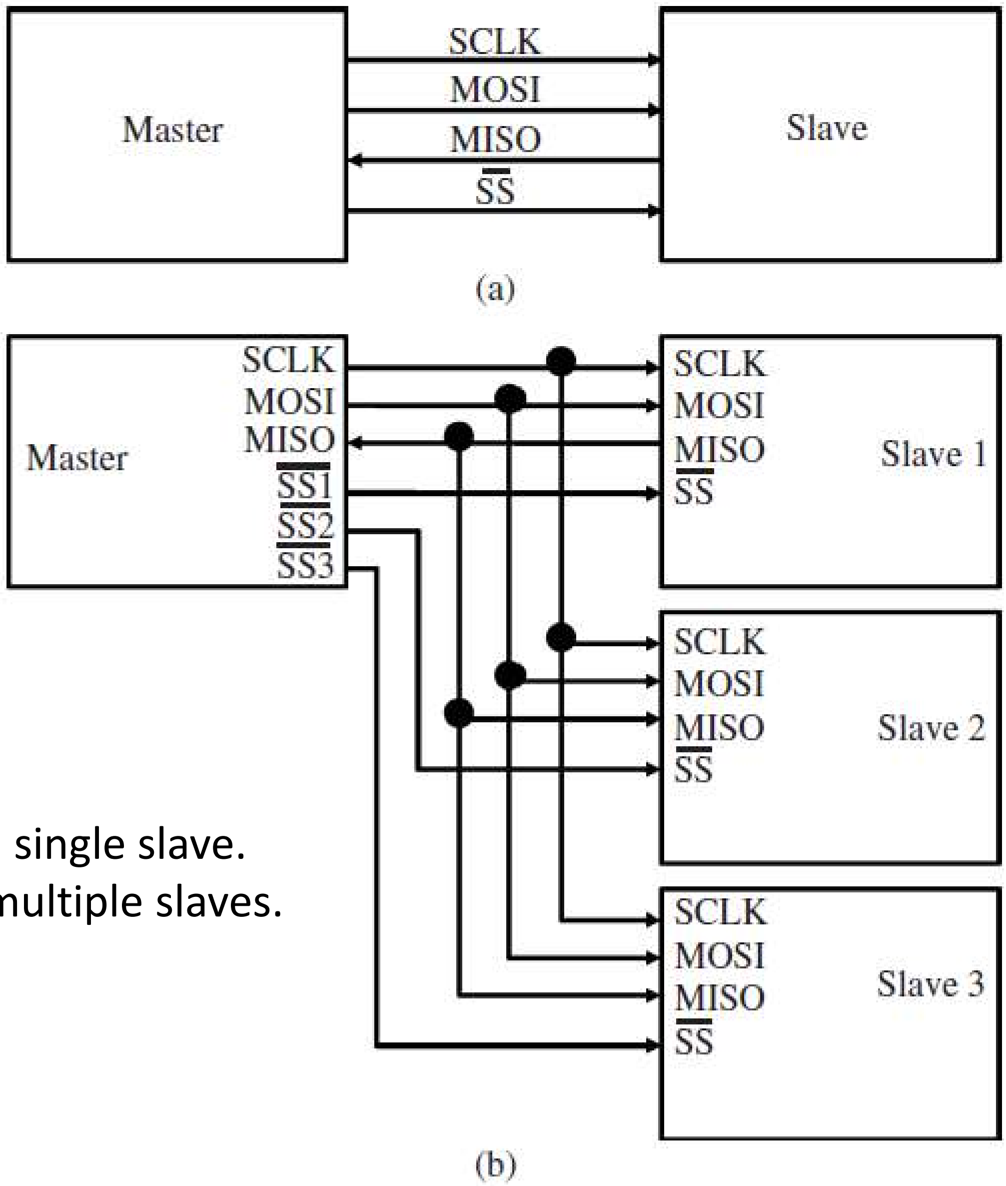
Node Architecture: Communication Interfaces

Serial Peripheral Interface (SPI)

- SPI (Motorola, in the mid-80s)!
  - *high-speed, full-duplex synchronous* serial bus
  - does *not* have an official standard, but use of the SPI interface should conform to the implementation specification of others - correct communication

- The SPI bus defines *four pins*:
  - MOSI (MasterOut/SlaveIn)
    - Used to transmit data *from the master to the slave* when a device is configured as a master
  - MISO (MasterIn/SlaveOut)
  - SCLK (Serial Clock)
    - Used by the *master* to send the clock signal that is needed to *synchronize transmission*
    - Used by the *slave* to read this signal synchronize transmission
  - CS (Chip Select) - communicate via the CS port

Node Architecture: Communication Interfaces



Connecting devices with the SPI bus.
(a) A single master connecting with a single slave.
(b) A single master connecting with multiple slaves.

Node Architecture: Communication Interfaces

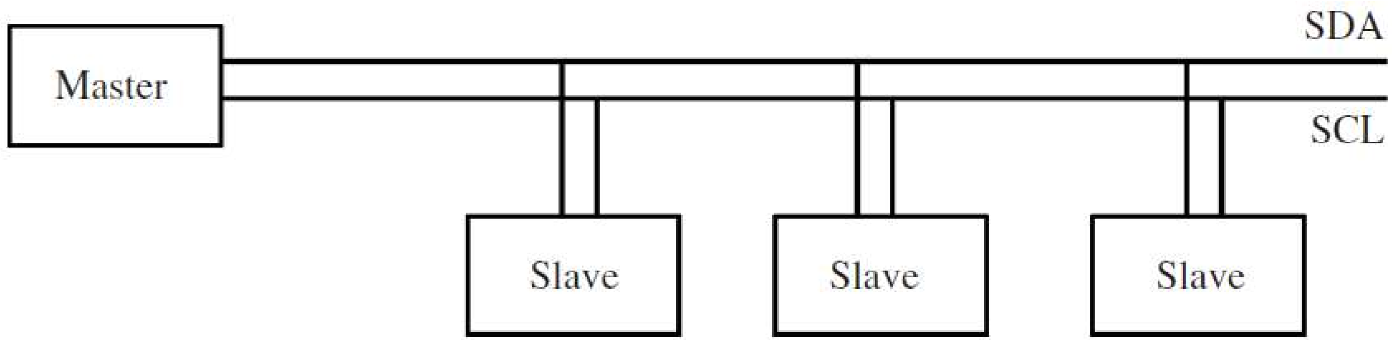
- Both master and slave devices hold a shift register
- Every device in every transmission must read and send data
- SPI supports a *synchronous communication protocol*
  - The master and the slave must agree on the timing
  - Master and slave should agree on two additional parameters
    - clock polarity (*CPOL*) - defines whether a clock is used as high- or low-active
    - clock phase (*CPHA*) - determines the times when the data in the registers is allowed to change and when the written data can be read

SPI mode	CPOL	CPHA	Description
0	0	0	SCLK is low-active. Sampling is allowed on odd clock edges. Data changes on even clock edges.
1	0	1	SCLK is low-active. Sampling is allowed on even clock edges. Data changes on odd clock edges.
2	1	0	SCLK is high-active. Sampling is allowed on odd clock edges. Data changes on even clock edges.
3	1	1	SCLK is high-active. Sampling is allowed on even clock edges. Data changes on odd clock edges.

Node Architecture: Communication Interfaces

Inter-Integrated Circuit

- Every device type that uses I2C must have a unique address that will be used to communicate with a device
- In earlier versions, a *7 bit address* was used, allowing 112 devices to be uniquely addressed - due to an increasing number of devices, it *is insufficient*
- Currently I2C uses *10 bit addressing*
- I2C is a *multi-master half-duplex synchronous serial* bus
    - only two bidirectional lines: (unlike SPI, which uses four)
        1. Serial Clock (SCL)
        2. Serial Data (SDA)



Connecting devices with the I2C serial bus

Node Architecture: Communication Interfaces

Inter-Integrated Circuit

- Since each master generates its own clock signal, communicating devices must *synchronize their clock speeds*
  - a slower slave device could wrongly detect its address on the SDA line while a faster master device is sending data to a third device!
- I2C requires arbitration between master devices *wanting* to send or receive data at the same time
  - *no* fair arbitration *algorithm*
  - rather the master that holds the SDA line low for *the longest time wins* the medium

Node Architecture: Communication Interfaces

Inter-Integrated Circuit

- enables a device to read data *at a byte level* for a fast communication
  - the device can hold the SCL low until it completes reading or sending the next byte - called *handshaking*

- The *aim* of I2C is *to minimize costs* for connecting devices
  - accommodating lower transmission speeds

- I2C defines two speed modes:
  - *a fast-mode* - a bit rate of up to 400Kbps
  - *high-speed mode* - a transmission rate of up to 3.4 Mbps
  - they are downwards compatible to ensure communication with older components

Node Architecture: Communication Interfaces

SPI	I^2C
• Four lines enable full-duplex transmission.	• Two lines reduce space and simplify circuit layout. Lowers costs.
• No addressing is required due to CS. This reduces overhead and increases throughput. However, one needs additional hardware configurations to connect more than one slave.	• Addressing enables multimaster mode, which in turn enables more than one device to initiate communication.
• Allowing only one master avoids conflicts.	• Multimaster mode is prone to conflicts when two or more master devices communicate simultaneously. Arbitration is required.
• Hardware requirement support increases with an increasing number of connected devices and therefore raises costs.	• Hardware requirement is independent of the number of devices using the bus.
• The master's clock is configured according to the slave's speed. This frees the slaves from requiring clocking a device. However, speed adaptation slows down the master.	• Slower devices may stretch the clock thereby increasing latency and keeping other devices waiting for accessing the bus.
• Speed depends on the maximum speed of the slowest device.	• Speed is limited to 3.4 MHz and all devices need to support the highest speed that is used in the system, otherwise a slower device may wrongly detect its device address.
• Heterogeneous register size allows flexibility in the devices that are supported.	• Homogeneous register size reduces overhead, since no additional control bits are required to be transmitted.
• Combined registers imply every transmission should be read.	• Devices that do not read or provide data are not forced to provide potentially useless bytes.
• The absence of an official standard leads to application specific implementations.	• Official standard eases integration of devices since developers can rely on a certain implementation.

Operating System

- An operating system (OS) in a WSN is
  - A thin software layer that
  - Logically resides between the node's hardware and the application.
  - Provides basic programming abstractions to application developers.

- Its main task is to
  - Enable applications to interact with hardware resources, to
  - Schedule and prioritize tasks, and
  - Arbitrate between contending applications and services that try to seize resources.

Operating System

- Additional features include:
  - Memory Management;
  - Power Management;
  - File Management;
  - Networking;
  - A set of programming environments and tools – commands, command interpreters, command editors, compiler, debuggers, etc. – to enable users to develop, debug, and execute their own programs; and
  - Legal entry points into the operating system for accessing sensitive resources such as writing to input components.

Operating System

Operating systems are classified as:

- Single-task/Multitasking and

Single-task OS	Multitasking OS
A single-task operating system processes one task at a time.	a multitasking operating system can execute multiple tasks simultaneously.
It requires a large amount of memory to manage the states of multiple tasks but they enable tasks with different complexity to execute in parallel.	In a single-task OS, one task is executed at a time, therefore, as a rule tasks should have a short duration.

- Single-user/Multiuser operating systems
  - In a single-user OS, one user (the owner of the resources) is active at a time, whereas a multi-user operating system allows multiple users to share the resources of a single system at the same time

Operating System

The choice of a particular OS depends on several factors;

- ***Functional Aspects***
  - *Data Types*
  - *Scheduling*
  - *Stacks*
  - *System Calls*
  - *Handling Interrupts*
  - *Multithreading*
  - *Thread-based vs. Event-based Programming*
  - *Memory Allocation*

- ***Non-Functional Aspects***
  - Separation of Concern
  - System Overhead
  - Portability
  - Dynamic Reprogramming

- ***Prototypes***
  - TinyOS
  - SOS
  - Contiki
  - LiteOS

Operating System- Functional Aspects

Data Types

- Different subsystems of WSN communicate with each other for various reasons such as to exchange data, delegate functionalities, and signaling.

- Interactions between the different subsystems take place through:
  - Well-formulated protocols
  - Data types

- Complex data types have strong expression power but consume resources - struct and enum.

- Simple data types are resource efficient but have limited expression capability - C programming language.

Operating System- Functional Aspects

Scheduling

- Task scheduling is process to organize, prioritize and execute the tasks to determines the efficiency of the OS.
- Two scheduling mechanisms:

1. ***Queuing-based Scheduling***
    - In a queuing-based scheduling, tasks originating from the various subsystems are temporarily stored in a queue and executed serially according to a predefined rule.
    - Some operating systems enable tasks to specify priority levels so that they can be given precedence.
    - Types of Queuing-based Scheduling
        - ***FIFO (first-in-first-out) Scheme***
            - Tasks are processed according to their arrival time.
            - A task that arrives first will be executed first as soon as the processor is free.
            - It's non-preemptive scheduling, OS will execute the task to the end before another task is admitted for execution.
            - FIFO - the simplest and has minimum system overhead, but treats tasks unfairly; since tasks of long duration may block short-duration tasks for a long time.
            - Poor in performance as average wait time is high.
            - Easy to understand and implement.
            - Its implementation is based on FIFO queue.

FCFS (Example)

Process	Duration	Order	Arrival Time
P1	24	1	0
P2	3	2	0
P3	4	3	0

Gantt Chart :

P1 waiting time : 0
P2 waiting time : 24
P3 waiting time : 27

The Average waiting time :

(0+24+27)/3 = 17

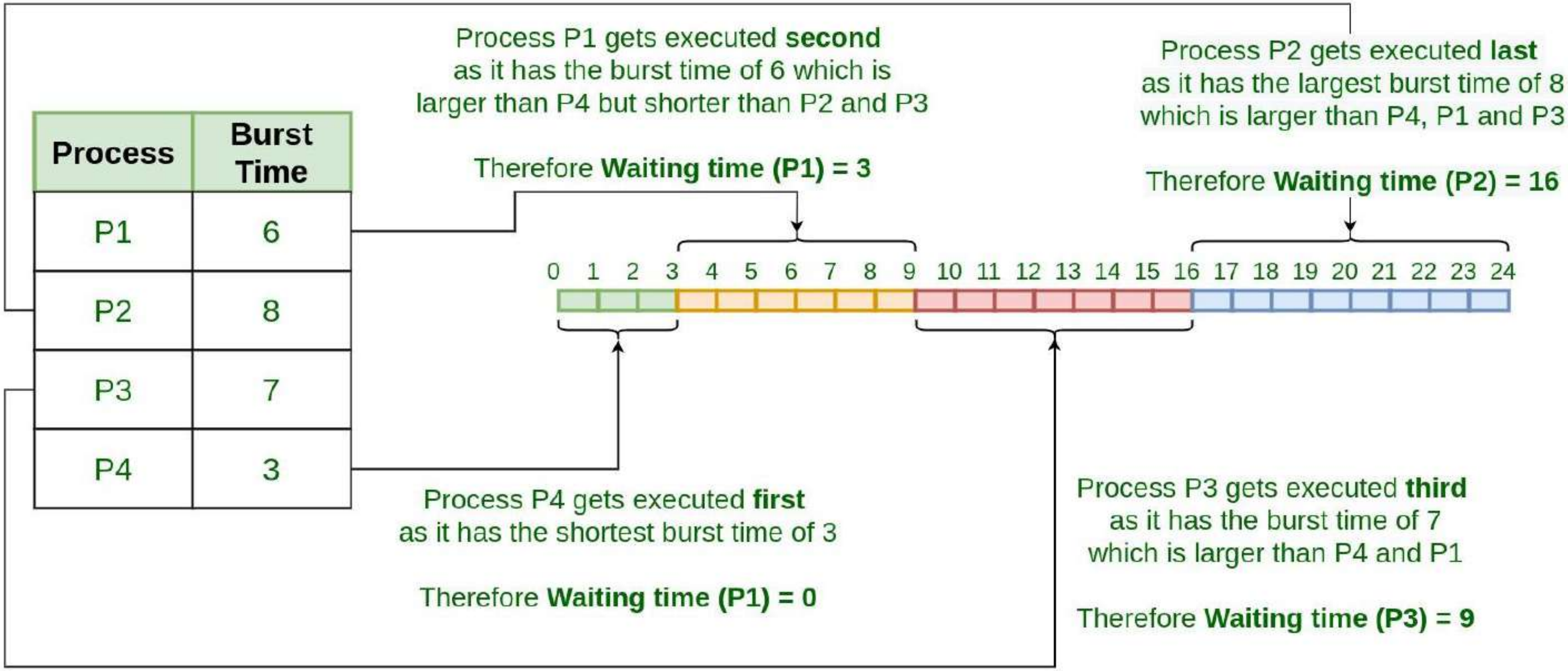
Operating System- Functional Aspects

❑ *Shortest Job First (SJF)*

- In a sorted queue scheme, tasks in a queue are sorted according to some criteria.
- One way is to sort tasks according to their estimated execution duration. This approach prevents long-duration tasks from blocking short-duration tasks. The approach is also known as the shortest job first (SJF) rule.
- Sorted queue - e.g., shortest job first (SJF), Sorting incurs system overhead, since each task in the queue must be evaluated to estimate execution duration and to sort tasks accordingly.
- This is the best approach to minimize waiting time.
- It is of two types:
    - Non Pre-emptive
    - Pre-emptive
- To successfully implement it, the burst time/duration time of the processes should be known to the processor in advance, which is practically not feasible all the time.
- This scheduling algorithm is optimal if all the jobs/processes are available at the same time. (either Arrival time is 0 for all, or Arrival time is same for all)
- Easy to implement in Batch systems where required CPU time is known in advance.

Operating System- Functional Aspects

Shortest Job First (SJF) Example



Operating System- Functional Aspects

- ***Non-preemptive SJF***
  - ✓ Sometimes process with short burst time have to wait for the current process's execution to finish, because in Non Pre-emptive SJF, on arrival of a process with short duration, the existing job/process's execution is not halted/stopped to execute the short job first.
  - ✓ This leads to the problem of **Starvation**, where a shorter process has to wait for a long time until the current longer process gets executed.
  - ✓ This happens if shorter jobs keep coming, but this can be solved using the concept of **aging**.
- ***Pre-emptive Shortest Job First***
  - ✓ In Preemptive Shortest Job First Scheduling, jobs are put into ready queue as they arrive, but as a process with **short burst time** arrives, the existing process is preempted or removed from execution, and the shorter job is executed first.

Operating System- Functional Aspects

Round Robin Scheduling

- It is mainly designed for time-sharing systems.

- This algorithm is similar to FCFS scheduling, but in Round Robin(RR) scheduling, preemption is added which enables the system to switch between processes.

- A fixed time is allotted to each process, called a **quantum**, for execution.

- Once a process is executed for the given time period that process is preempted and another process executes for the given time period.

- **Context switching** is used to save states of preempted processes.

- This algorithm is simple and easy to implement.

- This algorithm is **starvation-free** as all processes get a fair share of CPU.

- The length of time quantum is generally from 10 to 100 milliseconds in length.

Operating System- Functional Aspects

❑ *Round Robin Scheduling*

- It's Preemptive Algorithms.
- One of the oldest, easiest, and fairest algorithm.
- It is a real-time algorithm because it responds to the event within a specific time limit.
- In this algorithm, the time slice should be the minimum that is assigned to a specific task that needs to be processed. Though it may vary for different operating systems.
- It is a **hybrid model** and is clock-driven in nature.
- It is a widely used scheduling method in the traditional operating system.

Round Robin Scheduling Example

TIME SLICE = 4

PROCESS	ARRIVAL TIME	BURST TIME	
		TOTAL	REMAINING
P1	0	8	8
P2	1	6	6
P3	3	3	3
P4	5	2	2
P5	6	4	4

READY QUEUE:

P1	P2	P3	P1	P4	P5	P2

GANTT CHART:

P1	P2	P3	P1	P4	P5	P2

0 4 8 11 15 17 21 23

Process	Duration	Order	Arrival Time
P1	3	1	0
P2	4	2	0
P3	3	3	0

Suppose time quantum is 1 unit.

P1	P2	P3	P1	P2	P3	P1	P2	P3	P2

0 **10**

P1 waiting time : 4

P2 waiting time: 6

P3 waiting time: 6

The average waiting time(AWT) : (4+6+6)/3=5.33

Operating System- Functional Aspects

Regardless of how tasks are executed, a scheduler can be either!

- A non-preemptive scheduler - a task is executed to the end, may not be interrupted by another task.
- Preemptive scheduler - a task of higher priority may interrupt a task of low priority.

Parameter	Preemptive Scheduling	Non-preemptive Scheduling
Basic	In this resources(CPU Cycle) are allocated to a process for a limited time.	Once resources(CPU Cycle) are allocated to a process, the process holds it till it completes its burst time or switches to waiting state.
Interrupt	Process can be interrupted in between.	Process can not be interrupted until it terminates itself or its time is up.
Starvation	If a process having high priority frequently arrives in the ready queue, low priority process may starve.	If a process with long burst time is running CPU, then later coming process with less CPU burst time may starve.
Overhead	It has overheads of scheduling the processes.	It does not have overheads.
CPU Utilization	In preemptive scheduling, CPU utilization is high.	It is low in non preemptive scheduling.
Examples	Examples of preemptive scheduling are Round Robin and Shortest Remaining Time First.	Examples of non-preemptive scheduling are First Come First Serve and Shortest Job First.

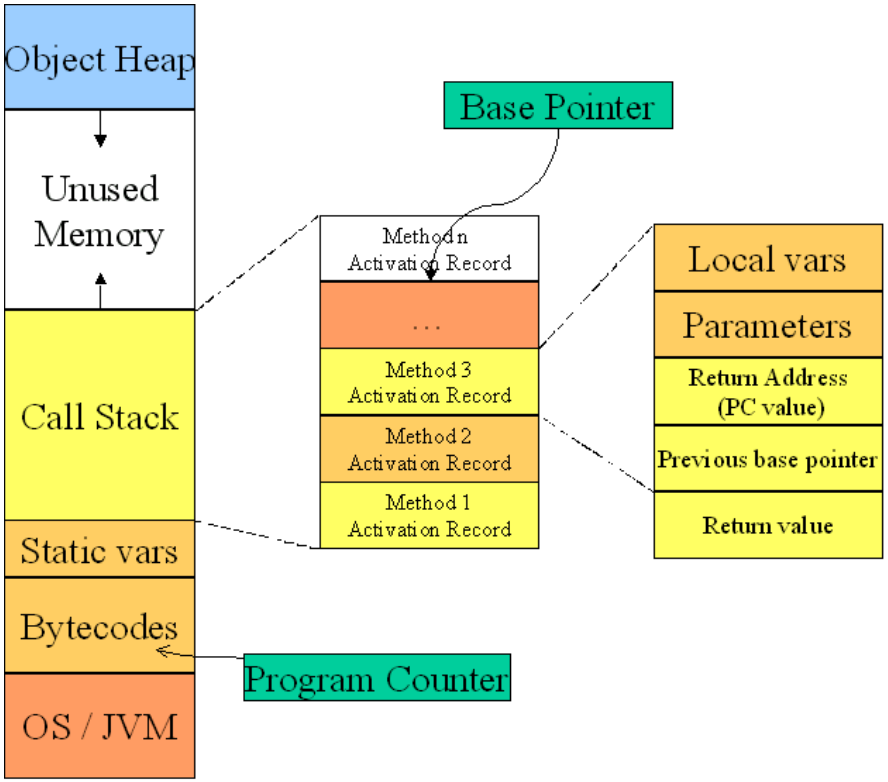
Operating System- Functional Aspects

Stack

- A stack is a data structure that is used to temporarily store data objects in memory by piling one upon another.
- The objects are accessed on a last-in first-out (LIFO) basis.
- The processor core uses stacks to store system state information when it begins executing subroutines. This way it “remembers” where to return after the subroutine is completed.
- Subroutines can also call other subroutines by storing the state of the current subroutine on top of the previous state information in the stack. When the subroutine is completed, the processor pulls the first address it finds on the top of the stack and jumps to that location.
- In a multithreaded OS, each thread requires its own stack to manage state information.
- This is one of the reasons why multithreaded operating systems are expensive in WSNs.

Operating System- Functional Aspects

Stack



Call and return process

- When a method/function is called

1. An **Activation Record is** created; its size depends on the number and size of the local variables and parameters.
2. The **Base Pointer** "Register" value is saved in the special location reserved for it
3. The **Program Counter** "Register" value is saved in the **Return Address** location
4. The Base Pointer is now reset to the new base (top of the call stack prior to the creation of the AR)
5. The Program Counter is set to the location of the first bytecode of the method being called
6. Copies the calling parameters into the Parameter region
7. Initializes local variables in the local variable region

- While the method executes, the local variables and parameters are simply found by adding a constant associated with each variable/parameter to the Base Pointer.

- When a method returns

1. Get the program counter from the activation record and replace what's in the PC register
2. Get the base pointer value from the AR and replace what's in the BP register
3. Pop the Activation Record entirely from the stack.

There is also a **Stack Pointer** that allow other temporary use of the stack above the top AR.

Operating System- Functional Aspects

System Calls

- A **system call** is a mechanism that provides the interface between a process and the operating system.

- It is a programmatic method in which a computer program requests a service from the kernel of the OS.

- System call offers the services of the operating system to the user programs via API (Application Programming Interface).

- System calls are the only entry points for the kernel system.

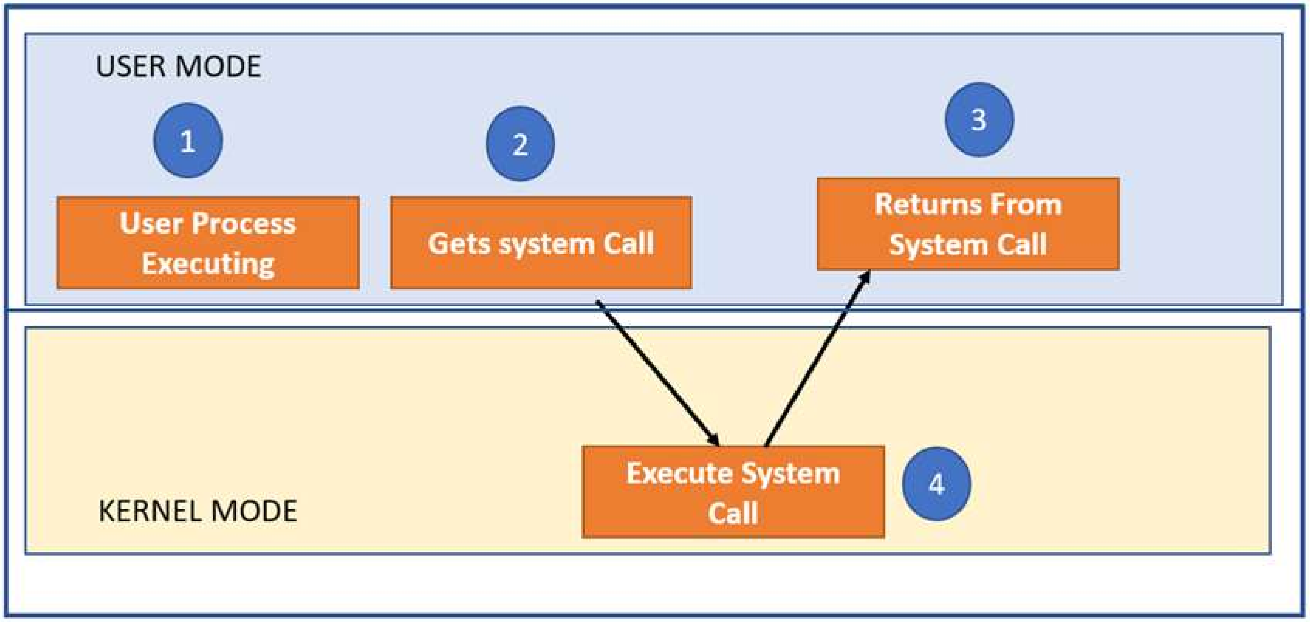
- When a program makes a system call, the mode is switched from user mode to kernel mode. This is called a **context switch**.

- Generally, system calls are made by the user level programs in the following situations:
  - Creating, opening, closing and deleting files in the file system.
  - Creating and managing new processes.
  - Creating a connection in the network, sending and receiving packets.
  - Requesting access to a hardware device, like a mouse or a printer.

Operating System- Functional Aspects

System Calls

- Users invoke these operations whenever they wish to access a hardware resource such as a sensor, watchdog timer, or the radio without the need to concern themselves how the hardware is accessed



Step 1) The processes executed in the user mode till the time a system call interrupts it.

Step 2) After that, the system call is executed in the kernel-mode on a priority basis.

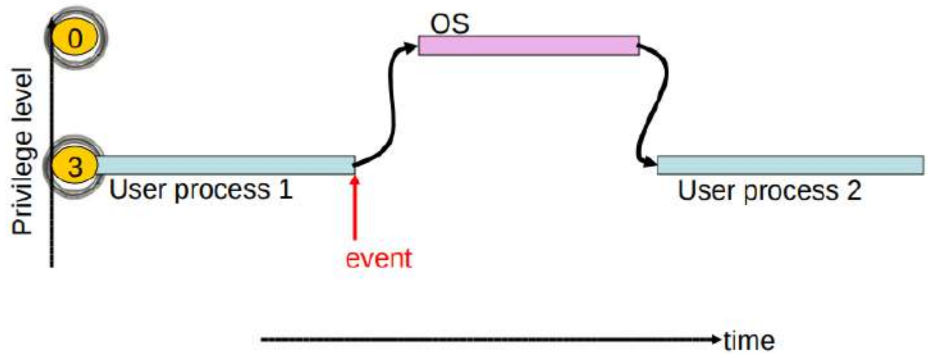
Step 3) Once system call execution is over, control returns to the user mode.,

Step 4) The execution of user processes resumed in Kernel mode.

Operating System- Functional Aspects

Handling Interrupts

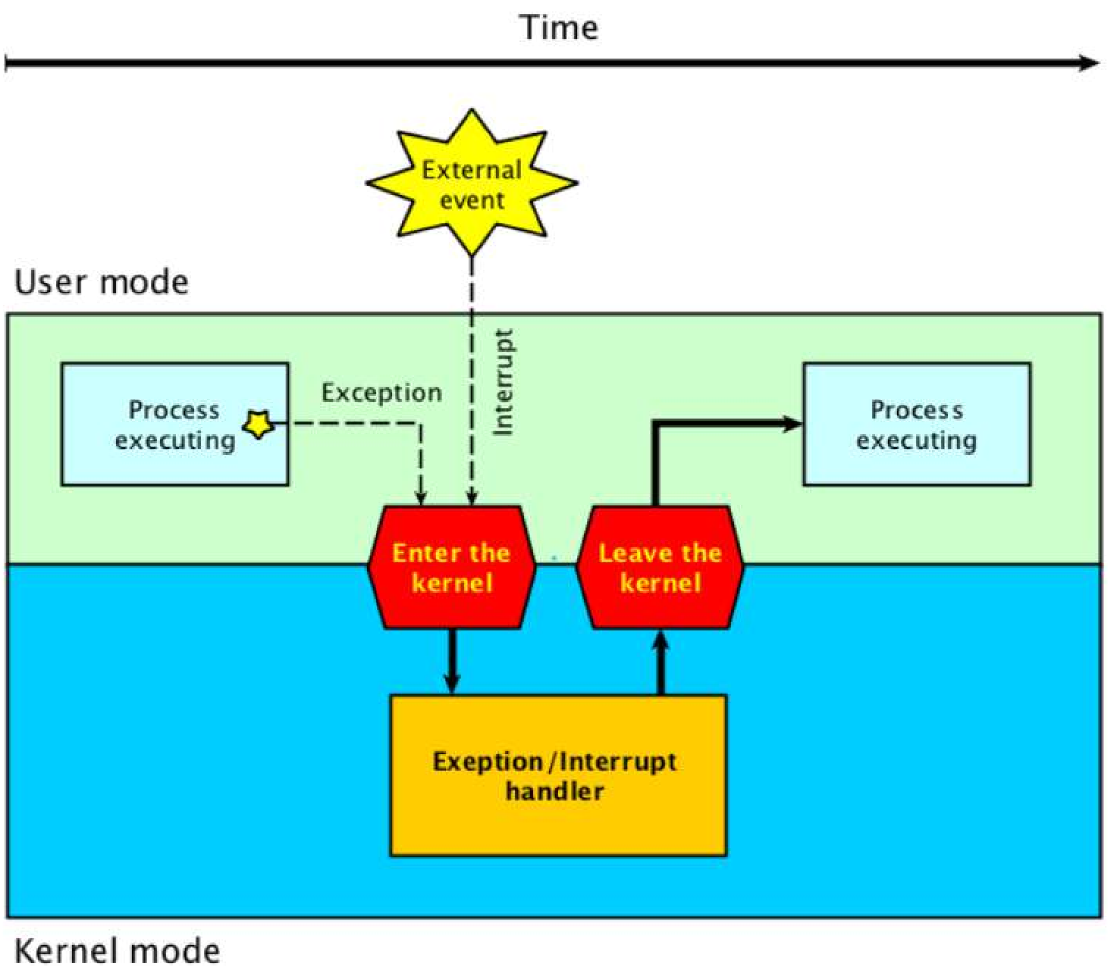
- An interrupt is an asynchronous signal generated by
  - a hardware device (a sensor, a watchdog timer, a radio)
  - several system events
  - OS itself



- An interrupt causes:
  - the processor to interrupt executing the present instruction
  - to call for an appropriate interrupt handler
- Interrupt signals can have different priority levels, a high priority interrupt can interrupt a low level interrupt
- Interrupt mask: let programs choose whether or not they wish to be interrupted.

Operating System- Functional Aspects

- Interrupts are important because they give the user better control over the computer.
- Without interrupts, a user may have to wait for a given application to have a higher priority over the CPU to be ran. This ensures that the CPU will deal with the process immediately.



Operating System- Functional Aspects

Handling Interrupts

There are three types of interrupts:

- **Hardware Interrupts** are generated by hardware devices to signal that they need some attention from the OS.
  - They may have just received some data (e.g., keystrokes on the keyboard or an data on the Ethernet card); or
  - They have just completed a task which the operating system previous requested, such as transferring data between the hard drive and memory.

- **Software Interrupts** are generated by programs when they want to request a system call to be performed by the operating system.

- **Traps** are generated by the CPU itself to indicate that some error or condition occurred for which assistance from the operating system is needed.

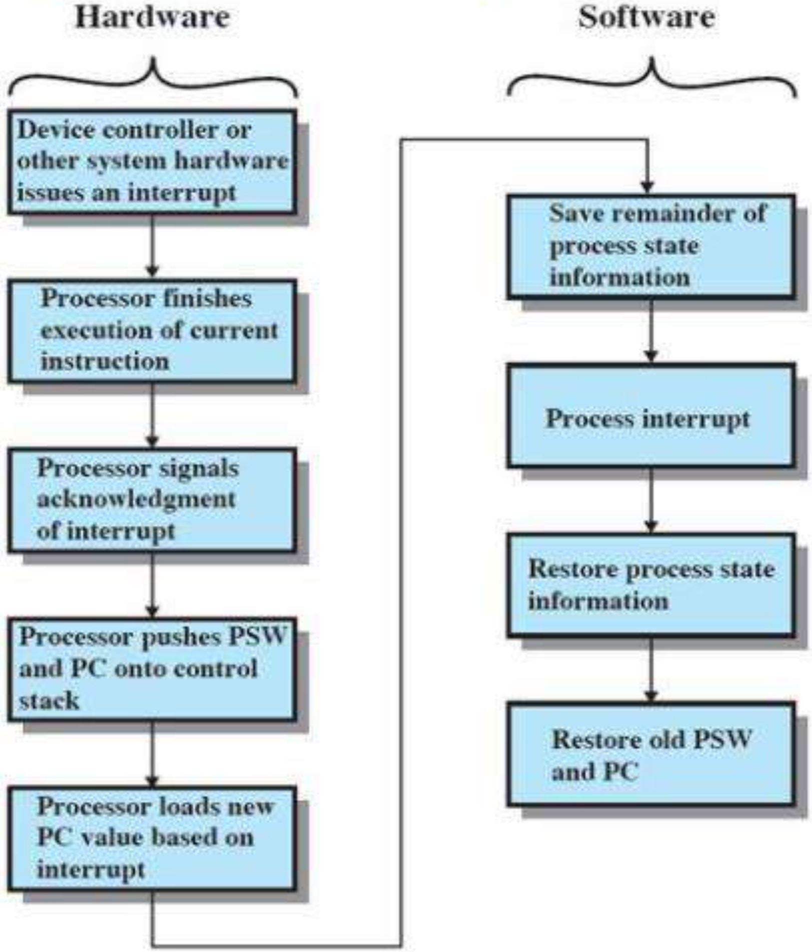
Operating System- Functional Aspects

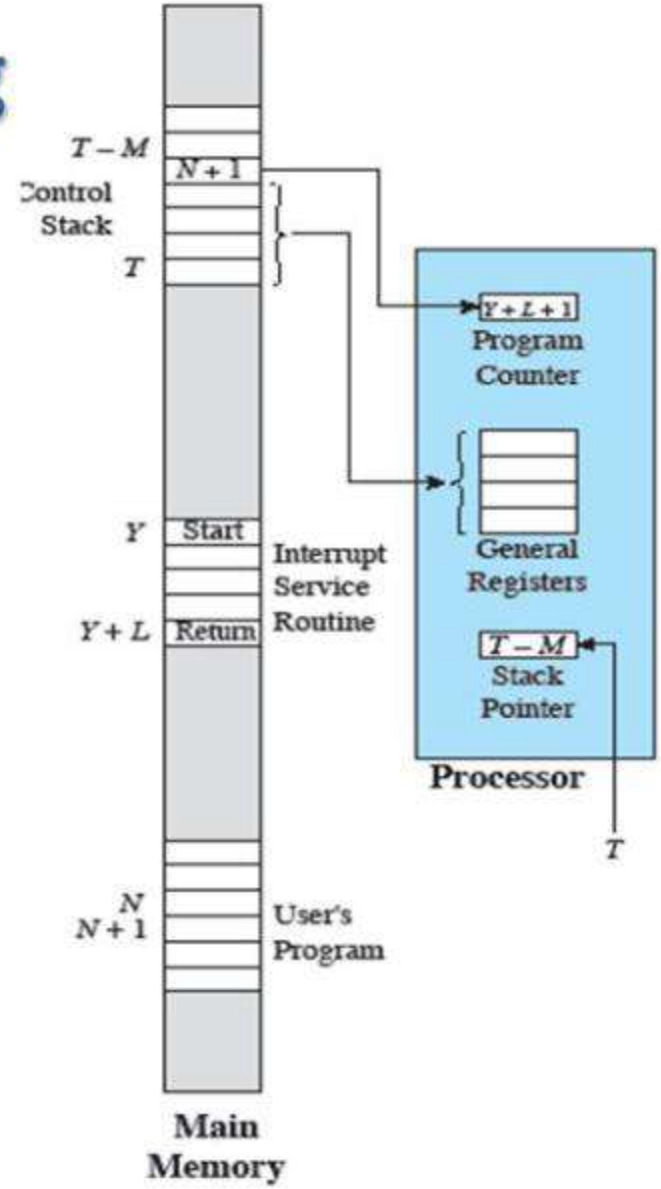
Handling Interrupts

When an interrupt gets active, the microcontroller goes through the following steps –

- The microcontroller closes the currently executing instruction and saves the address of the next instruction (PC) on the stack.
- It also saves the current status of all the interrupts internally (i.e., not on the stack).
- It jumps to the memory location of the interrupt vector table that holds the address of the interrupts service routine.
- The microcontroller gets the address of the ISR from the interrupt vector table and jumps to it. It starts to execute the interrupt service subroutine, which is RETI (return from interrupt).
- Upon executing the RETI instruction, the microcontroller returns to the location where it was interrupted. First, it gets the program counter (PC) address from the stack by popping the top bytes of the stack into the PC. Then, it start to execute from that address.

Simple Interrupt Processing





Operating System- Functional Aspects

Multithreading

- A *thread* is the path taken by a processor or a program during its execution
  - *Multi-threading* - a task is divided into several logical pieces
  - scheduled independent from each other
  - executed concurrently
  - Threads of the same task share a common data and address space and can communicate with each other if necessary.

- Two advantages of a multi-threaded OS:
  1. tasks do not block other tasks
  2. short-duration tasks can be executed along with long-duration tasks

- Threads cannot be created endlessly
  - the creation of threads *slows down* the processor
  - no sufficient resources to divide

- The OS can keep the number of threads to a *manageable size* using a thread pool

Operating System- Functional Aspects

Thread-based Vs. Event-based Programming

- Decision whether to use threads or events programming:
  - need for separate stacks
  - need to estimate maximum size for saving context information

- ***Thread-based programs*** use multiple threads of control within:
  - a single program
  - a single address space

- ***Advantage:***
  - a thread blocked can be suspended while other tasks are executed in different threads

- ***Disadvantages:***
  - must carefully protect shared data structures with locks
  - use condition variables to coordinate the execution of threads
  - In general, program code written for multithreading environments is complex, bug-prone, and may lead to deadlocks and race conditions.

Operating System- Functional Aspects

- In ***event-based programming***: use events and event handlers event-handlers register with the OS scheduler to be notified when a named event occurs.

- A loop function:
  - polls for events
  - calls the appropriate event-handlers when events occur

- An event is processed to completion
  - unless its handler reaches at a blocking operation (callback and returns control to the scheduler).

Operating System- Functional Aspects

Memory Allocation

- The memory unit is a precious resource
- Reading and writing to memory is costly
- How and for how long a memory is allocated for a piece of program determines the speed of task execution

Memory can be allocated to a program:

- *Statically* - a frugal approach, but the requirement of memory *must be known* in advance
  - memory is used efficiently
  - runtime adaptation is not allowed

- *Dynamically* - the requirement of memory is *not known* in advance (on a transient basis)
  - enables flexibility in programming
  - but produces a considerable management overhead

Operating System- Functional Aspects

Memory Allocation

- As a strategy to increase the memory capacity of a node, most architectures use EEPROM or flash memory to store program code.

- Consequently, it is possible to deploy relatively complex applications and communication protocols.

- However, reading and writing to flash memory is costly with respect to energy consumption.

Operating System- Non-functional Aspects

Separation of Concern

- In general-purpose operating systems,
  - separation between the operating system and the applications layer.
  - interact with each other through well-defined interfaces and system calls.
  - operating system itself has several distinct services that can be upgraded, debugged, or altogether removed independently.
- The operation systems can provide:
  - a number of lightweight modules - “wired” together, or
  - an indivisible system kernel + a set of library components for building an application, or
  - a kernel + a set of reconfigurable low-level services
- Separation of concern enables:
  - flexible and efficient reprogramming and reconfiguration
  - An update or an upgrade can be made as a whole or in part as required. It enables the efficient use of communication bandwidth as well as memory space during a software update.

Operating System- Non-functional Aspects

System Overhead

- An operating system executes program code – requires its own share of resources

- The resources consumed by the OS are the system's overhead, it depends on
  - the size of the operating system
  - the type of services that the OS provides to the higher-level services and applications

- Presently WSN, resources are measured in terms of kilobytes and a few megahertz.

- The resources have to be shared by programs that carry out:
  - sensing
  - data aggregation
  - self-organization
  - network management
  - network communication

Operating System- Non-functional Aspects

Portability

- Ideally, operating systems should be able to co-exist and collaborate with each other
- However, existing operating systems do *not* provide this type of support
- In order to accommodate unforeseen requirements, operating systems should be portable and extensible

Operating System- Non-functional Aspects

Dynamic Reprogramming

- Once a wireless sensor network is deployed, it may be necessary to reprogram some part of the application or the operating system for the following reasons:
  - the network may not perform optimally
  - both the application requirements and the network's operating environment can change over time
  - may be necessary to detect and fix bugs

- Manual replacement may not be feasible - develop an operating system to provide dynamic reprogramming support, which depends on
  - clear separation between the application and the OS
  - the OS can receive software updates and assemble and store it in memory
  - OS should make sure that this is indeed an updated version
  - OS can remove the piece of software that should be updated and install and configure the new version
  - all these consume resources and may cause their own bugs

Operating System- Non-functional Aspects

Dynamic Reprogramming

- Software reprogramming (update) requires robust *code dissemination protocols*:
  - splitting and compressing the code
  - ensuring code consistency and version controlling
  - providing a robust dissemination strategy to deliver the code over a wireless link

Operating System- Prototypes

Tiny OS

- TinyOS is *the most widely used*, *richly documented*, and *tool-assisted* runtime environment in WSN
  - static memory allocation
  - event-based system

- TinyOS's architecture consists of
  - a scheduler
  - a set of components, which are classified into!
    1. configuration components - "wiring" (how models are connected with each other)
    2. modules - the basic building blocks of a TinyOS program

Operating System- Prototypes

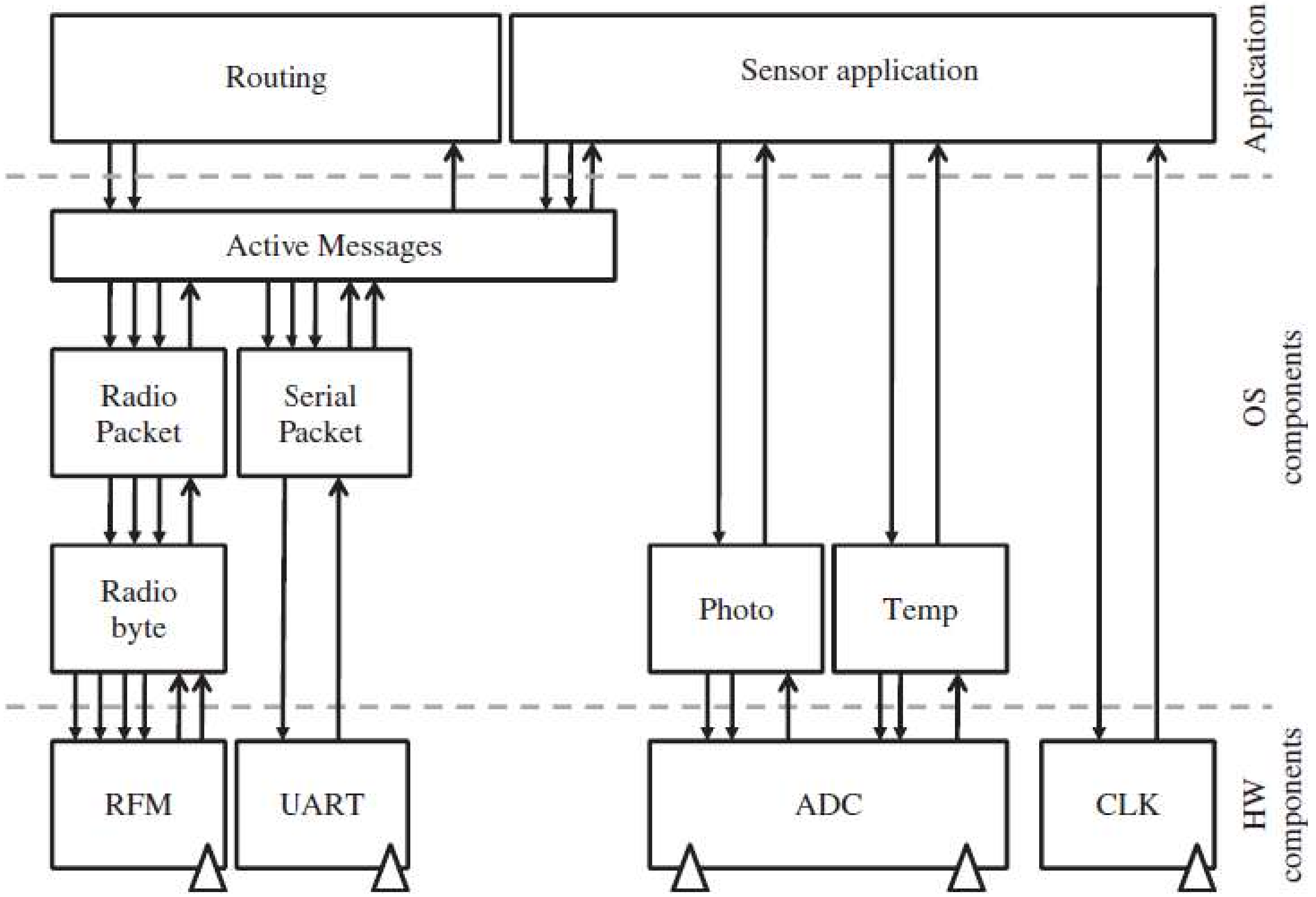
Tiny OS

- A component is made up of
  - a frame
  - command handlers
  - event handlers
  - a set of non-preemptive tasks

- A component is similar to an object in object-based programming languages:
  - it encapsulates state and interacts through well-defined interfaces
  - an interface that can define commands, event handlers, and tasks

Operating System- Prototypes

Tiny OS



Logical distinction between the low-level and high-level components

Operating System- Prototypes

Tiny OS

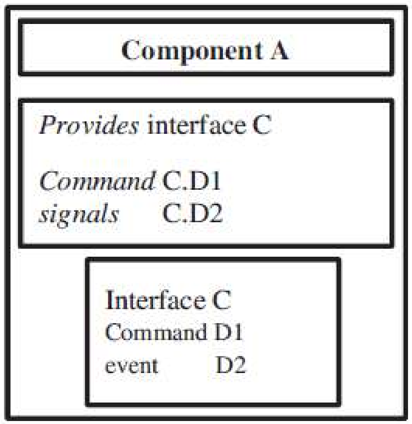
- Components are structured hierarchically and communicate with each other through commands and events:
  - higher-level components issue commands to lower-level components
  - lower-level components signal events to higher-level components

- In Figure, two components at the highest level communicate asynchronously through active messages
  - routing component - establishing and maintaining the network
  - sensor application - responsible for sensing and processing

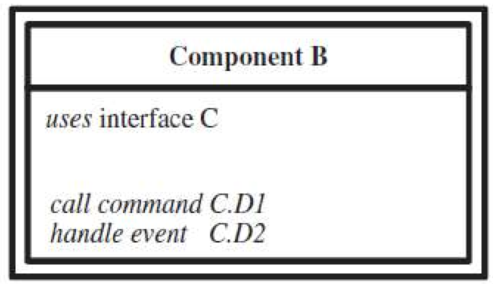
Operating System- Prototypes

Tiny OS

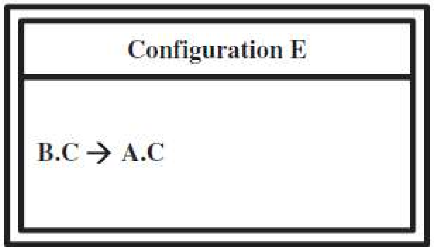
The logical structure of components and component configurations



A TinyOS component providing an interface.



A TinyOS component that uses an interface.



A TinyOS configuration that wires an interface provider and an interface user.

Component A declares its service by providing *interface C*, which in turn provides *command D1* and signals *event D2*.

Component B expresses interest in *interface C* by declaring a call to *command D1* and by providing an event handler to process *event D2*.

a binding between *Component A* and *Component B* is established through the *Configuration E*.

Operating System- Prototypes

Tiny OS

- The fundamental building blocks of a TinyOS runtime environment: *tasks, commands,* and *events*
  - enabling effective communication between the components of a single frame
- *Tasks :*
  - *monolithic processes* - should execute to completion – they cannot be preempted by other tasks, though they can be interrupted by events
  - possible to allocate a single stack to store context information
  - call lower level commands; signal higher level events; and post (schedule) other tasks
  - scheduled based on FIFO principle (in TinyOS)

Operating System- Prototypes

Tiny OS

- ***Commands:***
  - non-blocking requests made by higher-level components to
  - lower-level components
  - split-phase operation:
    - a function call returns immediately
    - the called function notifies the caller when the task is completed

- ***Events:***
  - events are processed by the event handler
  - event handlers are called when hardware events occur
  - an event handler may react to the occurrence of an event in different ways
    - deposit information into its frame, post tasks, signal higher level events, or call lower level commands

Operating System- Prototypes

SOS

- **The SOS operating system**
  - establishes a balance between flexibility and resource efficiency
  - supports runtime reconfiguration and reprogramming

- **The SOS operating system consists of:**
  - a kernel :
    - provides interfaces to the underlying hardware
    - provides priority-based scheduling mechanism
    - supports dynamic memory allocation
  - a set of modules – can be loaded and unloaded - a position independent binary
    - enables SOS to dynamically link modules with each other

Operating System- Prototypes

<u>*SOS- Interaction*</u>

Interaction with a module through:

*1. **messages*** **(asynchronous communication)**

- a message that originates from module A to module B
  - the message goes through the scheduler
  - the kernel calls the appropriate message handler in module B and passes the message to it

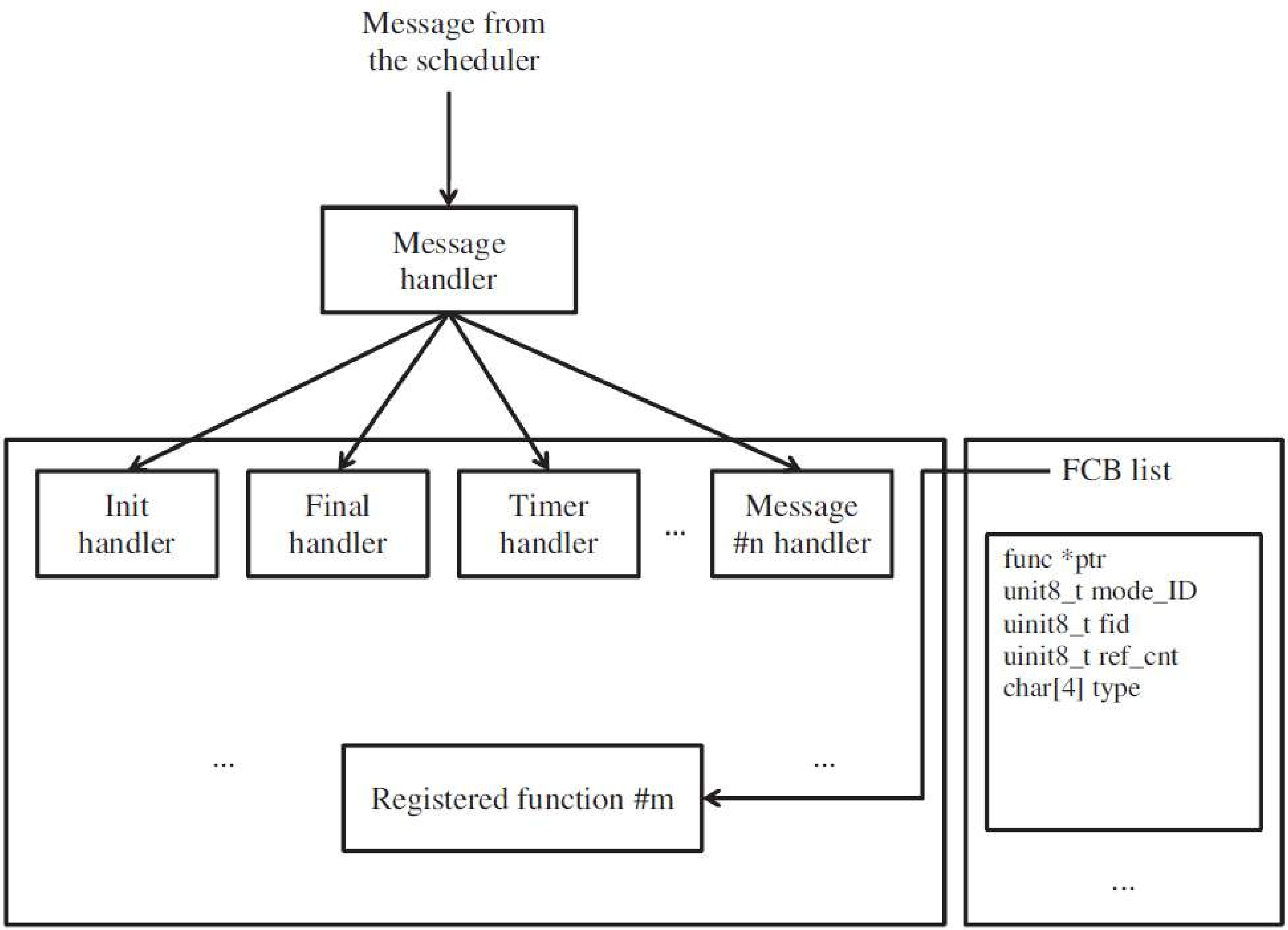
*2. **Direct Calls to registered functions*** **(synchronous communication)**

- requires modules to register their public functions at the kernel – all modules can subscribe to these functions
  - the kernel creates a *function control block (FCB)* to store key information about the function
  - this information is used to:
    - handle function subscription
    - support dynamic memory management
    - support runtime module update (replacement)

Operating System- Prototypes

SOS- Interaction

Interaction through a function call is faster than message-based communication.



Interaction between modules in SOS system (Han *et al*. 2005).

Operating System- Prototypes

SOS- Dynamic Reprogramming

Five basic features enable SOS to support dynamic reprogramming

- modules are *position independent binaries*
  - they use relative addresses rather than absolute addresses - they are re-locatable

- every SOS module implements two types of handlers – the *init* and *final* message handlers
  - the *init* message handler - to set the module's initial state
  - the *final* message handler - to release all resources the module owns and to enable the module to exit the system gracefully
  - after the *final* message, the kernel performs garbage collection

- SOS *uses a linker script* to place the *init* handler of a module at a known offset in the binary
  - enables easy linking during module insertion

- SOS keeps the state of a module *outside* of it
  - enables the newly inserted module to inherit the state information of the module it replaces

Operating System- Prototypes

SOS- Dynamic Reprogramming

- Whenever a module is inserted, SOS generates and *keeps* metadata that *contains information*:
  - the ID of the module
  - the absolute address of the *init* handler
  - a pointer to the dynamic memory holding the module state

- In SOS, dynamic module replacement (update) takes place in *three steps*:

1. a code distribution protocol *advertises* the new module in the network
2. the protocol *proceeds* with downloading the module and examines the metadata
   - the metadata contains the size of the memory required to *store the local state of the module*
   - if a node does not have sufficient *RAM,* module insertion is immediately *aborted*
3. if everything is correct, module insertion takes place and the kernel invokes the handler by *scheduling* an *init* message for the module

BASIC ARCHITECTURAL FRAMEWORK

MAC LAYER (Medium Access Control)

OUTLINES

- Overview
- Wireless MAC protocols
  - Carrier Sense Multiple Access
  - Multiple Access with Collision Avoidance (MACA) and MACAW
  - MACA By Invitation
  - IEEE 802.11
  - IEEE 802.15.4 and ZigBee
- Characteristics of MAC Protocols in Sensor Networks
  - Energy Efficiency
  - Scalability
  - Adaptability
  - Low Latency and Predictability
  - Reliability
- Contention-Free MAC Protocols
- Contention-Based MAC Protocols
- Hybrid MAC Protocols

MAC LAYER

- In most networks, multiple nodes share a communication medium for transmitting their data packets
- The medium access control (MAC) protocol is primarily responsible for regulating access to the shared medium
- The choice of MAC protocol has a direct bearing on the reliability and efficiency of network transmissions
  - due to errors and interferences in wireless communications and to other challenges
- Energy efficiency also affects the design of the MAC protocol
  - trade energy efficiency for increased latency or a reduction in throughput or fairness

MAC LAYER

FUNCTIONS

- It provides an abstraction of the physical layer to the LLC and upper layers of the OSI network.

- It performs multiple access resolutions when more than one data frame is to be transmitted. It determines the channel access methods for transmission.

- It also performs collision resolution and initiating retransmission in case of collisions.

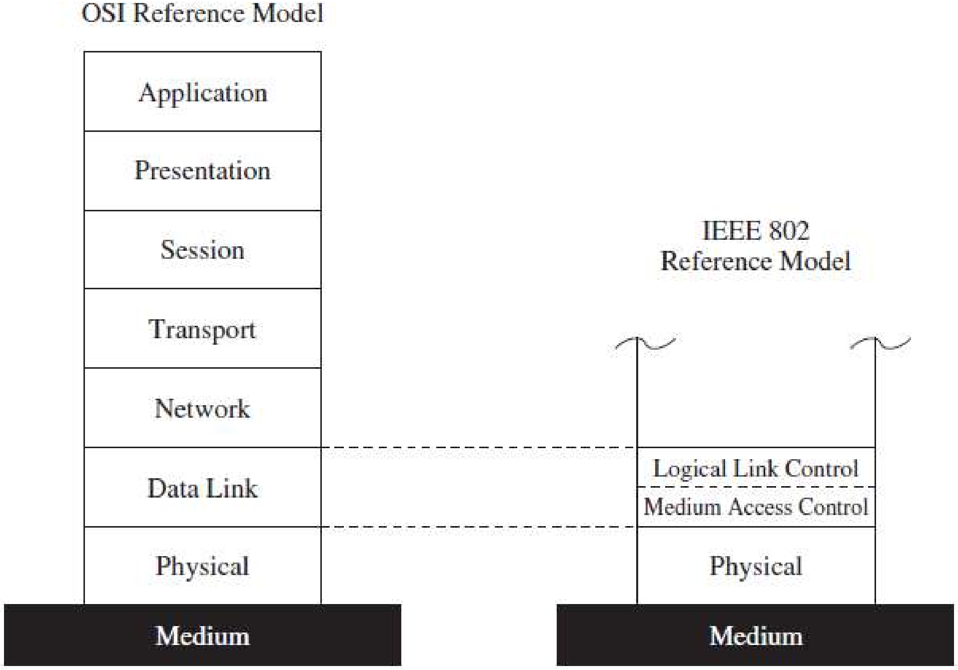
- **Framing:** It is responsible for encapsulating frames so that they are suitable for transmission via the physical medium.

- **Addressing:** It resolves the addressing of source station as well as the destination station, or groups of destination stations.

- **Flow Control:** It generates the frame check sequences and thus contributes to protection against transmission errors.

MAC LAYER

- *Responsibilities* of MAC layer include:
  - ❑ Decide when a node accesses a shared medium
  - ❑ Resolve any potential conflicts between competing nodes like Collision Resolution and Retransmission of frames in case of collision
  - ❑ Correct communication errors occurring at the physical layer
  - ❑ Perform other activities such as framing, addressing, and flow control
- Second layer of the OSI reference model (data link layer) or the IEEE 802 reference model (which divides data link layer into ***logical link control*** and ***medium access control layer***)



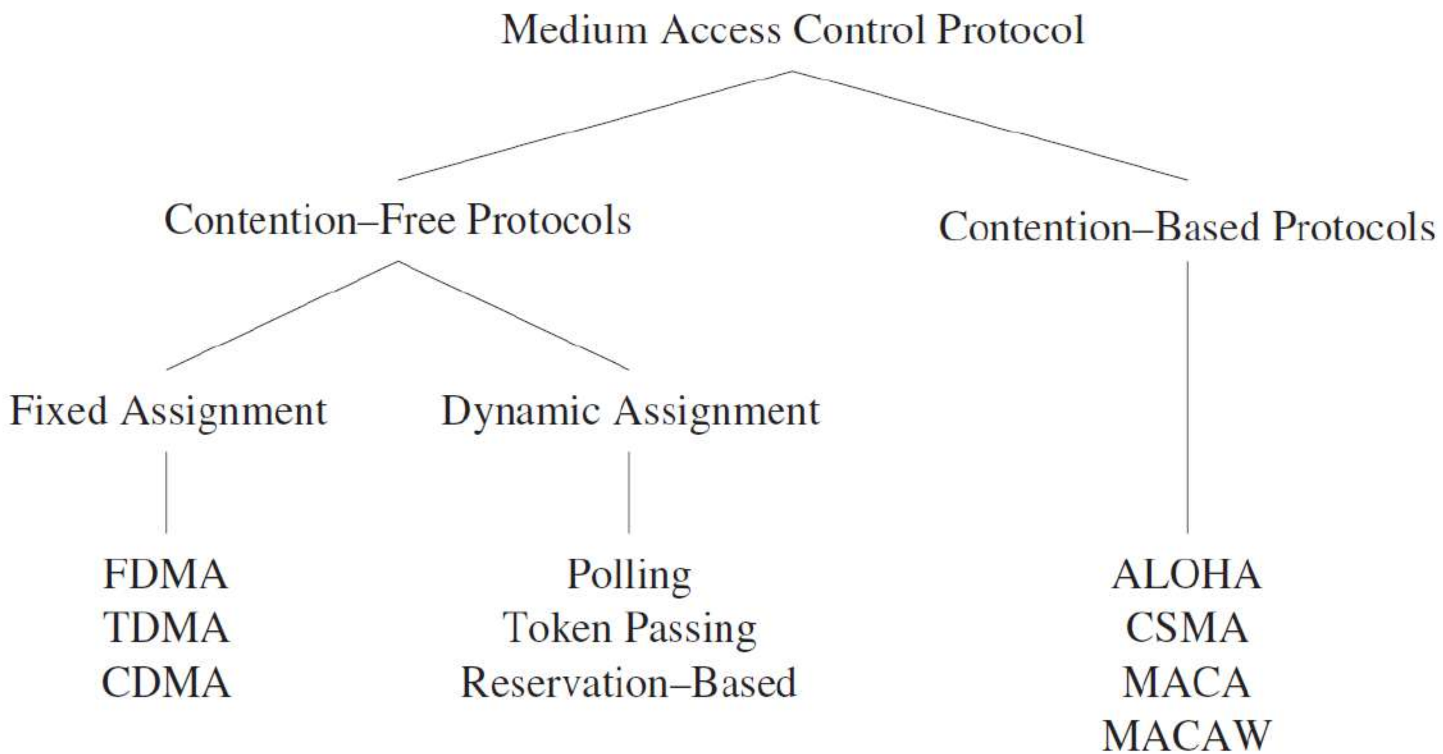
The MAC layer in the IEEE 802 reference model.

MAC LAYER

Most MAC protocols fall either into the categories of *contention-free* or *contention-based* protocols.

Contention-free Protocols:

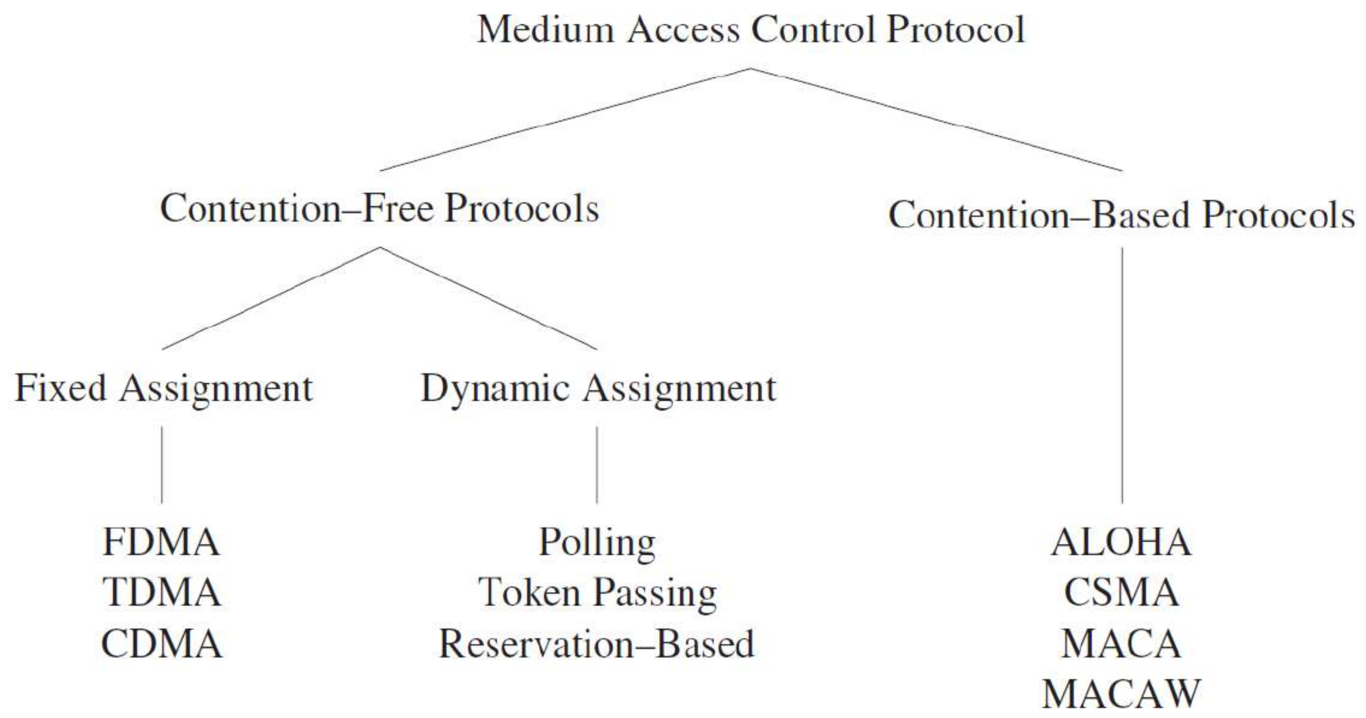
- In the first category, MAC protocols provide a medium sharing approach that ensures that only one device accesses the wireless medium at any given time.
- This category can further be divided into ***fixed*** and ***dynamic*** assignment classes, indicating whether the slot reservations are fixed or on demand.



MAC LAYER

Contention-based protocols allow nodes to access the medium simultaneously, but provide mechanisms to reduce the number of collisions and to recover from such collisions.

Finally, some MAC protocols do not easily fit into this classification since they share characteristics of both contention-free and contention-based techniques. These ***hybrid* approaches** often aim to inherit the advantages of these main categories, while minimizing their weaknesses.



MAC LAYER – CONTENTION-FREE MEDIUM ACCESS

- Collisions can be avoided by ensuring that each node can use its allocated resources exclusively

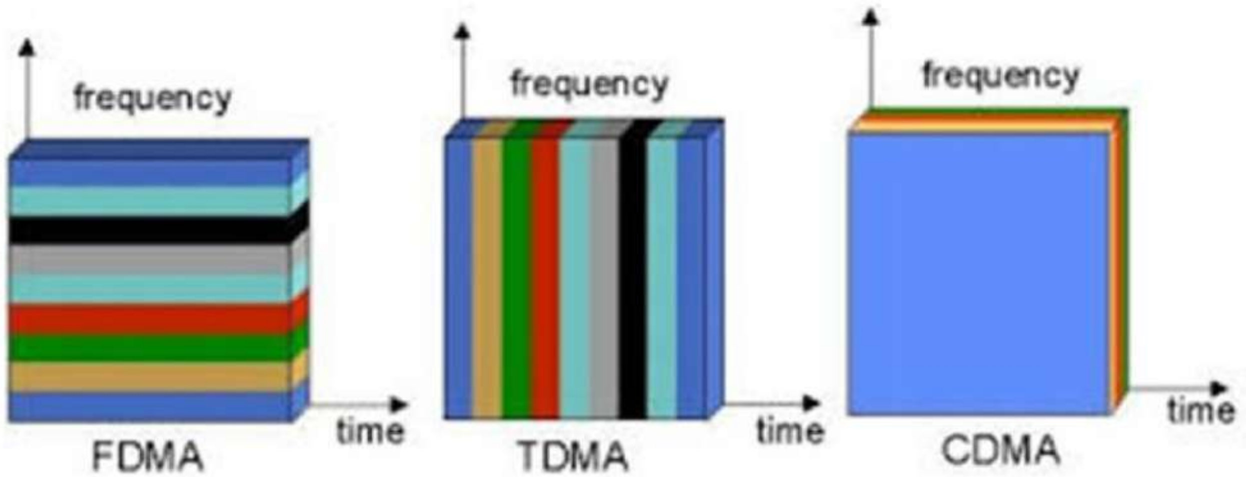
- Examples of fixed assignment strategies:

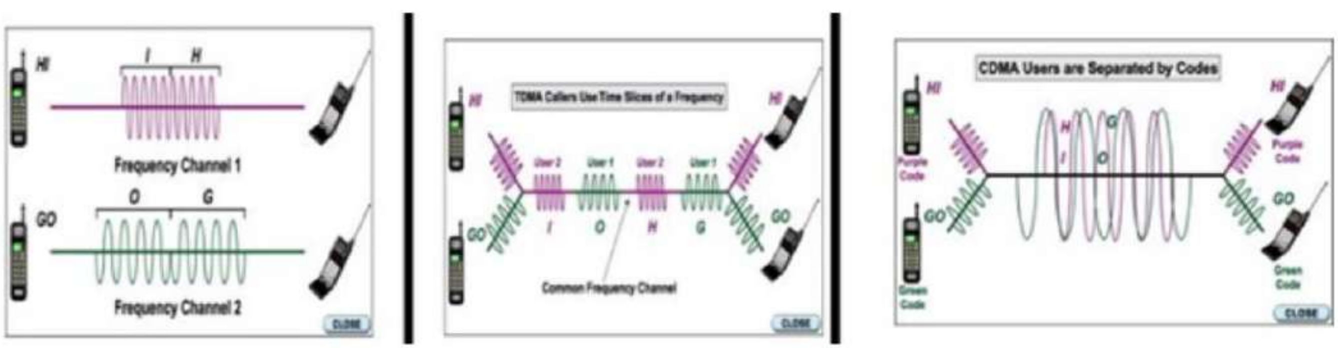
❑ **FDMA:** Frequency Division Multiple Access
  - the frequency band is divided into several smaller frequency bands
  - the data transfer between a pair of nodes uses one frequency band
  - all other nodes use a different frequency band

❑ **TDMA:** Time Division Multiple Access
  - multiple devices to use the same frequency band
  - relies on periodic time windows (frames)
    - frames consist of a fixed number of transmission slots to separate the medium accesses of different devices
    - a time schedule indicates which node may transmit data during a certain slot

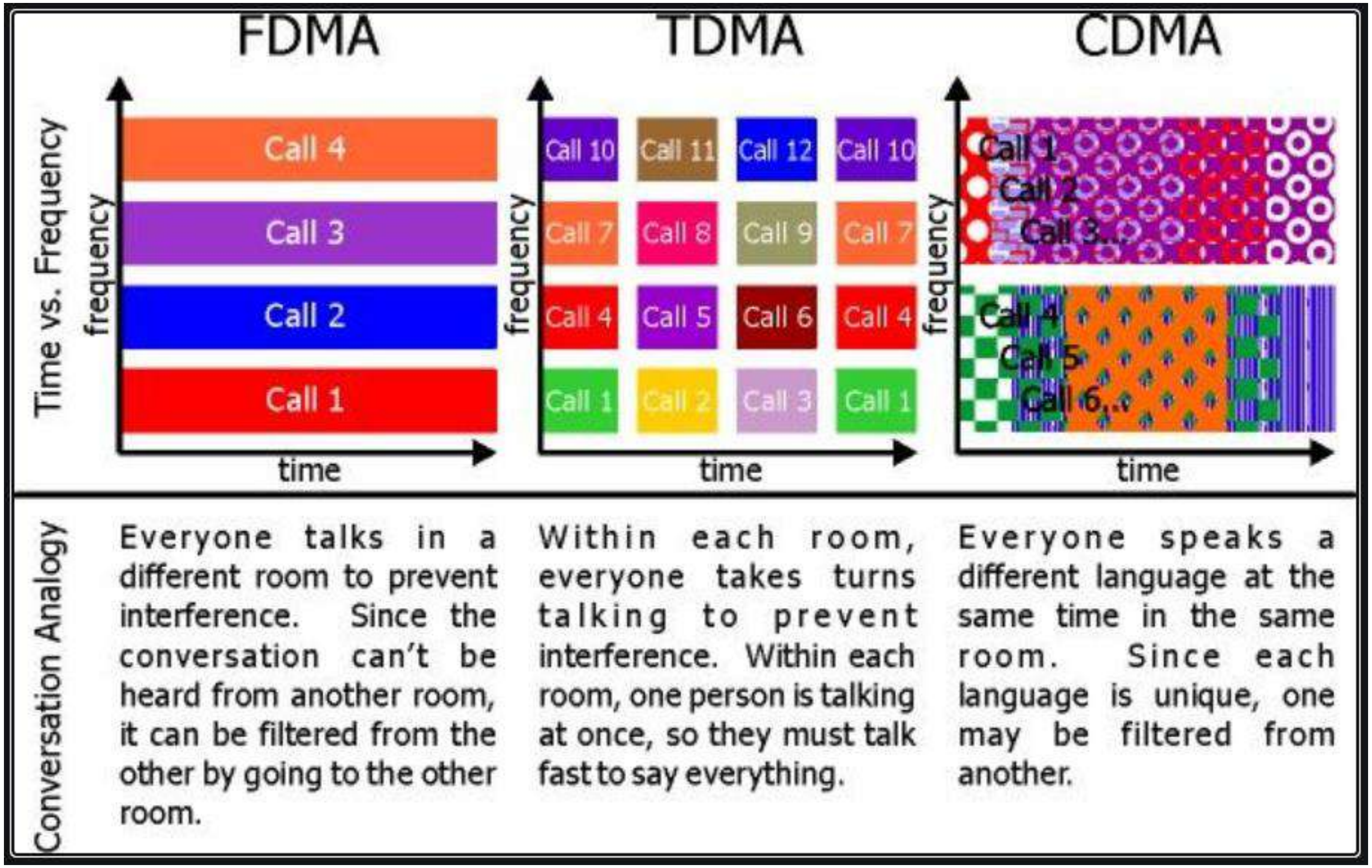
MAC LAYER – CONTENTION-FREE MEDIUM ACCESS

- **CDMA:** Code Division Multiple Access
  - simultaneous accesses of the wireless medium are supported using different codes
  - if these codes are orthogonal, it is possible for multiple communications to share the same frequency band
  - forward error correction (FEC) at the receiver is used to recover from interferences among these simultaneous communications





MAC LAYER – CONTENTION-FREE MEDIUM ACCESS



- Fixed assignment strategies are *inefficient*
  - it is impossible to reallocate slots belonging to one device to other devices if not needed in every frame
  - generating schedules for an entire network can be a taunting task
  - these schedules may require modifications every time the network topology or traffic characteristics in the network change

MAC LAYER – CONTENTION-FREE MEDIUM ACCESS

Dynamic assignment strategies: allow nodes to access the medium on demand:

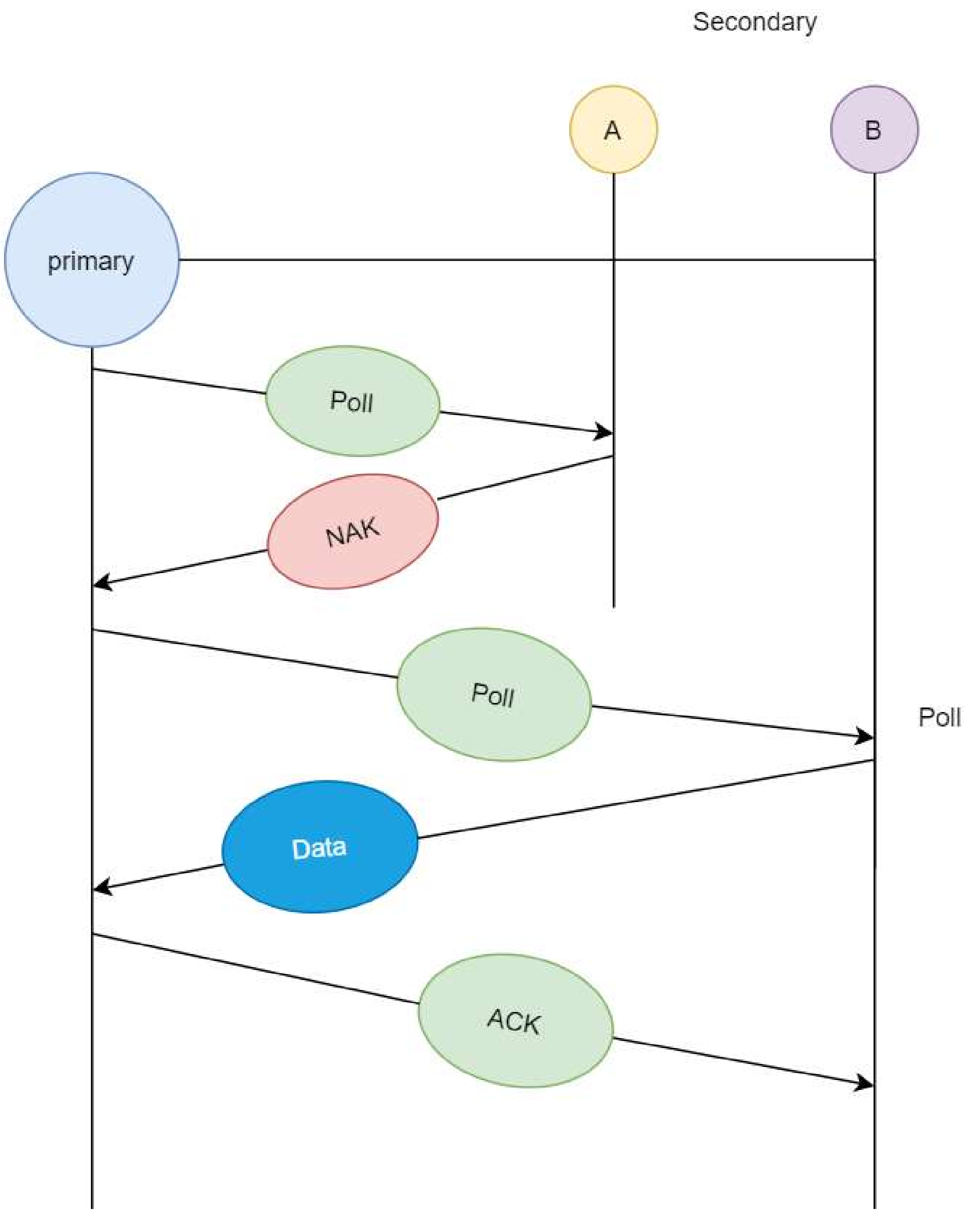
- **Polling-based Protocols**
  - In a computer network there is a primary station or controller (teacher) and all other stations are secondary (students), the primary station sends a message to each station. The message which is sent by the primary station consists of the address of the station which is selected for granting access in a round-robin fashion.

  - The point to remember is that all the nodes receive the message but the addressed one responds and sends data in return, but if the station has no data to transmit then it sends a message called **Poll Reject or NAK** (negative acknowledgment).

MAC LAYER – CONTENTION-FREE MEDIUM ACCESS

❑ Polling-based Protocols

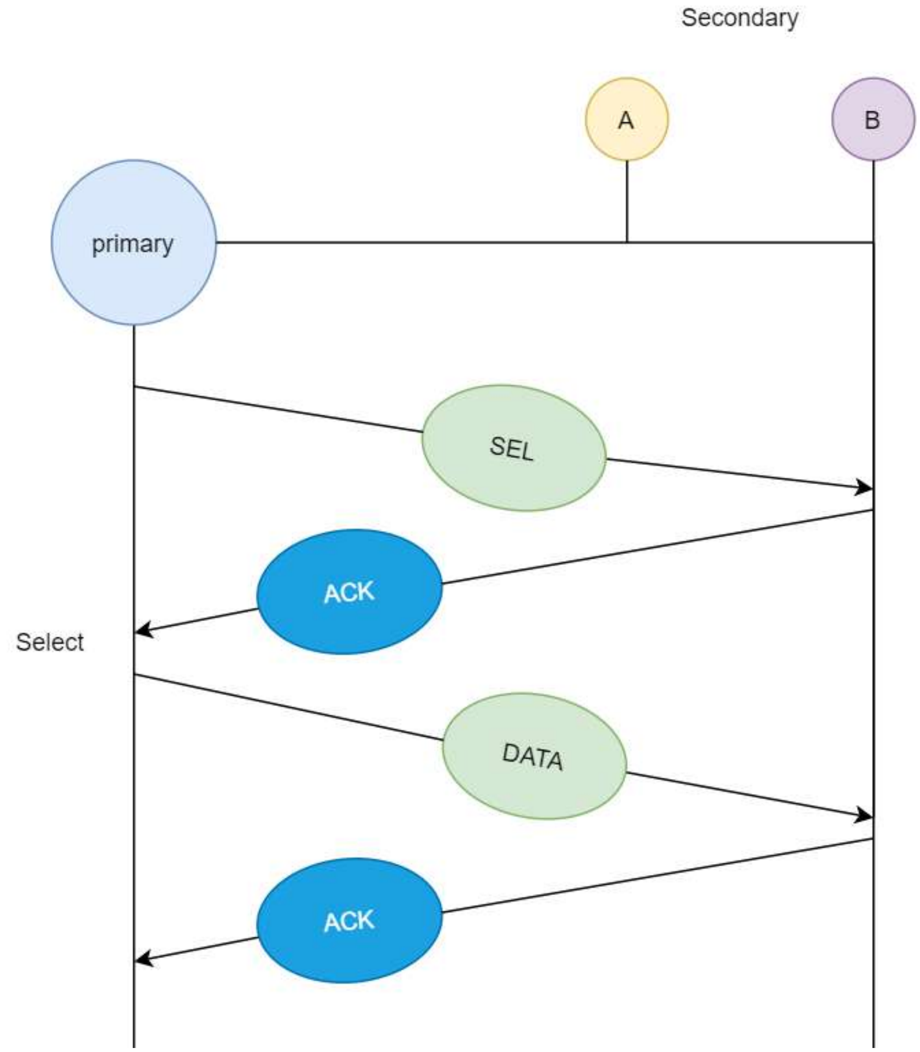
- Whenever the primary station wants to receive the data, it asks the secondary stations present in its channel, this method is **polling**. In this diagram, we see that primary station asks station A if it has any data ready for transmission, since A does not have any data queued for transmission it sends NAK (negative acknowledgement), and then it asks station B, since B has data ready for transmission, so it transmits the data and in return receives acknowledgement from primary station.



MAC LAYER – CONTENTION-FREE MEDIUM ACCESS

❑ Polling-based Protocols

- In the next case, if primary station wants to send data to the secondary stations, it sends a select message, and if the secondary station accepts the request from the primary station, then it sends back an acknowledgement and then primary station transmits the data and in return receives an acknowledgement.

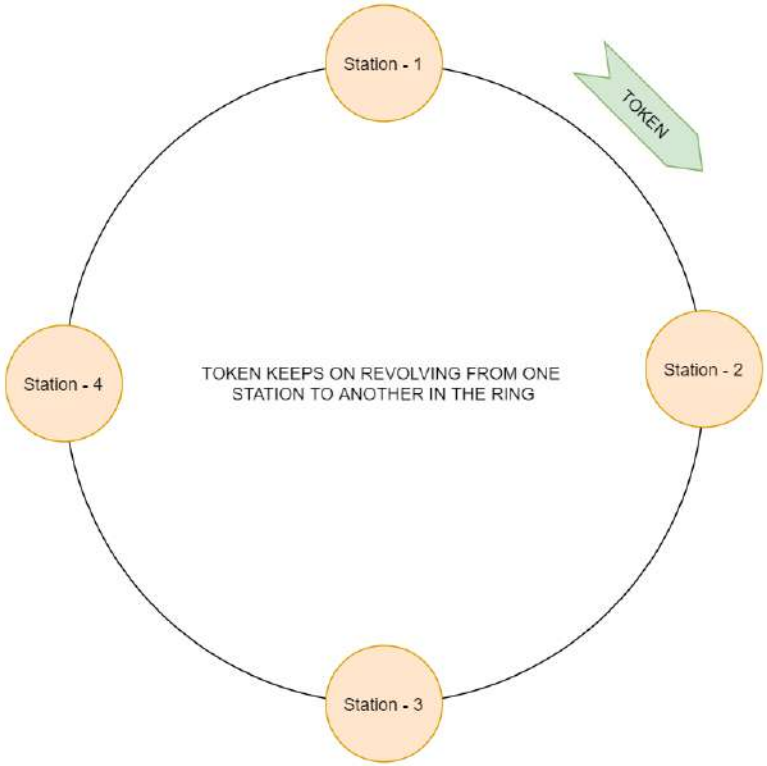


MAC LAYER – CONTENTION-FREE MEDIUM ACCESS

❑ Token Passing

- In computer networks a token is a special bit pattern that allows the token possessing system to send data or we can say that a token represents permission to transmit data.
- The token circulation around the table (or a network ring) is in a predefined order. A station can only pass the token to its adjacent station and not to any other station in the network.
- If a station has some data queued for transmission it can not transmit the data until it receives the token and makes sure it has transmitted all the data before passing on the received token.

In the diagram, when station-1 posses the token it starts transmitting all the data-frames which are in it's queue. now after transmission, station-1 passes the token to station-2 and so on. Station-1 can now transmit data again, only when all the stations in the network have transmitted their data and passed the token.



MAC LAYER – CONTENTION-FREE MEDIUM ACCESS

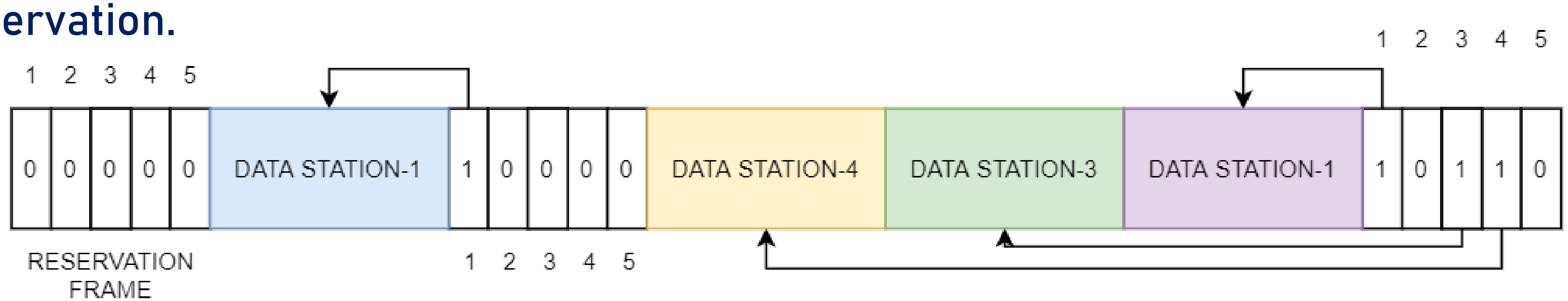
Reservation-based Protocols

- static time slots used to reserve future access to the medium
- e.g., a node can indicate its desire to transmit data by toggling a reservation bit in a fixed location
- these often very complex protocols then ensure that other potentially conflicting nodes take note of such a reservation to avoid collisions

Consider there are 4 stations then the reservation intervals are divided into 4 slots so that each station has a slot. Means if n number of stations are there then n slot will be allotted.

Similarly, if station-1 transmits a 1-bit data-frame in slot-1 then at that time no other station can transmit its data-frames and they must wait for their time slot. After all the slots have transmitted and checked then each station knows which station now wishes for transmission.

The illustration below shows a scenario with five stations with a five-slot reservation frame. here, in the time interval station 1,3,4 are the only stations with reservations and in the second interval station-1 is the only station with a reservation.



MAC LAYER – CONTENTION-BASED MEDIUM ACCESS

Contention-based Medium Access Protocol

- Contention is a media access method that is used to share the broadcast medium.
- Nodes may initiate transmissions at the same time or any time. It requires mechanisms to reduce the number of collisions and to recover from collisions.
- Node does not make any resource reservation a priori.
- Whenever it receives a packet to be transmitted, it contends with its neighbor for access to the shared channel.
- Nodes are not guaranteed periodic access to the channel.
- Examples are: ***Pure ALOHA, Slotted-ALOHA, CSMA, 802.11*** etc.
- The listen before talk operating procedure in 802.11 is the well known contention based protocol.

MAC LAYER – CONTENTION-BASED MEDIUM ACCESS

Pure ALOHA Protocol

- Packet Radio sends one packet per unit time.

- *Algorithm:* Node transmits packet whenever it requires. If Collision occurs, it waits for random time interval and retransmit the packet.

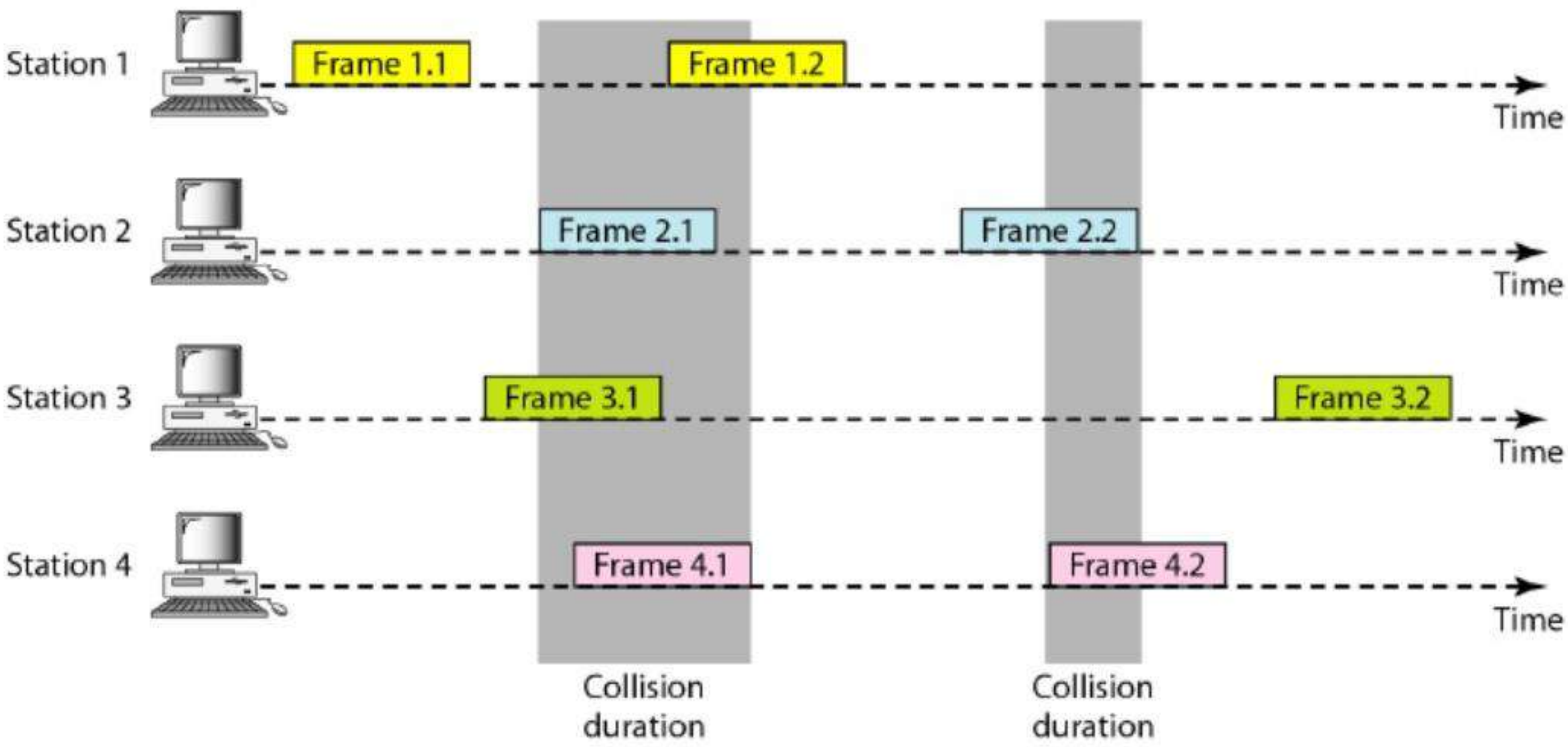
- *Topology*
  - All stations sends frame to central node, which broadcast packets to all stations.
  - Use of two distinct frequencies in a hub/star configuration.
  - The central station broadcast packets to everyone on the “Outbound” channel.
  - Various stations sends data packets to the central station on the “Inbound” channel.

- *Protocol:*
  - Whenever station has data, it transmits.
  - Sender finds out, whether the transmission was successful or not by listening to the broadcast from the central node.
  - If collision occurs, the sender retransmits frames/packets after some random time interval.

MAC LAYER – CONTENTION-BASED MEDIUM ACCESS

Pure ALOHA Protocol

- Very Simply to Apply
- Low utilization of channel
- Suitable for light loads only
- Chances of collisions are very high

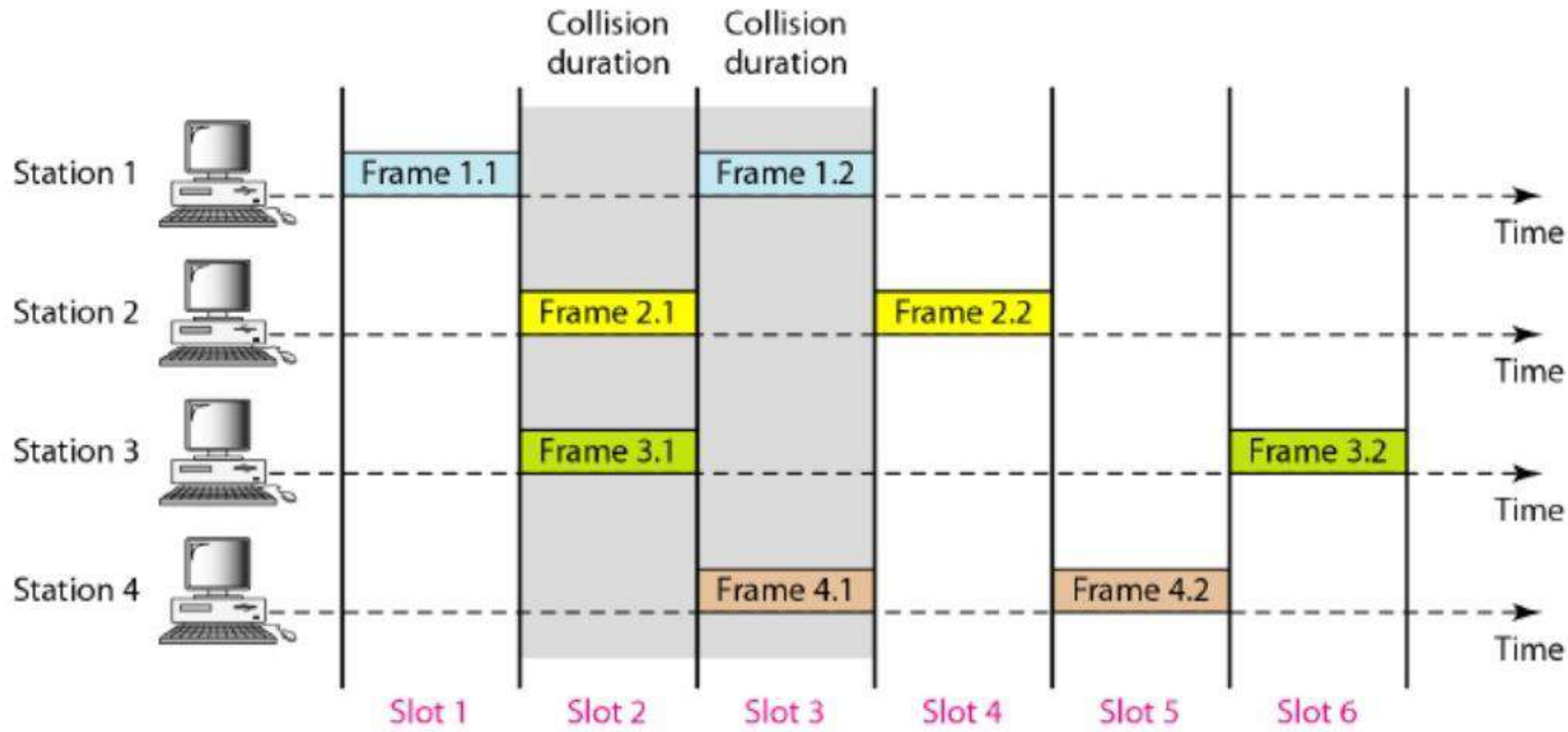


MAC LAYER – CONTENTION-BASED MEDIUM ACCESS

Slotted- LOHA Protocol

- Modified version of Pure ALOHA. It was invented to improve the efficiency of Pure ALOHA as chances of collisions in Pure ALOHA are very high.

- Slotted-ALOHA
  - It is a ALOHA with an additional constraint.
  - Time is divided into discrete time intervals (slots).
  - A station can only transmits at the beginning of the time frame.



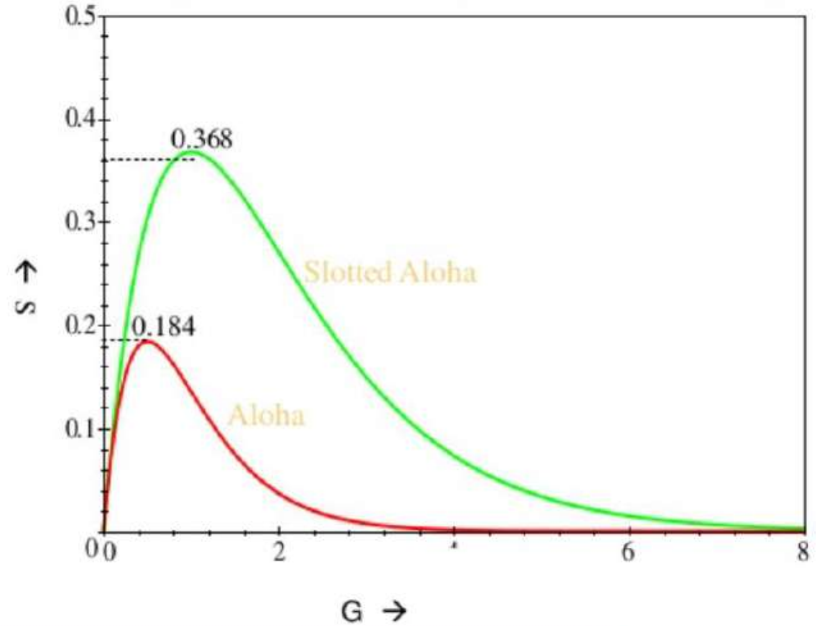
MAC LAYER – CONTENTION-BASED MEDIUM ACCESS

Slotted- LOHA Protocol

- Then, Slotted-ALOHA
  - Transmission synchronized to the start of the time frame.
  - Window of vulnerability is 1 time unit not 2.
  - It requires timing mechanism.
  - No partial collision.

- In Slotted ALOHA, there is still possibility of collision, if two stations try to sends at the beginning of the same time slot.

- Slotted-ALOHA is still has an edge over Pure ALOHA as chances of collisions are reduced to one-half.



MAC LAYER – CONTENTION-BASED MEDIUM ACCESS

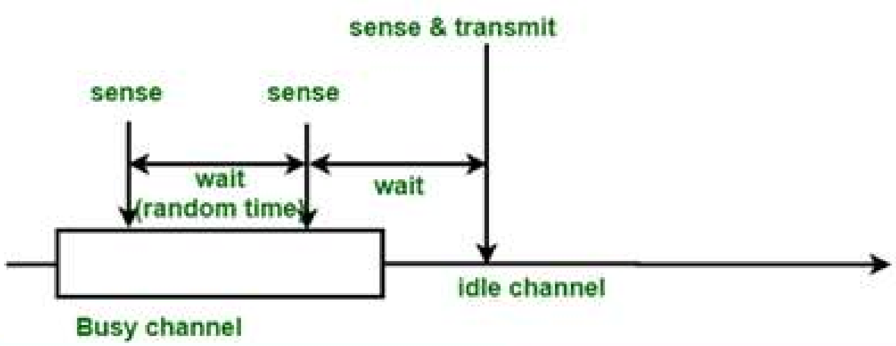
Carrier Sense Multiple Access (CSMA) - Non-persistent CSMA

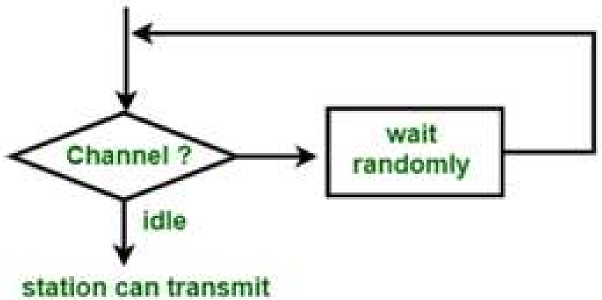
- When a node becomes ready to transmit a packet, it first senses the carrier to determine is another transmission in progress?
- To avoid collisions by listening to the carrier due to transmission from another user.
- If channel is idle or free, the node transmits its packet immediately and waits for the acknowledgement.
- In setting the acknowledgment timeout value, the node must take into account the round-trip propagation delay and the fact that the receiving node must also contend for the channel to transmit the acknowledgment.
- Estimating the average contention time required for a successful transmission is difficult, as it depends on the traffic load and the number of stations contending.
- In the absence of an acknowledgment, before a timeout occurs, the sending node assumes that the data packet is lost due to collision or noise interference.
- The station schedules the packet for retransmission.

MAC LAYER – CONTENTION-BASED MEDIUM ACCESS

Carrier Sense Multiple Access (CSMA) - Non-persistent CSMA

- If the channel is busy, the transmitting node "**backs off**" for a random period of time after which it senses the channel again.
- Non-Persistent CSMA protocol Depending on the status of the channel, the station transmits its packet if the channel is idle, or enters the back-off mode if the channel is busy.
- This process is repeated until the data packet is transmitted successfully.
- The non-persistent CSMA protocol minimizes the interference between packet transmissions, as it requires stations that find the channel busy to reschedule their transmissions randomly.
- ***Drawback:***
  - Channel may become idle during the back-off time of a contending station.
  - The unnecessary waste of channel capacity can reduce significantly the overall network throughput.

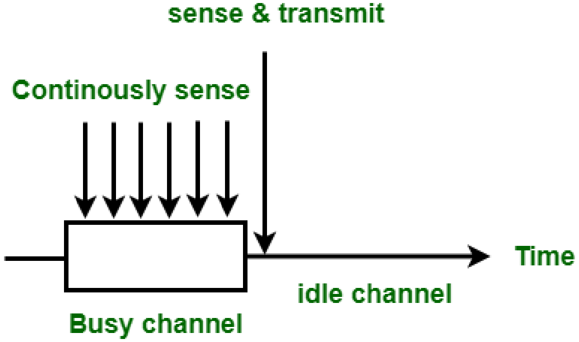


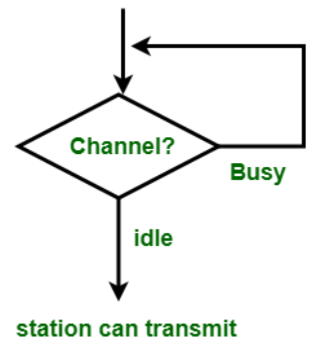


MAC LAYER – CONTENTION-BASED MEDIUM ACCESS

Carrier Sense Multiple Access (CSMA) - Persistent CSMA

- The need to address the shortcomings of non-persistent CSMA led to the development of a class of p-persistent CSMA schemes.
- These schemes differ in the algorithm they use to acquire a free channel.
- The 1-persistent scheme never allows the channel to remain idle if a node is ready to transmit.
- Based on this scheme,
  - A node ready to transmit a data packet first senses the channel.
  - If the channel is free, the node transmits its message without any delay. It transmits the frame with probability 1. Due to probability 1, it is called 1-persistent CSMA.
  - If the channel is busy, however, the node persistently continues to listen until the channel becomes idle.





MAC LAYER – CONTENTION-BASED MEDIUM ACCESS

Carrier Sense Multiple Access (CSMA) - Persistent CSMA

The problem with this method is that there are a large number of chances for the collision it is because there is a chance when two or more stations found channel in idle state and the transmit frames at the same time. On the time when collision occurs the station has to wait for the random time for the channel to be idle and to start all again.

MAC LAYER – CONTENTION-BASED MEDIUM ACCESS

Carrier Sense Multiple Access (CSMA) - Persistent CSMA

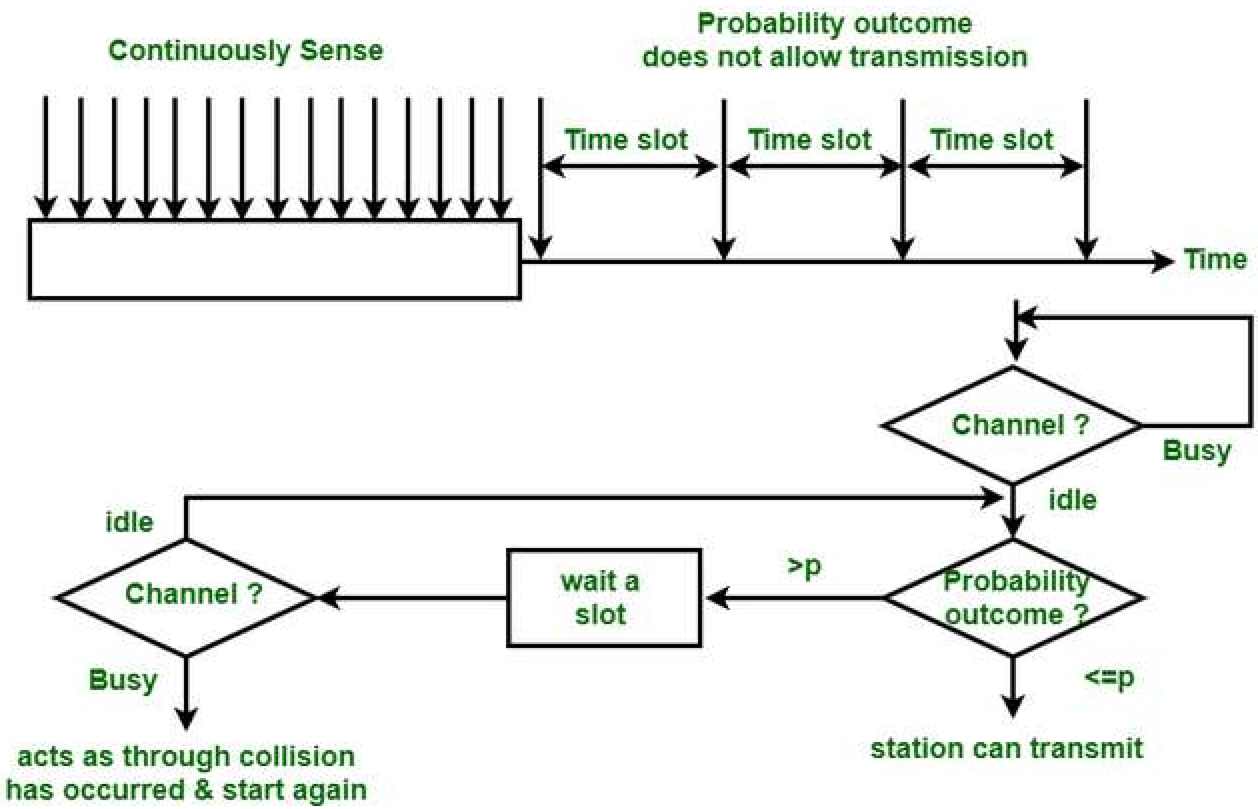
- The **p-persistent algorithm** represents a compromise between the non-persistent & 1-persistent schemes.

- Based on this algorithm,
    - This is the method that is used when channel has time-slots and that time-slot duration is equal to or greater than the maximum propagation delay time.
    - When the station is ready to send the frames, it will sense the channel.
    - If the channel found to be busy, the channel will wait for the next slot.
    - If the channel found to be idle, it transmits the frame with probability p, thus for the left probability i.e. q which is equal to 1-p the station will wait for the beginning of the next time slot.
    - In case, when the next slot is also found idle it will transmit or wait again with the probabilities p and q.
    - This process is repeated until either the frame gets transmitted or another station has started transmitting.

MAC LAYER – CONTENTION-BASED MEDIUM ACCESS

Carrier Sense Multiple Access (CSMA) - Persistent CSMA

P-persistent algorithm



MAC LAYER – CONTENTION-BASED MEDIUM ACCESS

Carrier Sense Multiple Access /Collision Detection (CSMA/CD)

CSMA:

- In networks where the propagation delay is small relative to the packet transmission time, the CSMA scheme and its variants can result in smaller average delays and higher throughput than with the ALOHA protocols.
- This performance improvement is due to the fact that carrier sensing reduces the number of collisions and, more important, the length of the collision interval.

Drawback CSMA:

- Contending stations continue transmitting their data packets even when collision occurs.
- For long data packets, the amount of wasted bandwidth is significant compared with the propagation time.
- Nodes may suffer unnecessarily long delays waiting for the transmission of the entire packet to complete before attempting to transmit the packet again.

To overcome the shortcomings:

- Networks use CSMA/CD, to extend the capabilities of communicating node to listen while transmitting.

MAC LAYER – CONTENTION-BASED MEDIUM ACCESS

Carrier Sense Multiple Access /Collision Detection (CSMA/CD)

CSMA/CD

This allows the node:

- To monitor the signal on the channel and
- To detect a collision when it occurs.

OPERATION :

- More specifically, if a node has data to send, it first listens to determine if there is an ongoing transmission over the communication channel.

- In the absence of any activity on the channel, the node starts transmitting its data and continues to monitor the signal on the channel while transmitting.

- If an interfering signal is detected over the channel, the transmitting station immediately aborts its transmission.

ADVANTAGE:

- This reduces the amount of bandwidth wasted due to collision to the time it takes to detect a collision.

MAC LAYER – CONTENTION-BASED MEDIUM ACCESS

Carrier Sense Multiple Access /Collision Detection (CSMA/CD)

STRATEGY & ALGORITHM:

- When a collision occurs, each contending station involved in the collision waits for a time period of random length before attempting to retransmit the packet.

- The length of time that a colliding node waits before it schedules packet retransmission is determined by a probabilistic algorithm, referred to as the truncated binary exponential back-off algorithm.

DRAWBACK:

- The need to provision sensor nodes with collision detection capabilities.

- As the Sensor nodes have a very limited amount of storage, processing power and energy resources, these limitations impose severe constraints on the design of the MAC layer.

- Another important factor that works against using a CSMA/CD based strategy to regulate access to a shared medium in a wireless environment is the difficulty of detecting collision in a wireless environment.

MAC LAYER – CONTENTION-BASED MEDIUM ACCESS

Carrier Sense Multiple Access /Collision Avoidance (CSMA/CA)

- Carrier Sense Multiple Access with Collision Avoidance (CSMA/CA) is a network protocol for carrier transmission that operates in the Medium Access Control (MAC) layer.

- In contrast to CSMA/CD (Carrier Sense Multiple Access/Collision Detection) that deals with collisions after their occurrence, CSMA/CA prevents collisions prior to their occurrence.

- Nodes sense the medium, but do not immediately access the channel when it is found idle.

- Instead, a node waits for a time period called DCF interframe space (DIFS) plus a multiple of a slot size.

- In case there are multiple nodes attempting to access the medium, the one with the shorter back-off period will win.

MAC LAYER – CONTENTION-BASED MEDIUM ACCESS

Carrier Sense Multiple Access /Collision Avoidance (CSMA/CA)

Algorithm

The algorithm of CSMA/CA is:

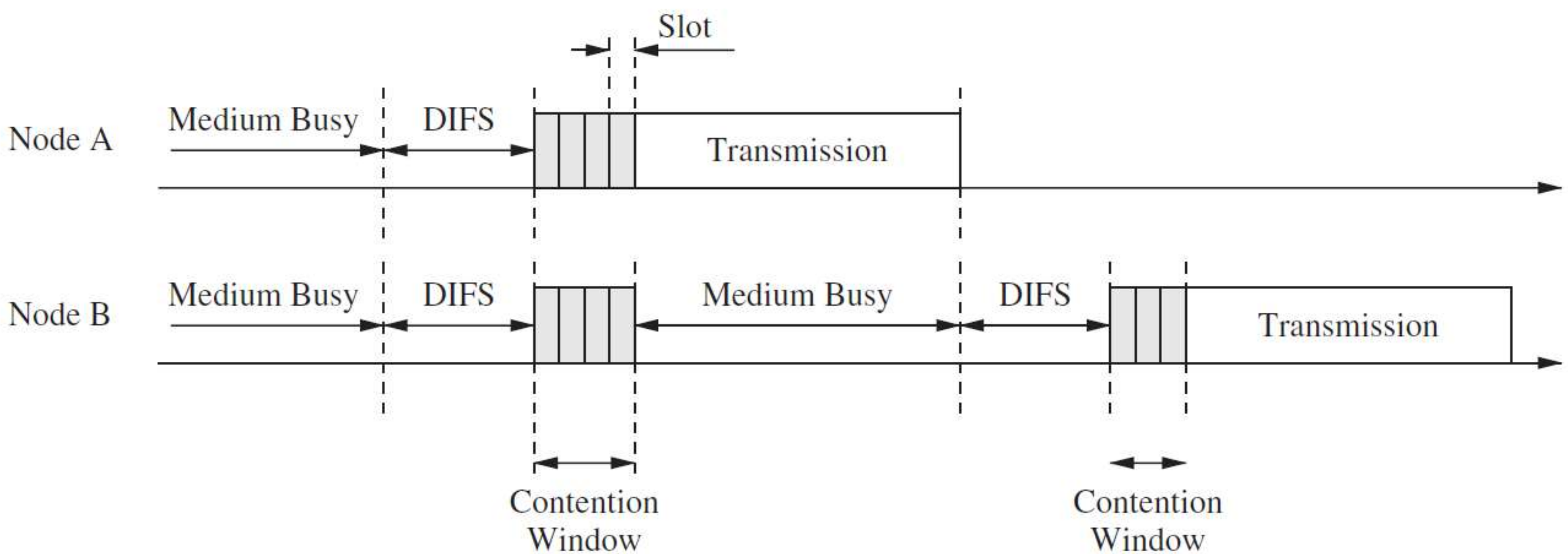
- When a frame is ready, the transmitting station checks whether the channel is idle or busy.
- If the channel is busy, the station waits until the channel becomes idle.
- If the channel is idle, the station waits for an Inter-frame gap (IFG) amount of time and then sends the frame.
- After sending the frame, it sets a timer.
- The station then waits for acknowledgement from the receiver. If it receives the acknowledgement before expiry of timer, it marks a successful transmission.
- Otherwise, it waits for a back-off time period and restarts the algorithm.

MAC LAYER – CONTENTION-BASED MEDIUM ACCESS

Carrier Sense Multiple Access /Collision Avoidance (CSMA/CA)

Example:

- Node A waits for DIFS + 4 $*$ s (where s represents the slot size), while node B's back-off is DIFS + 7 $*$ s.
- Once node A begins with its transmission, node B freezes its own backoff timer and resumes the timer after node A completes its transmission plus another period of DIFS.
- Once node B's back-off timer expires, it can also begin its transmission.



Medium access using CSMA/CA

MAC LAYER – CONTENTION-BASED MEDIUM ACCESS

Carrier Sense Multiple Access /Collision Avoidance (CSMA/CA)

- Carrier sensing prior to transmission is an effective approach to increase the throughput efficiency in shared-medium access environments.

- In wireless environments, the scheme is susceptible to two problems, commonly referred to as the ***hidden-node*** and ***exposed-node*** problems.

- The hidden and exposed-node problems result indirectly from the time-varying properties of the wireless channel, which are caused by physical phenomena such as noise, fading, attenuation and path loss.

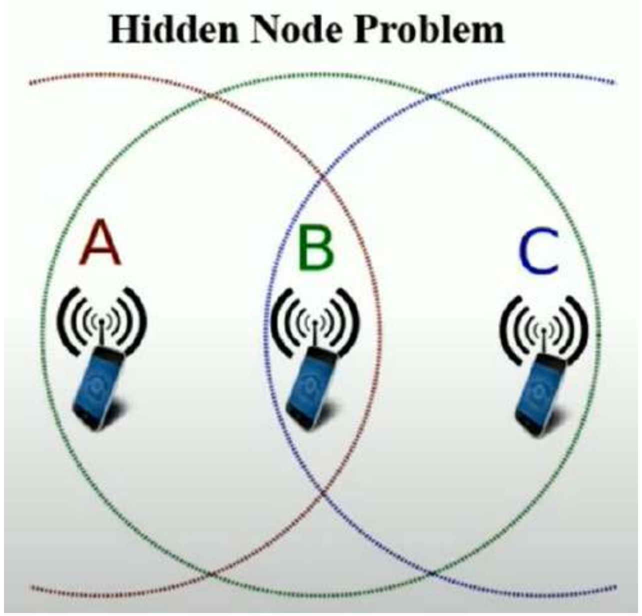
- These interferences, combined with the rapid decrease in the power received with the distance between the sender and receiver, limit the maximum transmission range that can be achieved by a sending node.

MAC LAYER – CONTENTION-BASED MEDIUM ACCESS

Carrier Sense Multiple Access /Collision Avoidance (CSMA/CA)

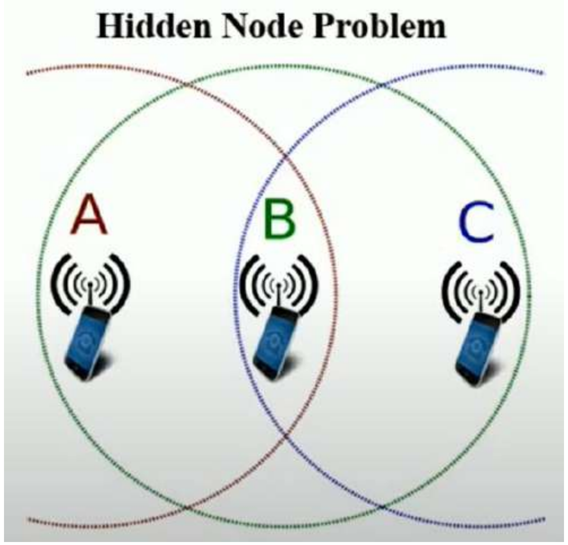
HIDDEN NODE PROBLEM

- A hidden node is defined as a node that is within the range of the destination node but out of range of the transmitting node.
- To illustrate this example, consider figure where node B is within the transmission range of nodes A and C.
- Furthermore, assume that nodes A and C are outside their mutual transmission ranges.
- Hidden node Consequently, any transmission from either of the two nodes will not reach the other node. { A $\leftrightarrow$ C}
- Given this network configuration, assume that node A needs to transmit a data packet to node B. {A $\rightarrow$ B}
- According to the CSMA protocol, node A senses the channel and determines that it is free.



MAC LAYER – CONTENTION-BASED MEDIUM ACCESS

Carrier Sense Multiple Access /Collision Avoidance (CSMA/CA)



- Node A then proceeds to transmit its packet.
- Assume now that before node A completes its transmission to node B, node C decides to transmit a data packet to node B.
- Hidden node Using the CSMA protocol, node C senses the channel and also determines that the channel is free, since node C, which is outside the transmission range of node A, cannot hear the signal transmitted by node A.
- As a result, both transmissions collide at node B, thereby causing the loss of both data packets.
- Notice that neither node A nor node C is aware of the collision, since it happens at the receiver.
- This feature is intrinsic to wireless networks and constitutes a fundamental difference in the way that collisions are dealt with in wired and wireless environments.

MAC LAYER – CONTENTION-BASED MEDIUM ACCESS

Carrier Sense Multiple Access /Collision Avoidance (CSMA/CA)

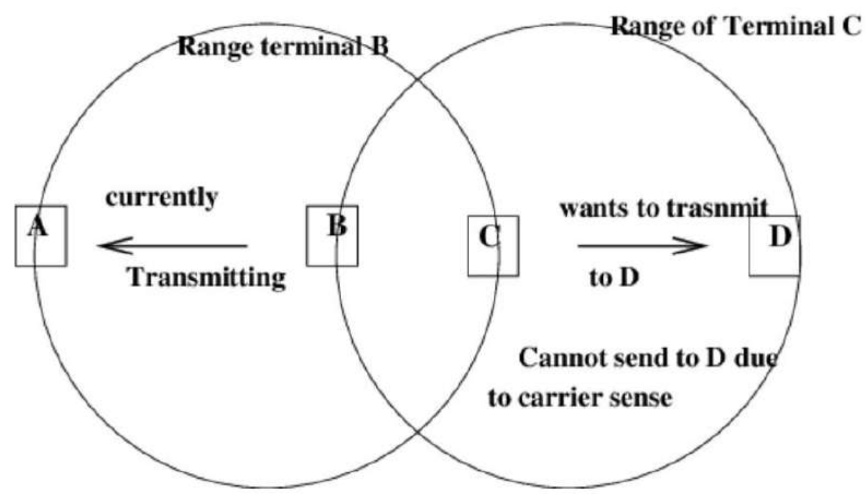
EXPOSED NODE PROBLEM

- An exposed node is a node that is within the range of the sender but out of the range of the destination.

- To illustrate problem, the consider exposed-node the network depicted in Figure, where node B is within the transmission range of nodes A and C, nodes A and C are outside their mutual transmission ranges, and node D is within the transmission range of node C.

- Assume that node B wants to transmit a message to node A.

- Node B executes the CSMA protocol to sense the channel, determines that the channel is free, and proceeds to transmit the data packet to node A.

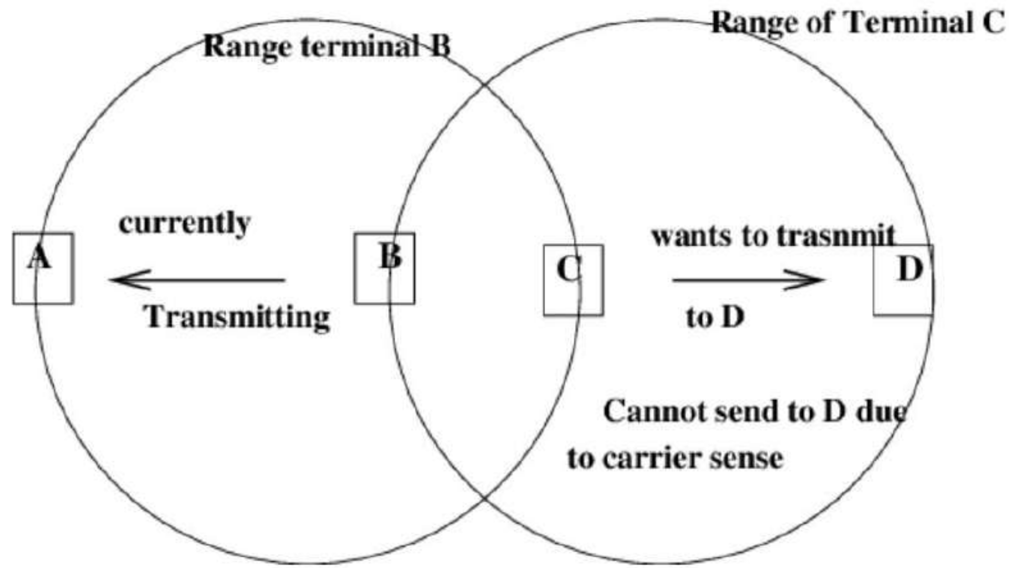


MAC LAYER – CONTENTION-BASED MEDIUM ACCESS

Carrier Sense Multiple Access /Collision Avoidance (CSMA/CA)

EXPOSED NODE PROBLEM

- Assume now that node C needs to send a packet to D.
- Node C follows the CSMA rule and first senses the channel.
- Due to the ongoing transmission between nodes B and A, node C determines that the channel is busy and delays the transmission of its packet to a later time.
- It is clear, however, that this delay is unnecessary, since the transmission from node C to node D would have been completed successfully, as node D is outside the range of node B.



MAC LAYER – CONTENTION-BASED MEDIUM ACCESS

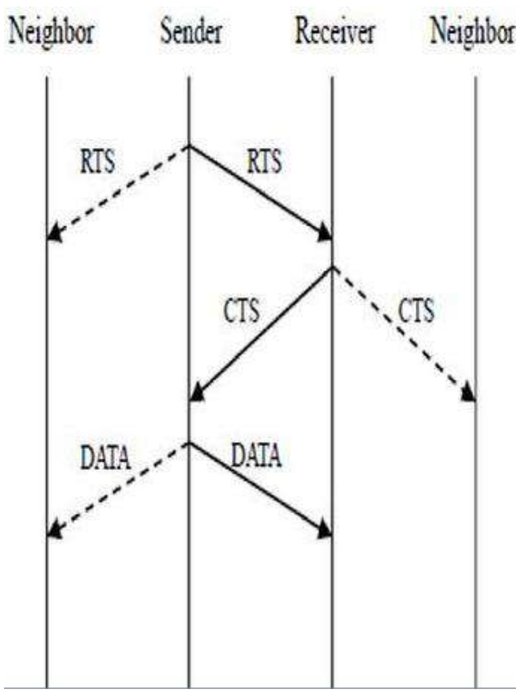
Multiple Access Collision Avoidance (MACA)

MACA

- It is a medium access control (MAC) layer protocol used in wireless networks, with a view to solve the hidden terminal problem and exposed terminal problem.
- The MAC layer protocol IEEE 802.11 RTS/CTS has been adopted from MACA.

Working Principle

- The MACA protocol works with the condition that the stations are synchronized and frame sizes and data speed are the same. It involves transmission of two frames called RTS and CTS prior to data transmission.
- RTS stands for Request to Send and CTS stands for Clear to Send.
- Once the sender receives the CTS packet without any error, it starts transmitting the data packet.
- If a packet transmitted by a node is lost, the node uses the binary exponential back-off (BEB) algorithm to back off a random interval of time before retrying. In this each time a collision occurs the node doubles its maximum back-off windows.



MAC LAYER – CONTENTION-BASED MEDIUM ACCESS

Multiple Access Collision Avoidance (MACA)

- The binary exponential back-off mechanism used in MACA might starves flows sometimes.

- The problem is solved by MACAW.

Let us consider that a transmitting station A has data frame to send to a receiving station B. The operation works as follows:

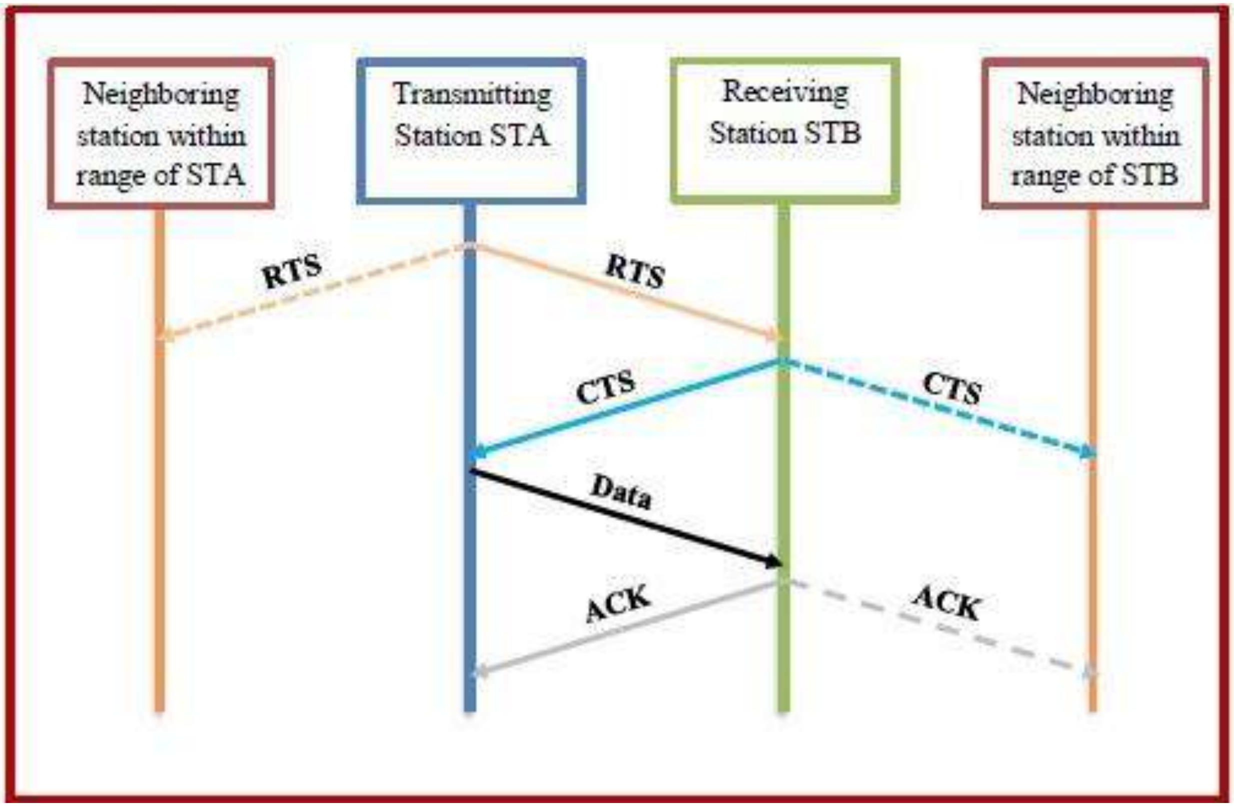
- Station A sends a RTS frame to the receiving station.
- On receiving the RTS, station B replies by sending a CTS frame.
- On receipt of CTS frame, station A begins transmitting its data frame.
- After successful receipt of the data frame, station B sends an ACK frame (acknowledgement frame).

MAC LAYER – CONTENTION-BASED MEDIUM ACCESS

Multiple Access /Collision Avoidance (MACA)

- Any station than can hear RTS is close to the transmitting station and remains silent long enough for the CTS, or waits for a certain time period. If the RTS is not followed by a CTS, the maximum waiting time is the RTS propagation time.

- Any station that can hear the CTS is close to the receiving station and remains silent during the data transmission. It attempts for transmission after hearing the ACK.

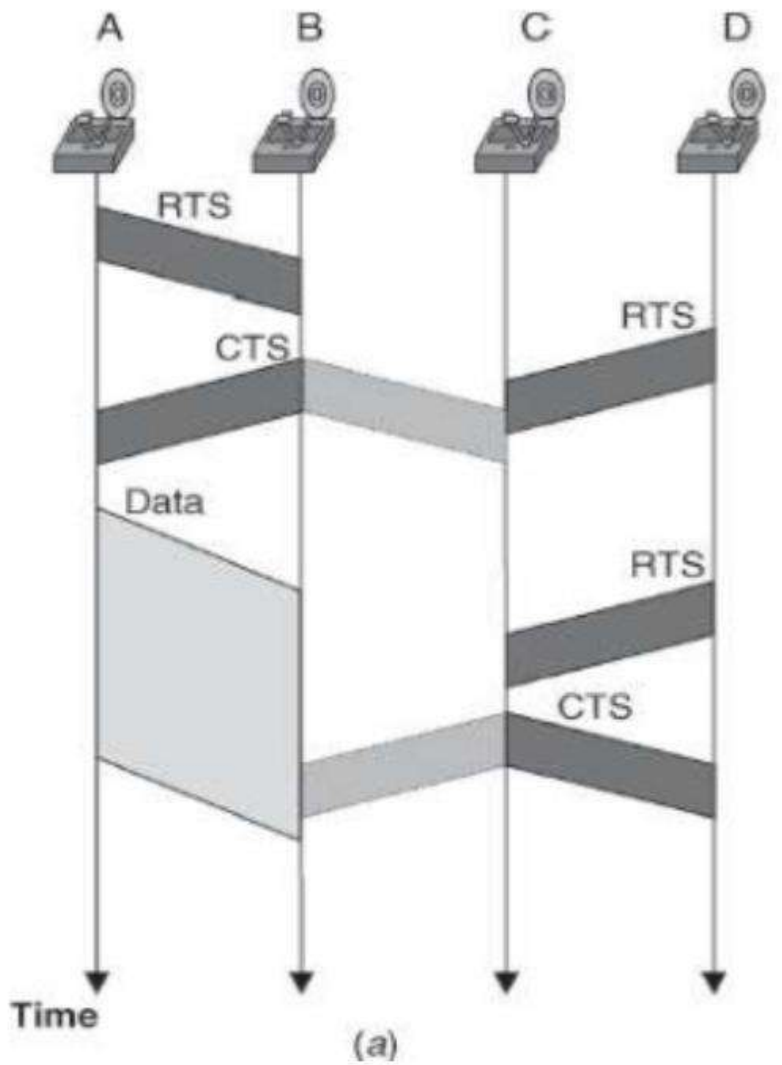


MAC LAYER – CONTENTION-BASED MEDIUM ACCESS

Multiple Access Collision Avoidance (MACA)

RTS & CTS handshake Approach: limitation Case-I

- Node A senses the channel to be free and sends an RTS packet to node B. In reply, node B sends a CTS packet.
- Node C, which is in the transmission range of node B, starts receiving the CTS packet.
- Before the reception of this packet is complete, however, node D, which is in the transmission range of node C, sends a RTS packet. The latter packet collides with the CTS packet sent by node B.
- Meanwhile, node A, which receives the CTS packet correctly, proceeds to transmit its data packet to node B

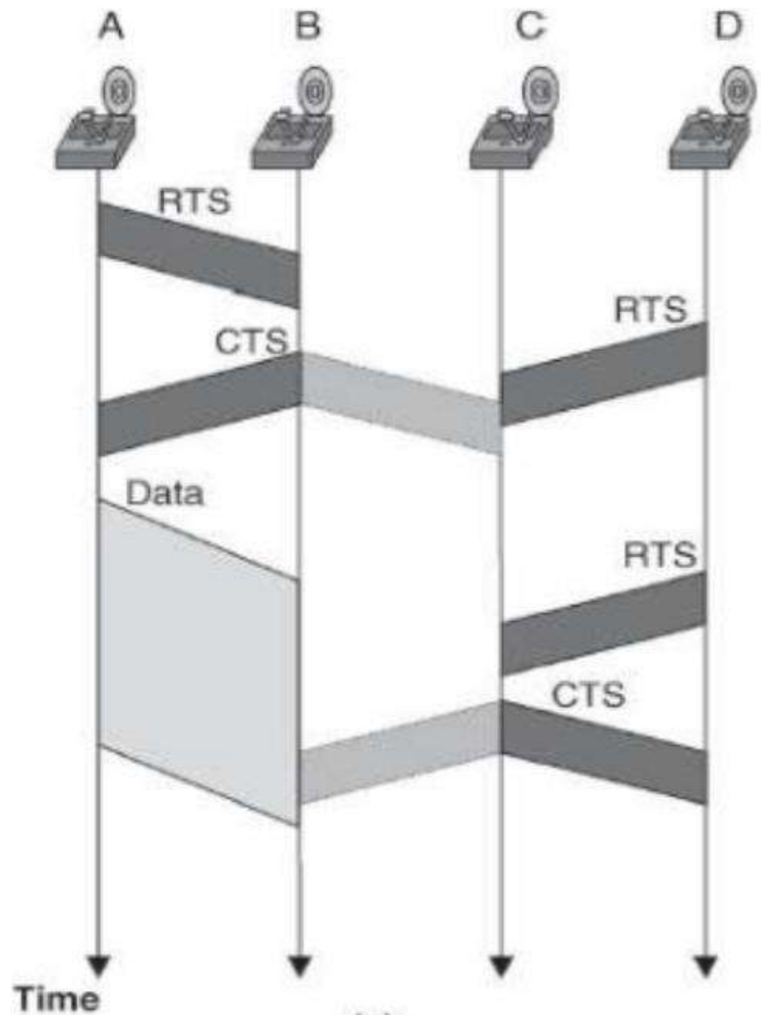


MAC LAYER – CONTENTION-BASED MEDIUM ACCESS

Multiple Access Collision Avoidance (MACA)

RTS & CTS handshake Approach: limitation Case-I

- Node D later times-out and retransmits its RTS packet.
- Since node C never received node B's CTS packet, it assumes that the channel is free and replies with a CTS packet to node D.
- Since node B is within the transmission range of node C, the latter packet collides with the data packet being transmitted by node A.



(a)

MAC LAYER – CONTENTION-BASED MEDIUM ACCESS

Multiple Access /Collision Avoidance (MACA)

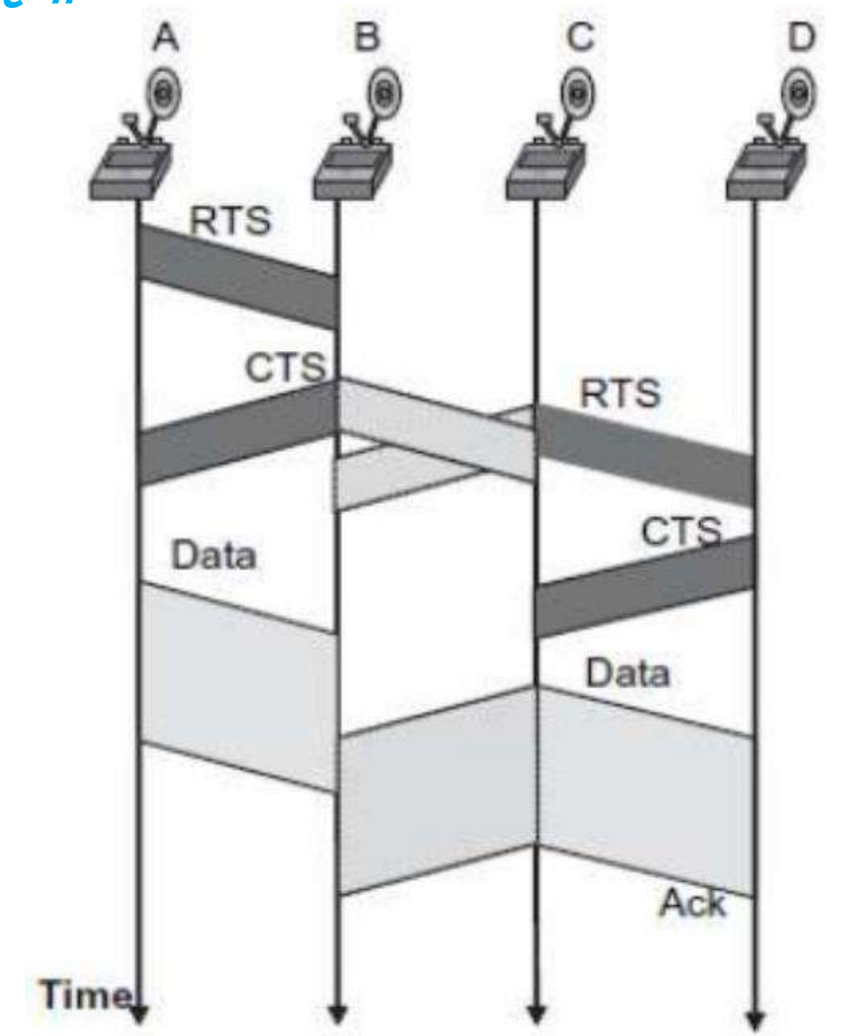
RTS & CTS handshake Approach: limitation Case-II

- Node A senses the channel to be free and sends an RTS packet to node B. In reply, node B sends a CTS packet to node A.

- The CTS packet is received correctly by node A, which allows it to transmit its packet.

- The CTS packet is also received by node C, which is within the transmission range of node B.

- Since node C has started transmitting an RTS packet to node D, nearly at the same time that node B is transmitting its CTS packet; node C does not receive correctly the CTS packet sent by B.



(b)

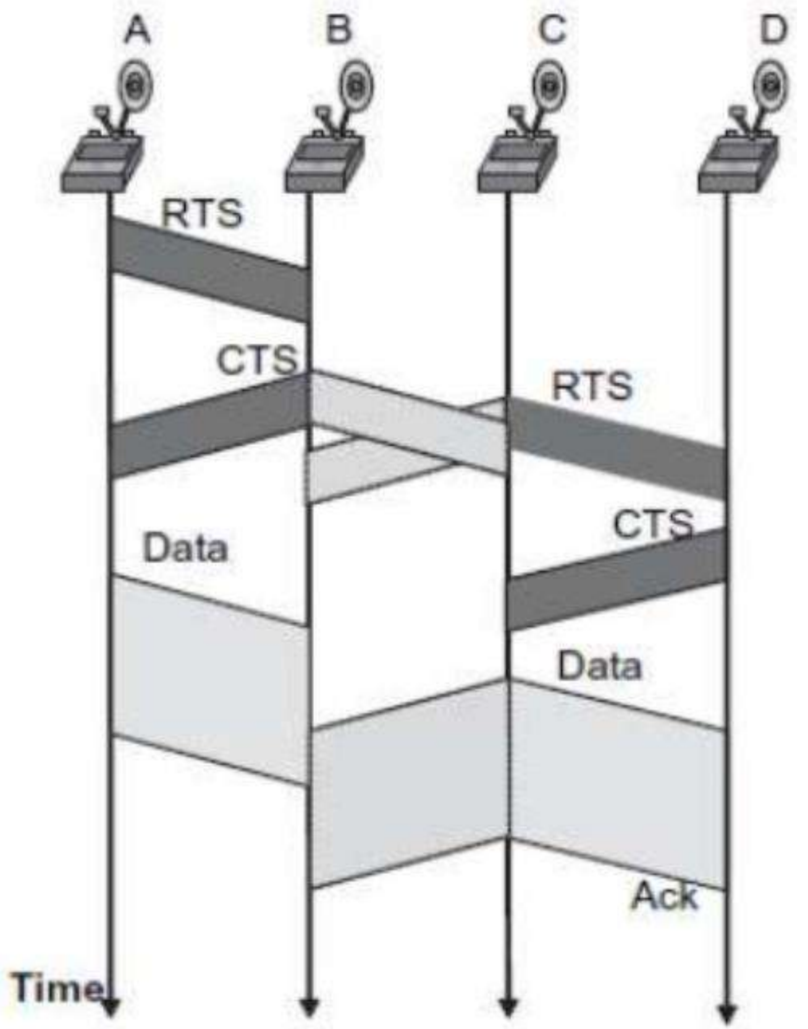
MAC LAYER – CONTENTION-BASED MEDIUM ACCESS

Multiple Access Collision Avoidance (MACA)

RTS & CTS handshake Approach: limitation Case-II

- Node D, however, receives correctly the RTS packet sent by node C. In response, it sends a CTS packet to node C, thereby allowing it to start transmitting its data packet.

- Since node A did not complete transmission of its data packet to node B, node C's data transmission causes a collision at node B.



(b)

MAC LAYER – CONTENTION-BASED MEDIUM ACCESS

Multiple Access Collision Avoidance (MACA)

- There is no carrier sensing in MACA so Collision occurs during the RTS-CTS phase

- Each mobile host adds a random amount of time to the minimum interval required to wait after overhearing an RTS or CTS control message.

- In MACA, the slot time is the duration of an RTS packet.

- If two or more stations transmit an RTS concurrently, resulting in a collision, these stations will wait for a randomly chosen interval and try again, doubling the average interval on every attempt.

- The station that wins the competition will receive a CTS from its responder, thereby blocking other stations to allow the data communication session to proceed.

- Compared to CSMA, MACA reduces the chances of data packet collisions. Since control messages (RTS and CTS) are much smaller in size compared to data packets, the chances of collision are also smaller

MAC LAYER – CONTENTION-BASED MEDIUM ACCESS

Multiple Access Collision Avoidance With Wireless (MACAW)

MACAW (MACA with Wireless)

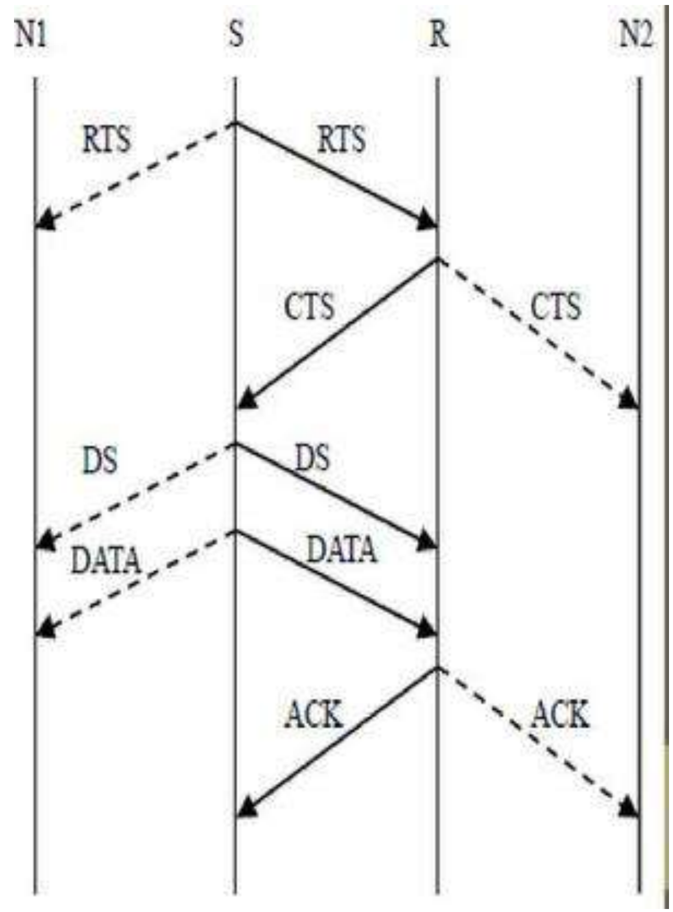
- The binary exponential back-off mechanism used in MACA might starves flows sometimes.
- The problem is solved by MACAW.
- The packet header has current back-off counter value of transmitting node.
- It implements per flow fairness as opposed to the per node fairness in MACA.
- MACAW is proposed as a series of improvements to the basic MACA algorithm.
  1. Suggest a less aggressive backoff algorithm:
     - a multiplicative increase and linear decrease(MILD) backoff mechanism is used.
     - Increasing BO by 1.5 after a timeout, and decreasing it by 1 after a successful RTS-CTS pair.
  2. proposes that receivers should send an ACK to the sender after successfully receiving a data message.
  3. Propose two related techniques for allowing transmitters to avoid contention more effectively:
     - Data sending (DS)
     - Request-for-request-to-send (RRTS)

MAC LAYER – CONTENTION-BASED MEDIUM ACCESS

Multiple Access Collision Avoidance With Wireless (MACAW)

MACAW (MACA with Wireless)

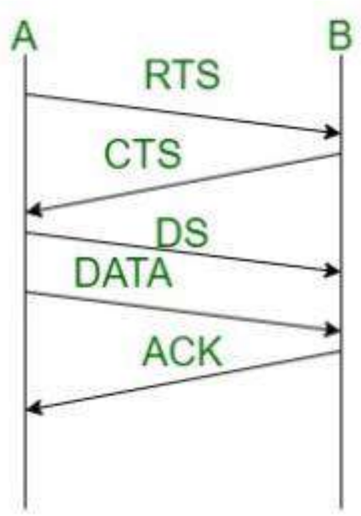
- It is a revision of MACA. The sender transmits a RTS (Request To Send) frame if no nearby station transmits a RTS.
- The receiver replies with a CTS (Clear To Send) frame.
- Neighbors
  - see CTS, then keep quiet.
  - see RTS but not CTS, then keep quiet until the CTS is back to the sender.
- The receiver sends an ACK when receiving a frame.
  - Neighbors keep silent until see ACK.
- Collisions
  - There is no collision detection.
  - The senders know collision when they don't receive CTS.
  - They each wait for the exponential back-off time.



MAC LAYER – CONTENTION-BASED MEDIUM ACCESS

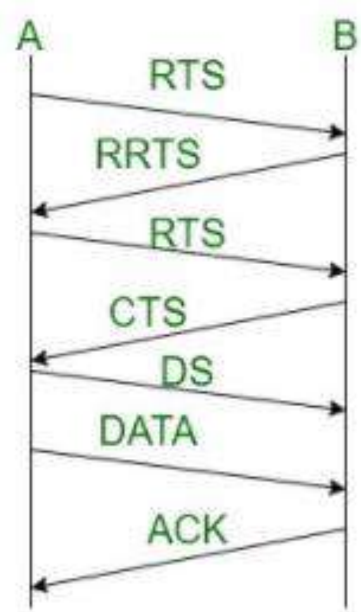
Multiple Access Collision Avoidance With Wireless (MACAW)

MACAW (MACA with Wireless)



- ***DATA Sending (DS) Packets :***
  - A DS packet should be sent after a successful RTS-CTS exchange, just before the data message itself.
  - The idea is to explicitly announce that the RTS-CTS succeeded, so that if a node can hear an RTS but not the CTS response, it does not attempt to transmit a message during the subsequent data transfer period.

  In MACA , an exposed node can received only the RTS and not the CTS packet



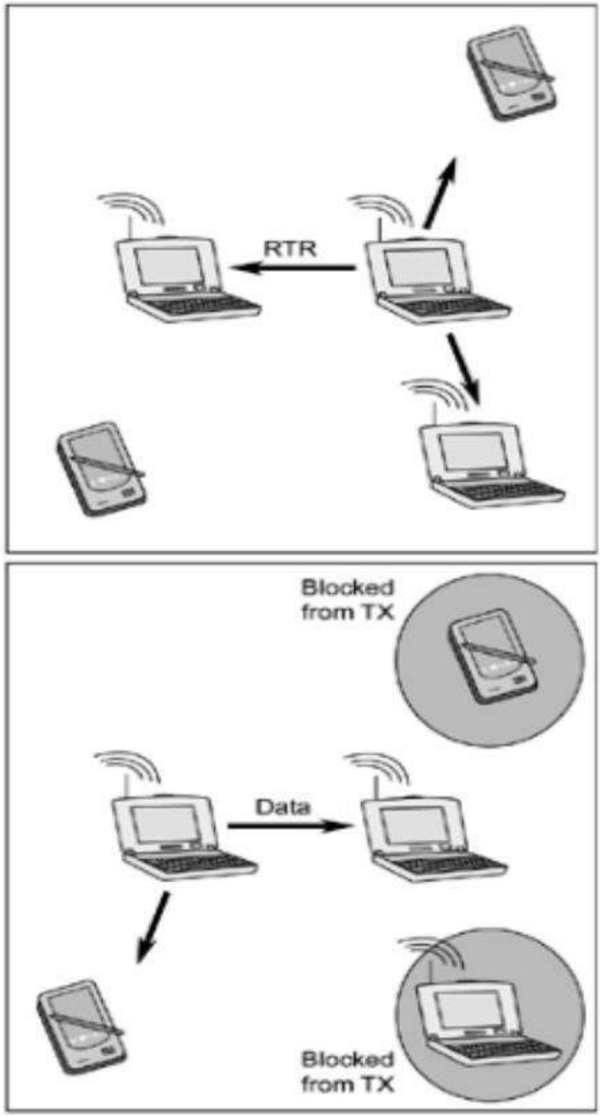
- ***READY for RTS (RRTS):***
  - if a receiver hears an RTS while it is deferring any transmissions, at the end of the deferral period it replies with an RRTS ("ready for RTS") packet, prompting the sender to resend the RTS.

MAC LAYER – CONTENTION-BASED MEDIUM ACCESS

Multiple Access Collision Avoidance By Invitation(MACA-BI)

MACA-BI(MACA By Invitation)

- MACA-BI uses only a two-way handshake.
- No RTS. the CTS message is renamed as RTR (Ready To Receive).
- **Type:** Receiver initiated MAC Protocol
  - A node cannot transmit data unless it has received an invitation from the receiver.
- Receiver node does not necessarily know that the source has data to transmit.
  - receiver needs to predict if node has data to transmit to it.
- The timeliness of the invitation will affect communication performance.



MAC LAYER – CONTENTION-BASED MEDIUM ACCESS

Multiple Access Collision Avoidance By Invitation(MACA-BI)

MACA-BI(MACA By Invitation)

- Packet queue length and arrival rate information is piggyback into each data packet so that the receiver is aware of the transmitter's backlog.

- For constant bit rate (CBR) traffic, the efficiency of MACA-BI will be high since the prediction scheme will work fine. However, will not perform well in case of bursty traffic.

- To enhance the communication performance of MACA-BI under non-stationary traffic situations
  - a node may still transmit an RTS if the transmitter's queue length or packet delay exceeds a certain acceptable threshold before an RTR is issued.

  - MACA-BI now reverts back to MACA.

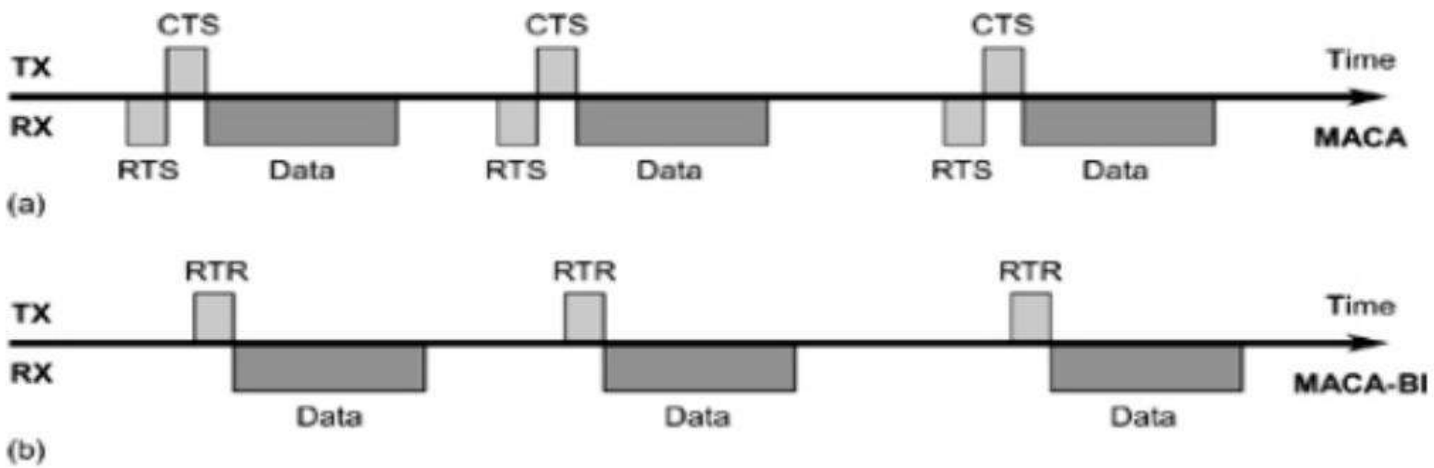
MAC LAYER – CONTENTION-BASED MEDIUM ACCESS

Multiple Access Collision Avoidance By Invitation(MACA-BI)

MACA Vs MACA-BI

- As MACA-BI only uses a single control message, this turn around limitation is reduced (i.e., up to 25 microseconds).

- MACA-BI is less likely to suffer from control packet collision since it uses half as many control packets as MACA



MAC LAYER – CONTENTION-BASED MEDIUM ACCESS

IEEE 802.11

- Published in 1999 by the Institute of Electrical and Electronics Engineers (IEEE)
  - specifies the physical and data link layers of the OSI model for wireless connections

- Often referred to as ***Wireless Fidelity (Wi-Fi)***
  - certification given by Wi-Fi Alliance, a group that ensures compatibility between hardware devices that use the 802.11 standard

- Wi-Fi combines concepts found in ***CSMA/CA*** and ***MACAW***, but also offers features to preserve energy

- Two modes of operation
  - ***Point Coordination Function (PCF)*** mode
    - Communication among devices goes through a central entity called an access point (AP) or base station (BS): **managed mode**

  - ***Distributed Coordination Function (DCF)*** mode
    - Devices communicate directly with each other: **ad-hoc mode**

MAC LAYER – CONTENTION-BASED MEDIUM ACCESS

IEEE 802.11

IEEE 802.11 is based on *CSMA/CA*

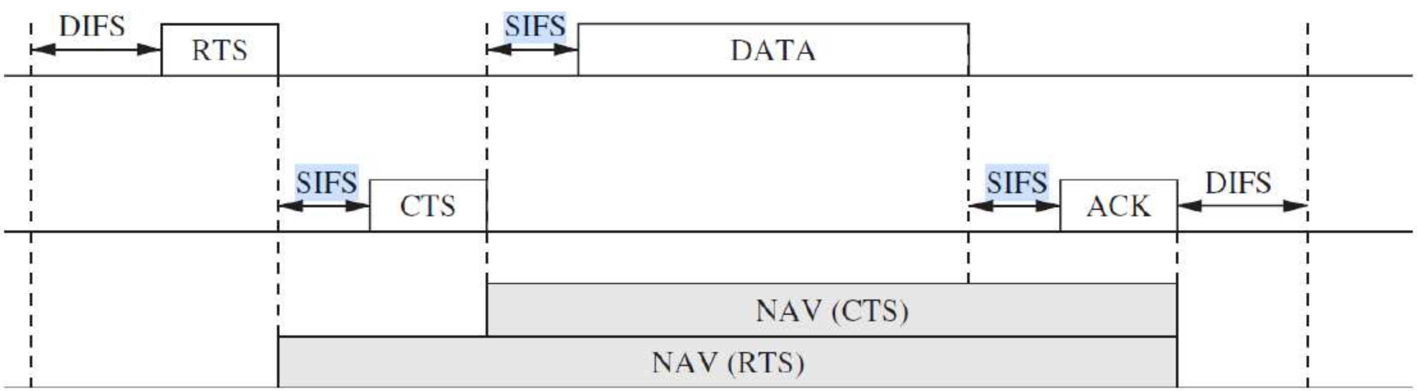
- before a node transmits, it first senses the medium for activity
- the node is allowed to transmit, if the medium is idle for at least a time period called the DCF interframe space (DIFS)
- otherwise the device executes a back-off algorithm to defer transmission to a later time
- this algorithm randomly selects a number of time slots to wait and stores this value in a back-off counter
- for every time slot that passes without activity on the network, the counter is decremented and the device can attempt transmission when this counter reaches zero
- if activity is detected before the counter reaches zero, the device waits until the channel has been idle for a period of DIFS before it continues to decrement the counter value

MAC LAYER – CONTENTION-BASED MEDIUM ACCESS

IEEE 802.11

- After a successful transmission
  - Receiver device responds with an acknowledgment after waiting for a time period called the ***short interframe space*** (SIFS)
  - The value of SIFS is smaller than the value of DIFS to ensure that no other device accesses the channel before the receiver can transmit its acknowledgment

- Once a node A makes a reservation using RTS and CTS control messages
  - Another neighboring node B, overhearing the RTS message, must refrain from accessing the medium until node A's transmission has been completed and acknowledged
  - However, this would mean that node B has to continuously sense the medium to detect when it becomes idle again



IEEE 802.11 medium access control

MAC LAYER – CONTENTION-BASED MEDIUM ACCESS

IEEE 802.11

- Instead, A's RTS message carries the size of the data it will transmit
  - Allowing node B to estimate how long the transmission will take and to decide whether to enter a low-power sleep mode
  - Some neighboring nodes may only overhear CTS (but not RTS), therefore, the data size is also carried in the CTS message
  - Using the data size information, neighboring nodes set a network allocation vector (NAV) that indicates how long the medium will be unavailable
    - reduces the need for continuously sensing the medium, allowing a node to save power

MAC LAYER – CONTENTION-BASED MEDIUM ACCESS

IEEE 802.11

PCF mode

- Access point (AP) coordinates channel access to ensure collision-free communication
  - periodically broadcasts a beacon to its client devices (includes list of devices with data pending at AP)

- During contention-free period, AP transmits these packets to its client devices

- AP can also poll client devices to allow them to initiate data transfers

- AP uses a wait period called the PCF interframe space (PIFS)
  - PIFS is shorter than DIFS, but longer than SIFS
  - Ensures that PCF traffic has priority over traffic generated by devices operating in the DCF mode, without interfering with control messages in the DCF mode such as CTS and ACK

MAC LAYER – CONTENTION-BASED MEDIUM ACCESS

IEEE 802.11

- Focus of IEEE 802.11 is on providing fair access to the medium with support for high throughput and mobility
  - Since devices spend a large amount of time listening to the medium and collisions occur frequently, this standard incurs large overheads, including significant energy costs

- *Energy consumption problem*
  - IEEE 802.11 offers a ***power saving mode (PSM)*** for devices operating in the PCF mode.

  - Devices can inform the AP that they wish to enter a low-power sleep mode using special control messages.

  - These devices wake up periodically to receive beacon messages from the AP to determine if they should stay awake to receive incoming messages.

  - Saves energy, but only works in the infrastructure mode and it is not specified when or how long devices should sleep.

MAC LAYER – CONTENTION-BASED MEDIUM ACCESS

IEEE 802.15.4

- The Institute of Electrical and Electronics Engineers (IEEE) supports many working groups to develop and maintain wireless and wired communications standards.

- For example:
  - 802.3 is Wired Ethernet and
  - 802.11 is for Wireless LANs (WLANs), also known as Wi-Fi.
  - The 802.15 group of standards specifies a variety of wireless personal area networks (WPANs) for different applications.
    - For instance, 802.15.1 is Bluetooth.
    - 802.15.3 is a high-data-rate category for ultra-wideband (UWB) technologies.
    - 802.15.6 is for body area networks (BAN). There are several others.

MAC LAYER – CONTENTION-BASED MEDIUM ACCESS

IEEE 802.15.4

- The 802.15.4 category is probably the largest standard for low-data-rate WPANs.

- It has many subcategories.
  - The 802.15.4 category was developed for low-data-rate monitor and control applications and extended-life low-power-consumption uses.

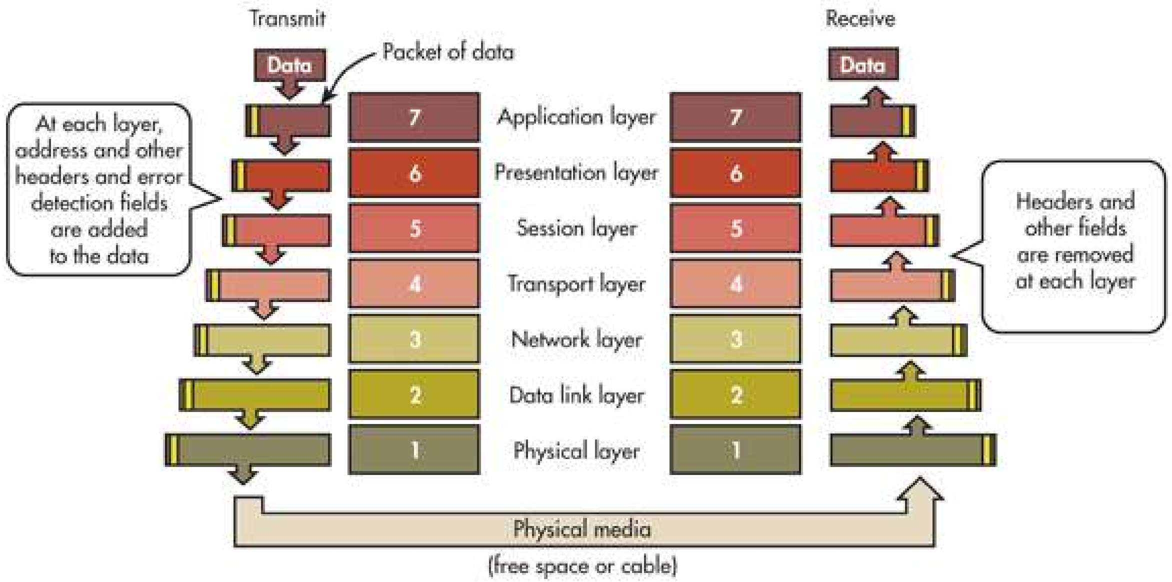
  - The basic standard with the most recent updates and enhancements is 802.15.4a/b, with 802.15.4c for China, 802.15.4d for Japan, 802.15.4e for industrial applications, 802.15.4f for active (battery powered) radio-frequency identification (RFID) uses, and 802.15.4g for smart utility networks (SUNs) for monitoring the Smart Grid.

  - All of these special versions use the same base radio technology and protocol as defined in 802.15.4a/b.

MAC LAYER – CONTENTION-BASED MEDIUM ACCESS

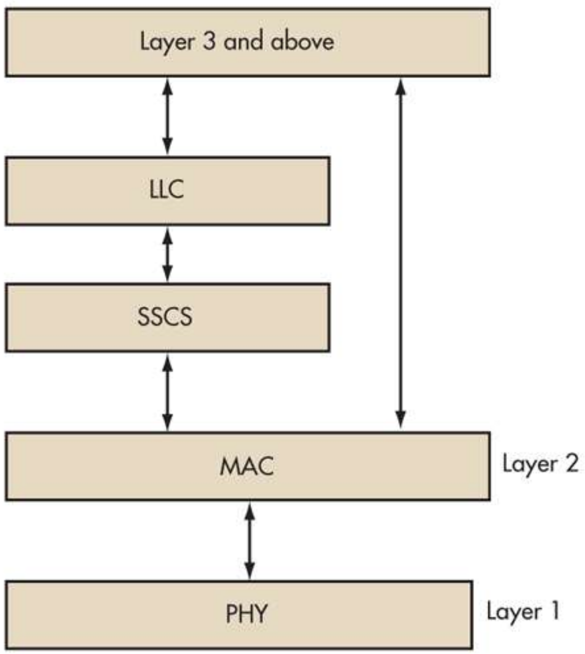
IEEE 802.15.4

The 802.15.4 standard defines the physical layer (PHY) and media access control (MAC) layer of the Open Systems Interconnection (OSI) model of network operation.



- *Most networking systems, both wired and wireless, use the OSI communications model. Most systems also use at least the first four layers, but many do not use all seven layers.*

- *The 802.15.4 standard uses only the first two layers plus the logical link control (LLC) and service specific convergence sub-layer (SSCS) additions to communicate with all upper layers as defined by additional standards.*



MAC LAYER – CONTENTION-BASED MEDIUM ACCESS

IEEE 802.15.4

- Created for low-power devices in the 868 MHz, 915 MHz, and 2.45 GHz frequency bands
- Supports two transmission modes:
  - UWB PHY
    – bit rates: 110 kbps, 851 kbps (nominal), 6.81 kbps, and 27.24 Mbps
  - CSS PHY
    – bit rates: 1 Mbps (nominal) and 250 kbps

- With regard to channel access, 802.15.4 uses carrier sense multiple access with collision avoidance (CSMA-CA). This multiplexing approach lets multiple users or nodes access the same channel at different times without interference.

OPTIONS FOR FREQUENCY ASSIGNMENTS

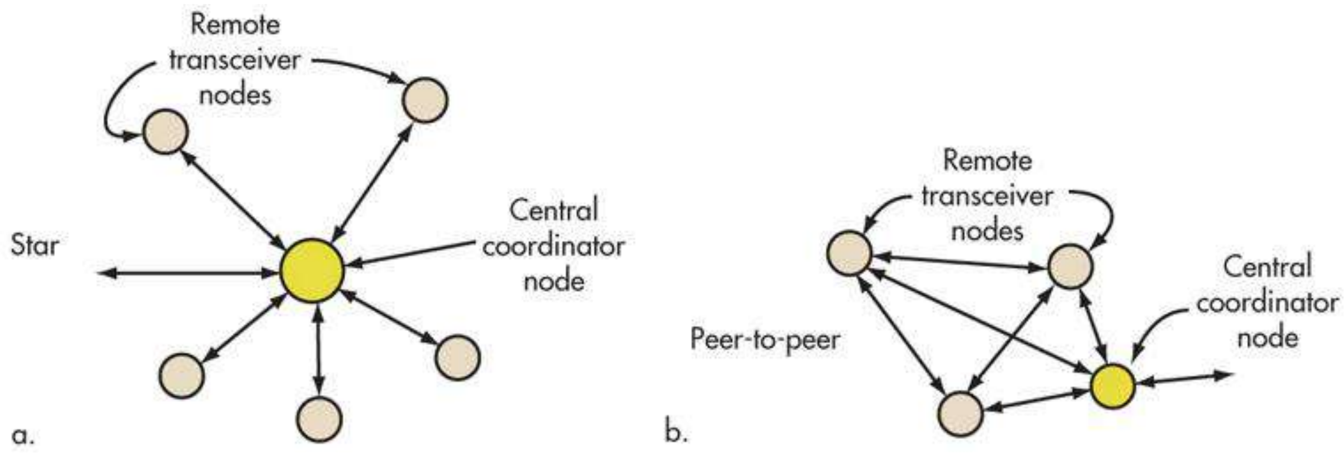
Geographical regions	Europe	Americas	Worldwide
Frequency assignment	868 to 868.6 MHz	902 to 928 MHz	2.4 to 2.4835 GHz
Number of channels	1	10	16
Channel bandwidth	600 kHz	2 MHz	5 MHz
Symbol rate	20 ksymbols/s	40 ksymbols/s	62.5 ksymbols/s
Data rate	20 kbits/s	40 kbits/s	250 kbits/s
Modulation	BPSK	BPSK	Q-QPSK

MAC LAYER – CONTENTION-BASED MEDIUM ACCESS

IEEE 802.15.4

- Transmission range varies considerably depending on the nature of the path that must for the most part be line of sight (LOS). Under the best conditions the range can be as great as 1000 meters with a clear outdoor path. Most applications cover a shorter range of 10 to 75 meters.

- With regard to networking capability, *802.15.4 standard defines the star and peer-to-peer common network topologies.*
    - One of them is a basic **Star**. All communications between nodes must pass through the central coordinator node.
    - A basic **peer-to-peer (P2P)** topology is also defined. Any device may then talk to any other device. This basic topology may be expanded into other topologies in the upper network layers, such as the popular mesh topology.



MAC LAYER – CONTENTION-BASED MEDIUM ACCESS

Zigbee

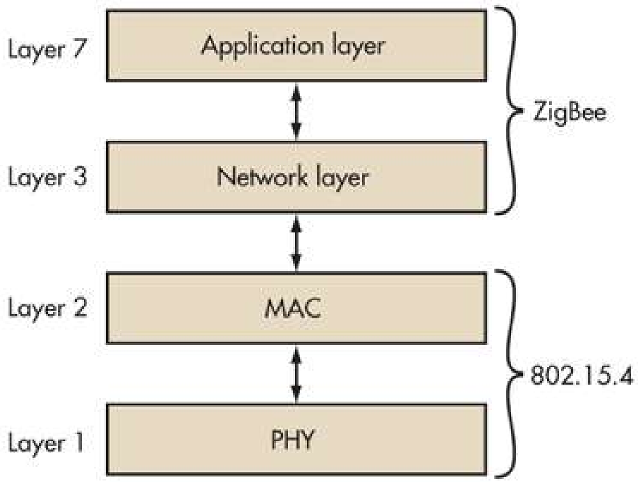
- The most widely deployed enhancement to the 802.15.4 standard is ZigBee, which is a standard of the ZigBee Alliance. The organization maintains, supports, and develops more sophisticated protocols for advanced applications.

- It uses layers 3 and 4 to define additional communications features *(Fig.)*.

- These enhancements include authentication with valid nodes, encryption for security, and a data routing and forwarding capability that enables mesh networking.

- The most popular use of ZigBee is wireless sensor networks using the mesh topology.

- The ZigBee protocol is defined by layer 3 and above. It works with the 802.15.4 layers 1 and 2.



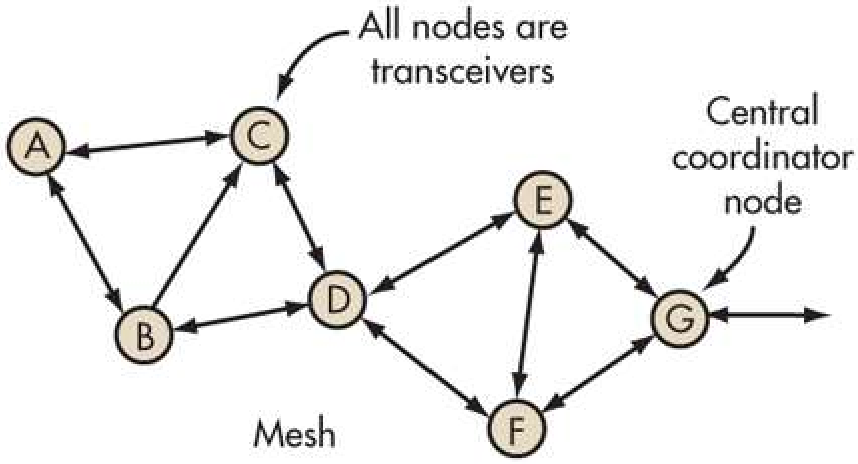
MAC LAYER – CONTENTION-BASED MEDIUM ACCESS

Zigbee

- There are three kinds of nodes in a ZigBee network:
    1. **Coordinator**: is the "master" device, it governs all the network
    2. **Routers**: they route the information which sent by the end devices
    3. **End device**: (the motes): they are the sensor nodes, the ones which take the information from the environment

- Coordinator and routes can not be battery powered, motes can. ZigBee creates **star topologies**. There are some basic rules:
    - The end devices connect to a router or a coordinator.
    - The routers can connect among them and with the coordinator.
    - The routers and coordinators can not sleep. They have to save in their buffer the packets which go to the end devices.
    - The end devices can sleep.

- In a mesh network, each node communicates with its closest neighbor as conditions permit. Note that there are alternate paths between any two nodes.



MAC LAYER – CONTENTION-BASED MEDIUM ACCESS

Zigbee

- ZigBee offers basically four kinds of different services:
  - **Extra Encryption services** (application and network keys implement extra 128bit AES encryption)
  - **Association and authentication** (only valid nodes can join to the network).
  - **Routing protocol**: AODV, a reactive ad hoc protocol has been implemented to perform the data routing and forwarding process to any node in the network.
  - **Application Services**: An abstract concept called **"cluster"** is introduced. Each node belongs to a predefined cluster and can take a predefined number of actions.
- ZigBee is also available in a version that supports energy harvesting where no battery or ac mains power is available.
- One of the key benefits of ZigBee is the availability of pre-developed applications. These upper-layer software additions implement specialized uses for ZigBee.

MAC LAYER – CONTENTION-BASED MEDIUM ACCESS

IEEE 802.15.4 Versus ZigBee

- 802.15.4 is thought to be a protocol to get point to point and energy efficient communications.
- ZigBee defines extra services (start topology routing, encryption, application services) over 802.15.4.
- ZigBee creates semi-centralized networks where just the end devices can sleep.
- Different completely distributed mesh algorithms are being used over 802.15.4 is the protocol used to create
- Both Waspmote and SquidBee benefit from all the 802.15.4, ZigBee and Digimesh protocols and support all the frequency bands 869MHz, 900MHz and 2.4GHz.

MAC LAYER – CONTENTION-BASED MEDIUM ACCESS

Characteristics of MAC Protocols in Sensor Networks

- Most MAC protocols are built for fairness
  - everybody should get an equal amount of resources
  - no one should receive special treatment

- In a WSN, all nodes cooperate to achieve a common purpose, therefore fairness is less of a concern.

- Instead, wireless nodes are mostly concerned with energy consumption.

- Sensing applications may value low latency or high reliability over fairness.

- The main characteristics and design goals for MAC protocols of WSNs:
  - *Energy Efficiency*
  - *Scalability*
  - *Adaptability*
  - *Low Latency and Predictability*
  - *Reliability*

MAC LAYER – CONTENTION-BASED MEDIUM ACCESS

Characteristics of MAC Protocols in Sensor Networks

Energy Efficiency

- Sensor nodes must operate using finite energy sources, therefore MAC protocols must consider energy efficiency

- Common technique: ***Dynamic Power Management (DPM)***
  - A resource can be moved between different operational modes such as active, idle, and asleep
  - For resources such as the network, the active mode can group together multiple different modes of activity, e.g., transmitting and receiving

- **Periodic Traffic Models** are very common in WSNs
  - Significant energy savings can be obtained by putting a device into a low-power sleep mode.
  - Fraction of time a sensor nodes spends in active mode is called the duty cycle.
  - Often very small due to the infrequent and brief data transmissions occurring in most sensor networks.

MAC LAYER – CONTENTION-BASED MEDIUM ACCESS

Characteristics of MAC Protocols in Sensor Networks

Energy Efficiency

- These are the main cause of energy consumption in WSN. It is required to use energy efficiently for the following operations.
  - idle listening (i.e., a device staying in idle mode unnecessarily)
  - inefficient protocol designs (e.g., large packet headers)
  - reliability features (collisions requiring retransmissions or other error control mechanisms)
  - control messages to address the hidden-terminal problem
  - choice of modulation scheme
  - choice of transmission rate
  - over emitting (that is, using larger transmit powers than necessary)

MAC LAYER – CONTENTION-BASED MEDIUM ACCESS

Characteristics of MAC Protocols in Sensor Networks

Scalability

- Many wireless MAC protocols have been designed for use in infrastructure based networks.
  - Access points or controller nodes arbitrate access to the channel and perform some centralized coordination and management functions.

- Most wireless sensor networks rely on multi-hop and peer-to-peer communications without centralized coordinators.

- MAC protocols must be able to allow for efficient use of resources without incurring unacceptable overheads, particularly in very large networks.

- MAC protocols based on CDMA have to cache a large number of code (may be impractical for resource-constrained sensor devices).

- WSNs are not only constrained in their energy resources, but also in their processing and memory capacities.

- Therefore, MAC protocols should not impose excessive computational burden should not require too much memory to save state information

MAC LAYER – CONTENTION-BASED MEDIUM ACCESS

Characteristics of MAC Protocols in Sensor Networks

Adaptability

- A key characteristic of a WSN is its ability to self-manage
  - adapt to changes in the network
  - including changes in topology, network size, density, and traffic characteristics

- A MAC protocol for a WSN should be able to gracefully adapt to such changes without significant overheads

- This requirement generally favors protocols that are dynamic in nature
  - Protocols that make medium access decisions based on current demand and network state

- Protocols with fixed assignments (e.g., TDMA with fixed-size frames and slots) may incur large overheads due to adaptations of such assignments that may affect many or all nodes in the network

MAC LAYER – CONTENTION-BASED MEDIUM ACCESS

Characteristics of MAC Protocols in Sensor Networks

Low Latency & Predictability

- Many WSN applications have timeliness requirements
  - Sensor data must be collected, aggregated, and delivered within certain latency constraints or deadlines.
  - Example: wildfire detection (sensor data must be delivered to monitoring stations in a timely fashion to ensure timely responses).

- MAC protocol design
  - Choice of frame size and slot allocations in TDMA-based protocols may lead to large delays.
  - In contention-based protocols, nodes may be able to access the wireless medium sooner (than TDMA), but collisions and the resulting retransmissions incur delays.
  - Choice of MAC protocol can affect how predictable the experienced delay is (expressed as upper latency bounds).
  - Some contention-based MAC protocols allow the theoretical possibility of starvation.

MAC LAYER – CONTENTION-BASED MEDIUM ACCESS

Characteristics of MAC Protocols in Sensor Networks

Reliability

- Finally, reliability is a common requirement for most communication networks.

- The design of the MAC protocol can contribute to increased reliability by detecting and recovering from transmission errors and collisions (e.g., using acknowledgments and retransmissions).

- Particularly in wireless sensor networks, where node failures and channel errors are common, reliability is a key concern for many link-layer protocols.

MAC LAYER – CONTENTION-BASED MEDIUM ACCESS

Contention-free Protocols

Concept:

- Allow only one sensor node to access the channel at any given time
- Thereby avoiding collisions and message retransmissions
- Assuming a perfect medium and environment
  - i.e., no other competing networks or misbehaving devices exist that could otherwise cause collisions or even jam a channel

- Contention-free protocols allocate resources to individual nodes to ensure exclusive resource access by only one node at any given time.

- Exposes a number of desirable characteristics
  - Node knows exactly when it has to turn on its radio
  - During all other times, radio can be turned off to preserve energy
  - Fixed slot allocations impose upper bounds on delay
  - Difficult to design schedules for large networks
  - Difficult to handle changes in topology, density, traffic load

MAC LAYER – CONTENTION-BASED MEDIUM ACCESS

Contention-based Protocols

- These protocols do not rely on transmission schedules, instead they require other mechanisms to resolve contention when it occurs

- The main advantage of contention-based techniques is their simplicity compared to most schedule-based techniques
  - Schedule-based MAC protocols must save and maintain schedules or tables indicating the transmission order
  - Most contention-based protocols do not require to save, maintain, or share state information
  - This also allows contention-based protocols to adapt quickly to changes in network topologies or traffic characteristics

- However, they typically result in higher collision rates and overheads due to idle listening and overhearing (overheads usually refer to additional bits in a packet or additional packets such as control packets)

- They may also suffer from fairness issues (i.e., some nodes may be able to obtain more frequent channel accesses than others)

MAC LAYER – CONTENTION-BASED MEDIUM ACCESS

Various MAC Layer Protocols Developed for WSN

MAC Protocols

Contention-free Protocols

1. Traffic-Adaptive Medium Access (TRAMA) protocol
2. *Y-MAC*
3. *DESYNC-TDMA*
4. Low-Energy Adaptive Clustering Hierarchy (LEACH) protocol
5. Lightweight Medium Access Control (LMAC) protocol

Contention-based Protocols

1. Power Aware Multi-Access with Signaling (PAMAS) protocol
2. Sensor MAC (S-MAC) protocol
3. Timeout MAC (T-MAC) protocol
4. Pattern MAC (PMAC) protocol
5. Routing-Enhanced MAC (RMAC) protocol
6. Data-Gathering MAC (DMAC)
7. *Preamble Sampling and Wise MAC*
8. Receiver-Initiated MAC (RI-MAC) protocol

Hybrid MAC Protocols

1. Zebra MAC
2. *Mobility Adaptive Hybrid MAC*

Questions asked in University Examination

Long Questions (5 Marks)

1. Discuss the energy efficiency in MAC protocol.
2. Discuss the distributed assignment of network wide unique MAC address for WSN.
3. Write short notes on MAC sublayer management.
4. Why CSMA protocol fails to avoid collisions and inefficient in WSN?
5. What is early sleep problem in T-MAC protocol? Explain the solution to avoid the problem?
6. Write Short Note on:
   a. S-MAC
   b. T-MAC
   c. D-MAC
7. Explain in brief about Directional MAC Protocols for Ad hoc wireless network?
8. Briefly explain the Classifications of MAC Protocols?
9. Explain in brief about Interleaved CSMA Protocol?
10. How low power listening mode is used to conserve energy in B-MAC protocol?
11. Explain why the relay diversity scheme may not work well with some sleep-oriented MAC protocols proposed for sensor networks.
12. Explain different characteristics of Contention-Free MAC Protocols.
13. How future-request-to-send (FRTS) is used to avoid the Early sleeping problem in case of T-MAC protocol.

Questions asked in University Examination

Short Questions (2 Marks)

1. What are the advantages of Contention-based MAC protocols over Contention-free MAC protocols?
2. Explain exposed node problem with suitable example.
3. Explain the different components of B-MAC protocol.
4. Explain different characteristics of Contention-Free MAC Protocols.
5. Explain different goals of MCA protocol in wireless sensor network.
6. Differentiate between S-Mac and T-MAC.
7. List the different disadvantages of ALOHA protocol.
8. What do you mean by slotted ALOHA?
9. what are the classification of MAC protocol?
10. What are the MAC services of IEEE 802.11 that are not provided in traditional LAN 802.3?
11. List the advantages of ALOHA protocol.

BASIC ARCHITECTURAL FRAMEWORK

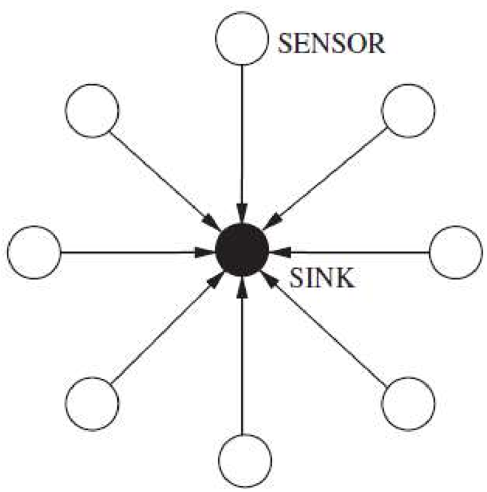
NETWORK LAYER

OUTLINES

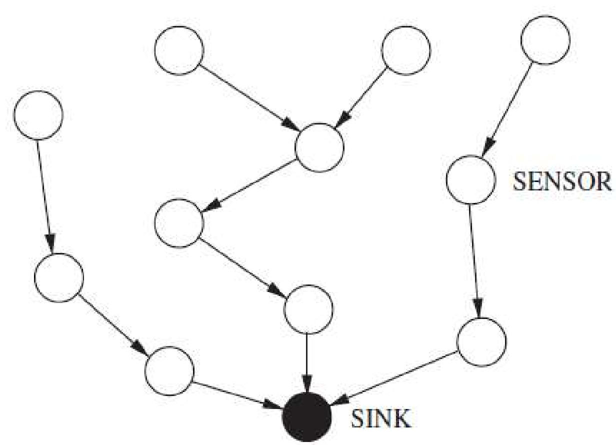
- Routing basics
- Data-centric routing
- Proactive routing
- On-demand routing
- Hierarchical routing
- Location-based routing
- QoS-based routing

NETWORK LAYER FUNCTIONS/RESPOSIBILITIES

- The key responsibility of the network layer is to find paths from data sources to sink nodes (e.g., gateways).
- In the *single-hop routing model*, all sensor nodes are able to communicate directly with the sink device.
- This *direct communication model* is the simplest approach, where all data travels a single hop to reach the destination.
- However, in practical settings, this single-hop approach is *unrealistic* and a *multi-hop communication model* must be used. In this case, the critical task of the network layer of all sensor nodes is to identify a path from the sensor to the sink across multiple other sensor nodes acting as *Relays.*



Single-hop routing model



multi-hop routing model (right)

NETWORK LAYER FUNCTIONS/RESPOSIBILITIES

- Data collected by sensor nodes in a Wireless Sensor Network (WSN) is typically propagated toward a base station (gateway) that links the WSN with other networks where the data can be visualized, analyzed, and acted upon.
- In large networks sensor nodes generate own information and serve as relays or forwarding nodes for other sensor nodes.

"Routing is the process of selecting or establishing the path through which message will be relayed to its destination. It is a responsibility of the network layer of the communication protocol stack."

- Routing protocol is responsible for finding and maintaining path from sensor to sink.
- When the nodes of a WSN are deployed in a *deterministic manner*, communication between them and the gateway can occur using *predetermined routes*.
- By contrast when the nodes are deployed in a *randomized manner*, the resulting *topologies are non-uniform and unpredictable*.

NETWORK LAYER FUNCTIONS/RESPOSIBILITIES

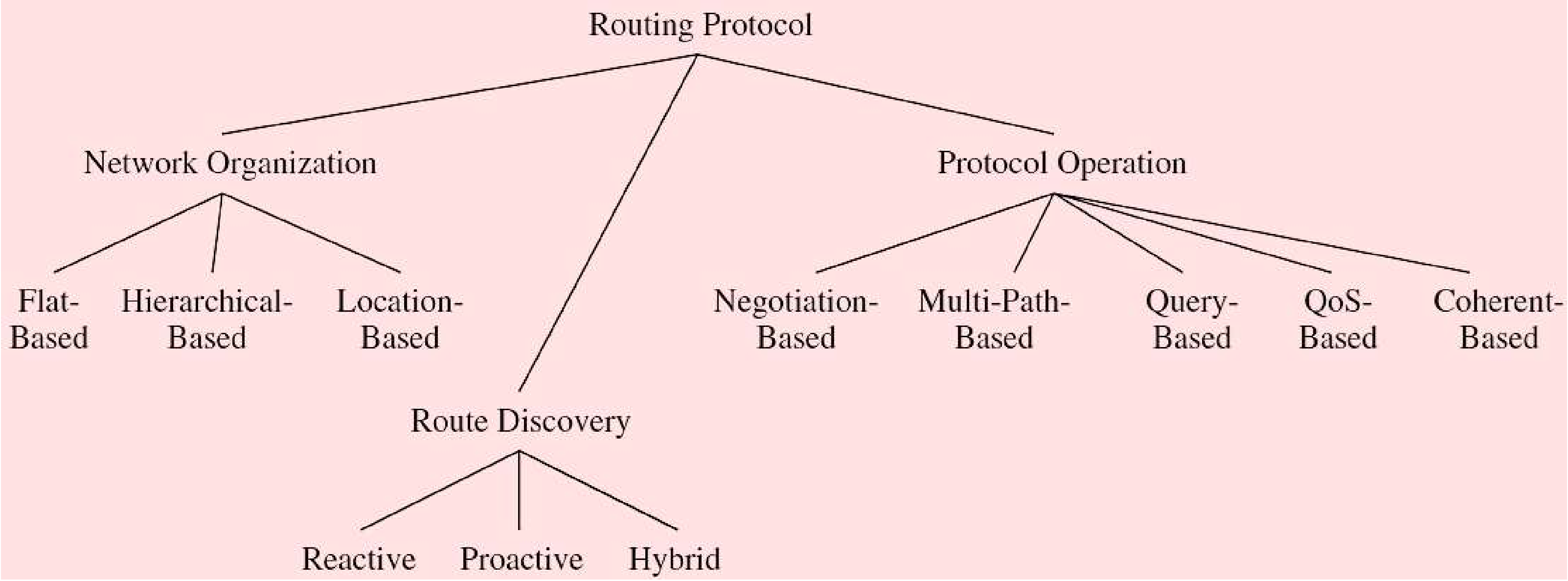
- Current problems at the network layer can be classified into three categories:
  - Topology control,
  - Routing, and
  - Coordination.
- A *well-organized network topology* can not only prolong the lifetime of a network, but also enhance data communications.
- *Quality of Service (QoS) routing* as well as multicast, broadcast, and geocast, the primary goal is to fulfill a given communication task successfully between nodes in the network.
- Wireless sensor actuator networks require *coordination* not only among sensors or actuators, but also between them.

NETWORK LAYER FUNCTIONS/RESPOSIBILITIES

- In WSN the design of a *routing protocol is challenging* due to the unique characteristics of the network as *resource scarcity and unreliability of the wireless medium.*

- The limited processing, storage, bandwidth, and energy capacities require routing solutions that are lightweight, while the frequent dynamic changes in a WSN require routing solutions that are adaptive and flexible.

- There are various ways to classify routing protocols. Figure presents three different classifications based on the network structure or organization, the route discovery process, and the protocol operation (Al-Karaki and Kamal 2004).



NETWORK LAYER FUNCTIONS/RESPOSIBILITIES

Network organization

Flat Topology

- Flat topology treats all nodes equally.
- Flat topology is mainly for homogeneous networks where all nodes are of same characteristics and have same functionality.
- Each router node routinely collects and distributes routing information with its neighboring routers. The entire participating node addressed by flat routing protocol performs an equal role in the overall routing mechanism.
- Examples are:
  - Gradient based routing (GBR),
  - Cougar,
  - Constrained anisotropic diffusion routing (CADR),
  - Rumor routing (RR)
- Advantages
  - Minimal overhead to maintain the infrastructure
  - Potential for the discovery of multiple routes between communicating nodes for fault tolerance

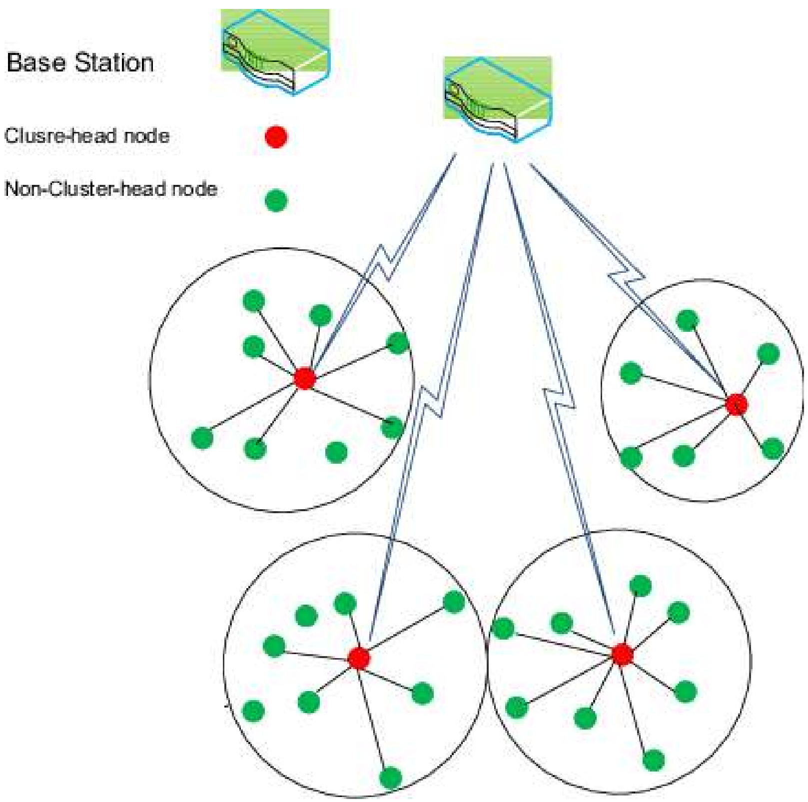
NETWORK LAYER FUNCTIONS/RESPOSIBILITIES

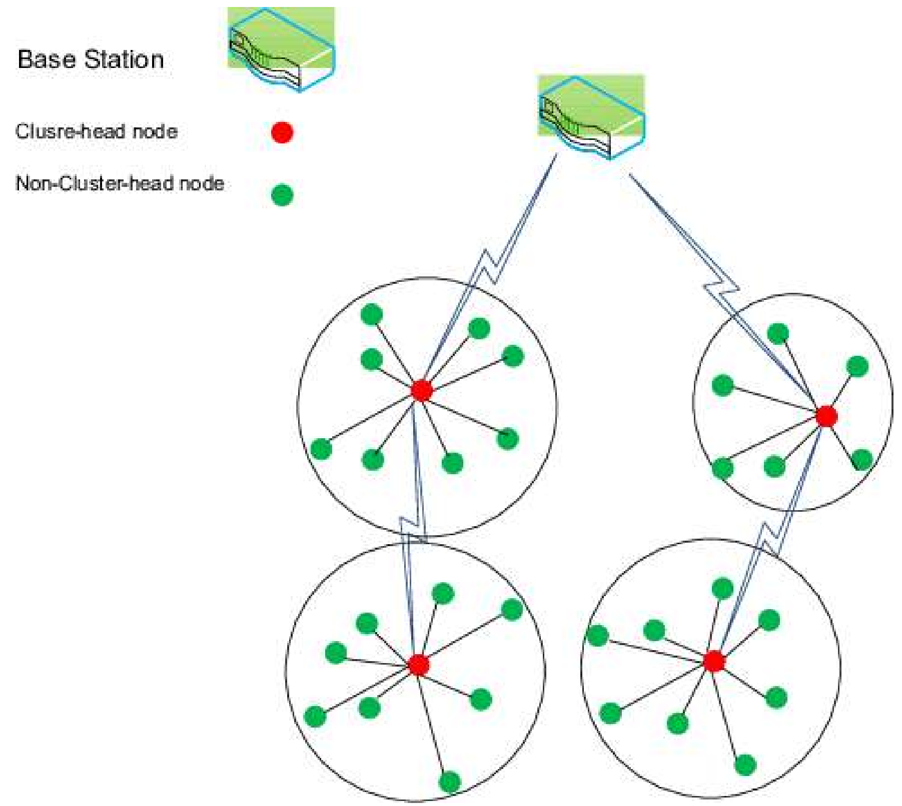
Network organization

Hierarchical

- Achieve energy efficiency, stability, and scalability.
- Different “roles” for different nodes (e.g., cluster heads versus cluster members).
- Mostly heterogeneous networks apply hierarchical routing protocols where some nodes are more advance and powerful than the other nodes.
- A node with higher residual energy will cluster head.
- The cluster head is responsible for coordinating activities within the cluster and forwarding information between clusters ▪
- The clustering scheme is more energy efficient and more easily manageable.
- Examples are:
  - Threshold sensitive energy efficient sensor network (TEEN)
  - Adaptive threshold sensitive energy efficient sensor network (APTEEN)
  - Low energy adaptive clustering hierarchy (LEACH)
  - The power-efficient gathering in sensor information systems (PEGASIS)
  - Virtual grid architecture routing (VGA)
  - Self-organizing protocol (SOP)
  - Geographic adaptive fidelity (GAF)

NETWORK LAYER FUNCTIONS/RESPOSIBILITIES





NETWORK LAYER FUNCTIONS/RESPOSIBILITIES

Network organization

Location-based:

- In location based routing the nodes have capability to locate their present location using various localization protocols.

- Location information helps in improving the routing procedure and also enables sensor networks to provide some extra services.

- Location- based routing is useful
  - In applications where the position of the node within the geographical coverage of the network is relevant to the query issued by the source node.

- Query may specify a specific
  - Area where a phenomenon of interest may occur or
  - The vicinity to a specific point in the network environment.

- Examples are:
  - SPEED
  - Geographical and energy aware routing (GEAR)
  - SPAN

NETWORK LAYER FUNCTIONS/RESPOSIBILITIES

Route Discovery Protocols

Reactive (on-demand):

- Find route only when needed.
- In Reactive Protocol if there are sudden changes in the sensed attribute beyond some pre-determined threshold value, the nodes immediately react. This type of protocol is used in time critical applications.
- Establish routes to a limited set of destinations on demand.
- Do not maintain global information across all nodes of the network.
- Rely on a dynamic route search to establish paths between a source and a destination.
- Typically involves flooding a route discovery query, with the replies traveling back along the reverse path.
- Strategies vary in the way
  - Control the flooding process to reduce communication overhead
  - The way routes are computed and reestablished when failure occurs.
- **Example:** The Threshold sensitive Energy Efficient sensor Network(TEEN) is an example of a reactive protocol.

NETWORK LAYER FUNCTIONS/RESPOSIBILITIES

Route Discovery Protocols

Proactive (Table-driven):

- Establish routes before they are needed.
- In a Proactive Protocol the nodes switch on their sensors and transmitters, sense the environment and transmit the data to a BS through the predefined route.
- Example: The Low Energy Adaptive Clustering hierarchy protocol (LEACH) utilizes this type of protocol.
- Also referred as table driven.
- Periodic dissemination of routing information.
- Maintain consistent and accurate routing tables across all nodes of the network. .
- The structure of the network can be either
  - Flat proactive routing – strategies needs to compute optimal paths – The overhead required to compute these paths in a dynamically changing environment
  - Hierarchical routing – Better suited to meet the routing demands of large ad hoc networks.

NETWORK LAYER FUNCTIONS/RESPOSIBILITIES

Route Discovery Protocols

Hybrid

- Protocols with reactive and proactive characteristics.
- They first compute all routes and then improve the routes at the time of routing. Example: Adaptive Periodic TEEN(APTEEN) is an example of Hybrid Protocols.
- To achieve stability and scalability in large networks.
- Network is organized into mutually adjacent clusters maintained dynamically as nodes join and leave their assigned clusters.
- A hybrid routing strategy can be adopted whereby – Proactive routing is used within a cluster – Reactive routing is used across clusters.
- The main challenge is to reduce the overhead required to maintain the clusters.

NETWORK LAYER FUNCTIONS/RESPOSIBILITIES

Protocol Operation

Multipath routing protocols

- Multi-path routing protocols provide multiple paths for data to reach the destination providing load balancing, low delay and improved network performance as a result.

- The multiple routing protocol also provide alternate path in case of failure of any path.

- Dense networks more interested in multiple path networks.

- To keep the paths alive some sort of periodic messages have to a send after some specific intervals hence multiple path routing is not more energy efficient.

- Multipath routing protocols are:
  - Multi path and Multi SPEED (MMSPEED)
  - Sensor protocols for information via negotiation (SPIN)

NETWORK LAYER FUNCTIONS/RESPOSIBILITIES

Protocol Operation

Query based routing protocol

- These type of routing protocols are mostly receiver-initiated.

- The sensor nodes will only send data in response to queries generated by the destination node.

- The destination node sends query of interest for receiving some information through the network and the target node sense the information and send back to the node that has initiated the request.

- **The examples are**
  - Sensor protocols for information via negotiation (SPIN)
  - Directed diffusion (DD)
  - COUGAR

NETWORK LAYER FUNCTIONS/RESPOSIBILITIES

Protocol Operation

Negotiation based routing protocols

- In these types of protocols, to keep the redundant data transmission level at minimum, the sensor nodes negotiate with the other nodes a and share their information with the neighboring nodes about the resources available and data transmission decisions are made after the negotiation process.

- Examples are:
    - Sensor protocols for information via negotiation (SPAN)
    - Sequential assignment routing (SAR)
    - Directed diffusion (DD)

NETWORK LAYER FUNCTIONS/RESPOSIBILITIES

Protocol Operation

QoS based routing protocols

- To get good Quality of Service these protocols are used.

- QoS aware protocols try to discover path from source to sink that satisfies the level of metrics related to good QoS like throughput, data delivery, energy and delay.

- But also making the optimum use of the network resources.

- **Examples are:**
  - Sequential assignment routing (SAR)
  - SPEED
  - Multi path and Multi SPEED (MMSPEED)

NETWORK LAYER FUNCTIONS/RESPOSIBILITIES

Protocol Operation

Coherent data processing routing protocol

- The nodes perform minimum processing (time stamping, data compression etc.) on the data before transmitting it towards the other sensor nodes or aggregators.

- Aggregator performs aggregation of data from different nodes and then passes to the sink node.

- Examples are:
  - Directed diffusion (DD)

NETWORK LAYER FUNCTIONS/RESPOSIBILITIES

Routing Metrices

Wireless sensor networks and their applications vary widely in their constraints and characteristics, which must be taken into consideration in the design of a routing protocol.

- Node deployment
- Energy consumption without losing accuracy
- Data reporting method
- Node/link heterogeneity
- Scalability
- Data aggregation
- Quality of service

NETWORK LAYER FUNCTIONS/RESPOSIBILITIES

Routing Metrices

Routing metrics are used to express a variety of objectives of a routing protocol with respect to the consumption of these resources or the performance an application perceives.

- Minimum hop (shortest hop)
  - The routing protocol attempts to find the path from the sender to the destination that requires the smallest number of relay nodes (hops).

- Energy
  - minimum energy consumed per packet
  - maximum time to network partition
  - minimize variance in node power levels
  - maximum (average) energy capacity
  - maximum minimum energy capacity

- Quality-of-Service
  - latency (delay), throughput, jitter, packet loss, error rate

- Robustness
  - link quality, link stability

NETWORK LAYER FUNCTIONS/RESPOSIBILITIES

Data Centric Routing

- Data centric routing is used to control the redundancy of data.
- This is because the sensors nodes don not have global identification number which specifies them uniquely; hence data is transmitted to each node with significant redundancy.
- In data centric routing, the sink requests for data by sending the query, so that the nearest sensor node transmits the data selected and that is understood in the query.
- The property of data is specified by attribute based naming.
- The protocols used in data centric routing include:
  - Flooding and Gossiping,
  - Sensor Protocols for Information via Negotiation (SPIN),
  - Directed Diffusion,
  - Energy–aware routing, Rumor routing,
  - Constrained Anisotropic Diffusion Routing (CADR),
  - COUGAR,
  - ACtiveQUery forwarding In sensoRnEtworks (ACQUIRE)

NETWORK LAYER FUNCTIONS/RESPOSIBILITIES

Traditional Routing Techniques

Flooding

- Every sensor node (re-)broadcasts sensor data to all of its neighbors.
- Simple and reliable technique.
- Incurs large traffic overhead (maximum-hop counts and sequence numbers can be used to limit broadcasts and eliminate duplicates).
- Flooding Uses a reactive approach whereby each node on receiving a data or control packet sends the packet to all its neighbors.
- After transmission, a packet follows all possible paths.
- Unless the network is disconnected, the packet will eventually reach its destination
- If network topology changes, the packet transmitted follows the new routes.
- **Disadvantages :**
  - Flooding cause packets to be replicated indefinitely by network nodes.

NETWORK LAYER FUNCTIONS/RESPOSIBILITIES

Traditional Routing Techniques

Flooding Problem

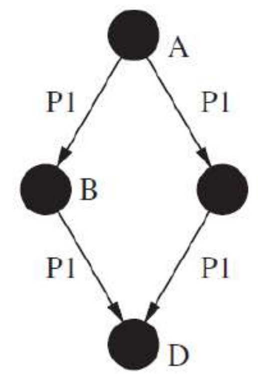
- **Packet circulates indefinitely in the network.**
  - ***Solution – Hop Count Field***
    - Field is included in the packet with value set to approximately the diameter of the network.
    - As the packet travels across the network, the hop count is decremented by one.
    - When the hop count reaches zero, the packet is simply discarded.
  - ***Other Solution – Using a time-to-live field***
    - To records the number of time units that a packet is allowed to live within the network
    - At the expiration of this time, the packet is no longer forwarded.
  - ***Drop all the packets that it has already forwarded***
    - Requires maintaining at least a recent history of the traffic, to keep track of which data packets have already been forwarded.

NETWORK LAYER FUNCTIONS/RESPOSIBILITIES

Traditional Routing Techniques

Drawbacks of Flooding

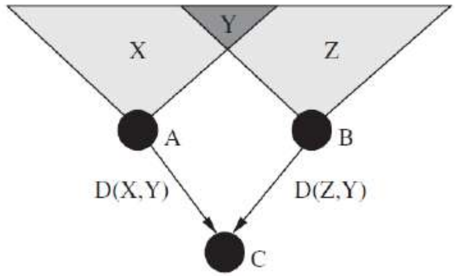
Traffic Implosion Effect is caused by duplicate control or data packets being sent repeatedly to the same node.



implosion problem

Node A broadcasts packet P1 to both of its neighbors, B and C. B forwards this packet to its own neighbor D and, finally, C also forwards this packet to node D. Even if D discards the duplicate packet, energy has been wasted on the transmission of the packet from C to D.

Overlapping Problem occurs when two nodes covering the same region send packets containing similar information to the same node



overlap problem

Here, sensors A and B share the region marked as Y. Therefore, these sensors gather overlapping data and both forward their collected information to their neighbor C.

Resource Blindness The forwarding rule used to route packets does not take into consideration the energy constraints of the sensor nodes.

Node's energy may deplete rapidly, reducing considerably the lifetime of the network.

NETWORK LAYER FUNCTIONS/RESPOSIBILITIES

Traditional Routing Techniques

Gossiping

- A variation of flooding is gossiping (Hedetniemi *et al.* 1988), where a node does not necessarily broadcast data.

- Instead, it uses a probabilistic approach, where it decides to forward the data to its own neighbors with a probability *p* and to discard the data with a probability derivative approach referred to as gossiping.

- Each node sends the incoming packet to a randomly selected neighbor.

- The neighbor selected randomly chooses one of its own neighbors and forwards the packet to the neighbor chosen.

- The process continues iteratively
  - Until the packet reaches its intended destination OR
  - The maximum hop count is exceeded

NETWORK LAYER FUNCTIONS/RESPOSIBILITIES

Traditional Routing Techniques

Gossiping

- *Advantages*
  - Gossiping avoids the implosion problem by limiting the number of packets that each node sends to its neighbor to one copy.

- *Disadvantages*
  - The latency that a packet suffers on its way to the destination may be excessive, particularly in a large network.
  - Caused primarily by the random nature of the protocol, which, in essence, explores one path at a time.

NETWORK LAYER FUNCTIONS/RESPOSIBILITIES

Sensor Protocols for Information via Negotiation (SPIN)

- Example of data-centric routing

- Objective – To efficiently disseminate observations gathered by individual sensor nodes to all the sensor nodes in the network

- Uses negotiations to address all problems of flooding
  - Implosion: nodes negotiate before data transmission
  - Overlap: nodes negotiate before data transmission
  - Resource blindness: resource manager keeps track of actual resource consumption and adapts routing and communication behavior

- SPIN uses meta-data to succinctly and completely describe sensor data

- Requirements:
  - if *x* describes the meta-data for some sensor data *X*, the size of *x* (in bytes) must be less than the size of *X.*
  - if two pieces of sensor data differ, their meta-data representations should differ too.

NETWORK LAYER FUNCTIONS/RESPOSIBILITIES

Sensor Protocols for Information via Negotiation (SPIN)

- The basic of protocols are
  - Data negotiation
  - Resource adaptation

- Nodes running SPIN "learn" about the content of the data before any data are transmitted between network nodes.

- Nodes associate metadata with data they produce to perform negotiations before transmitting the actual data (eliminates the possibility of overlap).

- A receiver that expresses interest in the data content can send a request to obtain the data advertised.

- It assures
  - Data are sent only to interested nodes
  - Eliminating traffic implosion
  - Reducing significantly the transmission of redundant data throughout the network

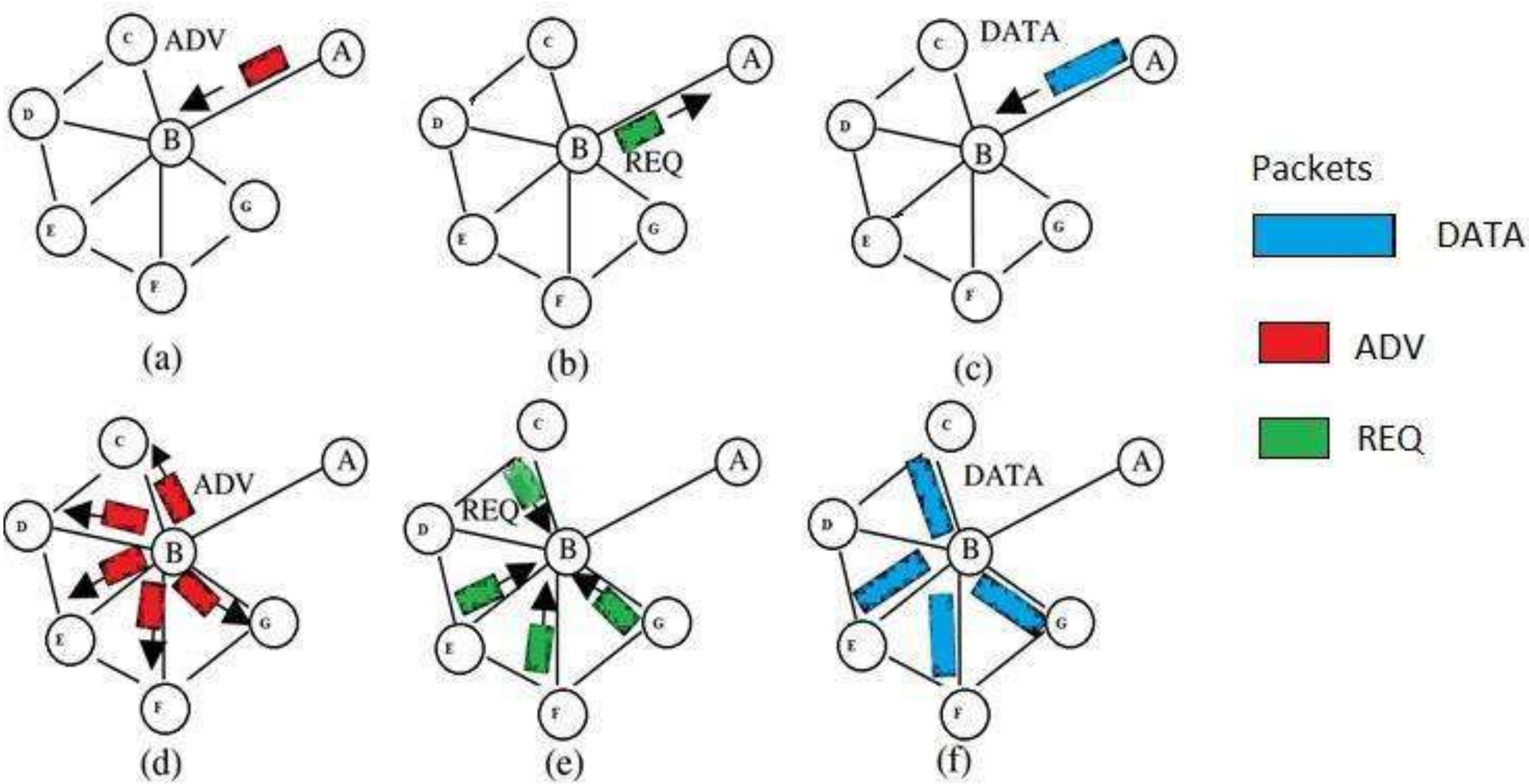
NETWORK LAYER FUNCTIONS/RESPOSIBILITIES

Sensor Protocols for Information via Negotiation (SPIN)

- To carry out negotiation and data transmission, nodes running SPIN use three types of messages:
  - ADV (advertisement), REQ (request), DATA
- *ADV (Advertisement)*
  - Used to advertise new data among nodes
  - A network node that has data to share can advertise its data
  - Transmitting an ADV message containing the metadata describing the data.
- *REQ:*
  - Used to request an advertised data of interest
  - Upon receiving an ADV containing metadata a network node interested in receiving specific data sends a REQ message to the metadata advertising node
- *DATA:*
  - Contains the actual data collected by a sensor, along with a metadata header
  - Message is typically larger than the ADV and REQ messages.

NETWORK LAYER FUNCTIONS/RESPOSIBILITIES

Sensor Protocols for Information via Negotiation (SPIN)



NETWORK LAYER FUNCTIONS/RESPOSIBILITIES

Sensor Protocols for Information via Negotiation (SPIN)

SPIN Protocols

- SPIN-PP (SPIN Point-to-Point)
  - a 3-stage handshake protocol for Point-to-Point media.

- SPIN-EC (SPIN Energy Conservation)
  - SPIN-PP with a low-energy threshold for Energy Conservation.

- SPIN-BC (SPIN Broadcast)
  - 3-stage handshake protocol for Broadcast media.

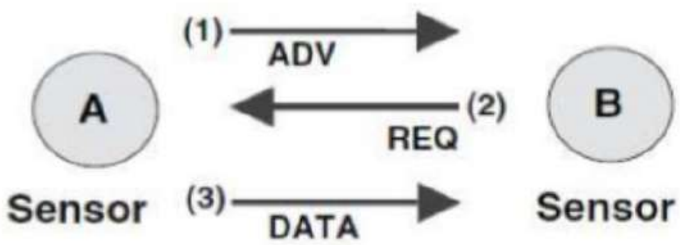
- SPIN-RL (SPIN Reliability)
  - SPIN-BC for lossy networks.

NETWORK LAYER FUNCTIONS/RESPOSIBILITIES

Sensor Protocols for Information via Negotiation (SPIN)

SPIN-PP (SPIN Point-to-Point)

- The simplest version of SPIN referred to as point –to- point comm. network.
- The three-step handshake protocol used by SPIN-PP
  1. The node holding the data, node A, issues an advertisement packet (ADV)
  2. Node B expresses interest in receiving the data by issuing a data request (REQ)
  3. Node A responds to the request and sends a data packet to node B (DATA)

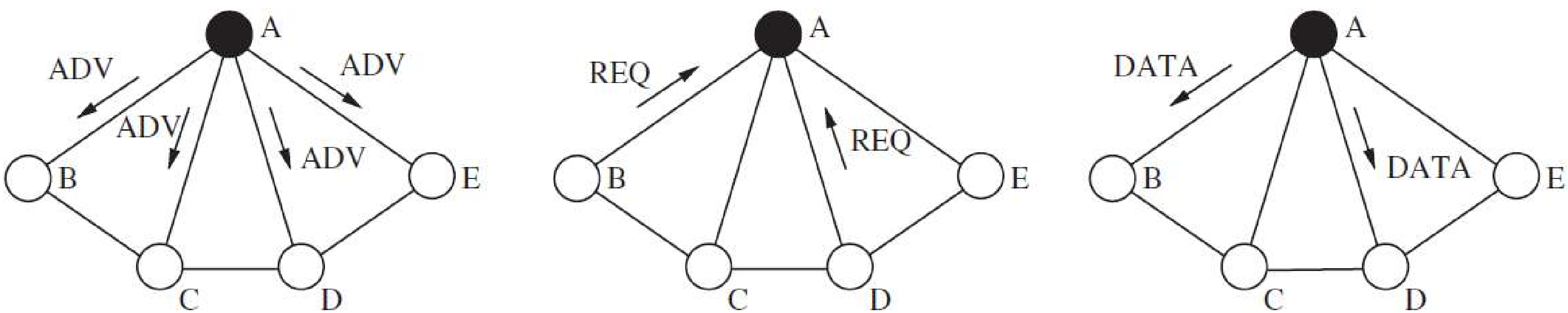


- SPIN-PP uses negotiation to overcome the implosion and overlap.
- The protocol also achieves high data dissemination rates.

NETWORK LAYER FUNCTIONS/RESPOSIBILITIES

Sensor Protocols for Information via Negotiation (SPIN)
SPIN-PP (SPIN Point-to-Point)

- Flooding occurs using a 3-way handshake:
  - Node A advertises newly available data using ADV (advertisement) message (containing data's meta-data)
  - Neighbor nodes (B, C, D, E) check if advertised data needed and if so, respond with REQ (request for data) message
  - A sends DATA message to nodes that responded (i.e., B and D)



(a) advertisement phase (b) request phase (c) data transmission

NETWORK LAYER FUNCTIONS/RESPOSIBILITIES

Sensor Protocols for Information via Negotiation (SPIN)
SPIN EC(SPIN Energy Conservation)

- SPIN-EC designed for point-to-point communication.
- Adds simple heuristic to protocol to add energy conservation.
- As long as energy sufficient, node participates in 3-way handshake.
- Incorporates a threshold based resource-awareness mechanism to complete data negotiation.
- Nodes does not participate if it believes that this will reduce its energy below a certain low-energy threshold.
- If a node receives an advertisement, it does not send out an REQ message if its energy resource is not high enough to transmit an REQ message and to receive data.
- Node initiates handshake only if it has sufficient energy to send DATA to all neighbors.

NETWORK LAYER FUNCTIONS/RESPOSIBILITIES

Sensor Protocols for Information via Negotiation (SPIN)

SPIN BC (SPIN Broadcast)

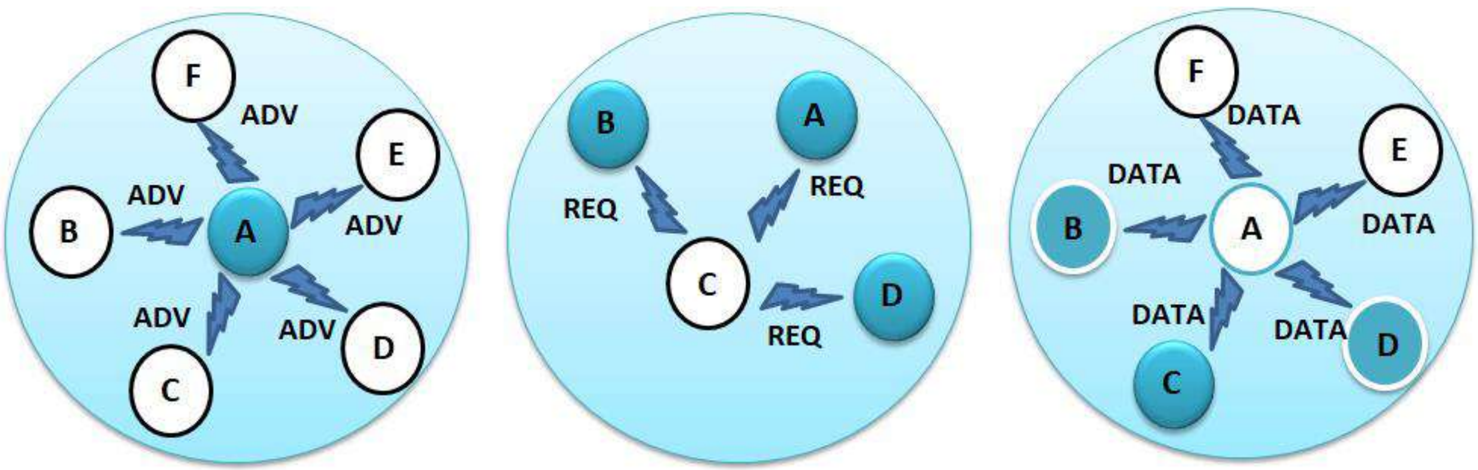
- Designed for broadcast networks.
- The packet transmitted by a node is received by all the other nodes within a certain range of the sending node.
- Node which has received an ADV message does not respond immediately with an REQ message.
- Node waits for a certain amount of time and Monitors the communications channel.
- If the node hears an REQ message issued by another node which is interested in receiving the data, it cancels its own request(eliminating any redundant requests for the same message).
- The advertising node sends the data message only once, even when it receives multiple requests for the same message.

NETWORK LAYER FUNCTIONS/RESPOSIBILITIES

Sensor Protocols for Information via Negotiation (SPIN)

SPIN BC (SPIN Broadcast)

- SPIN-BC protocol basic operations
  - A sends a ADV packet to advertise the data to its neighbors
  - All nodes hear the advertisement, but node C is first to issue a REQ packet (E & F not interested)
  - Nodes B and D hear the broadcast request and refrain from issuing their own REQ packets
  - Upon hearing node C's request, node A replies by sending the data packet.
  - All nodes within the transmission range of A receive the data packet, including nodes E and.



NETWORK LAYER FUNCTIONS/RESPOSIBILITIES

Sensor Protocols for Information via Negotiation (SPIN)
SPIN-RL (SPIN Reliability)

- Extends the capabilities of SPIN-BC to enhance
  - Reliability
  - Overcome message transmission errors by a lossy channel.

- Reliability is achieved by periodic broadcasting of ADV and REQ messages.

- Each node in SPIN-BC keeps track of the advertisements it hears and the nodes where these advertisements originate.

- If a node requesting specific data of interest does not receive the data requested within a certain period of time, it sends the request again.

- Improved reliability by re-advertising metadata periodically.

- SPIN-RL nodes limit the frequency of resend the data messages.

- After sending out a data message, a node waits for a certain time period before it responds to other requests for the same data message.

NETWORK LAYER FUNCTIONS/RESPOSIBILITIES

Sensor Protocols for Information via Negotiation (SPIN)

SPIN-RL (SPIN Reliability)

- The SPIN-RL protocol incorporates two adjustments to SPIN-BC to achieve reliability.

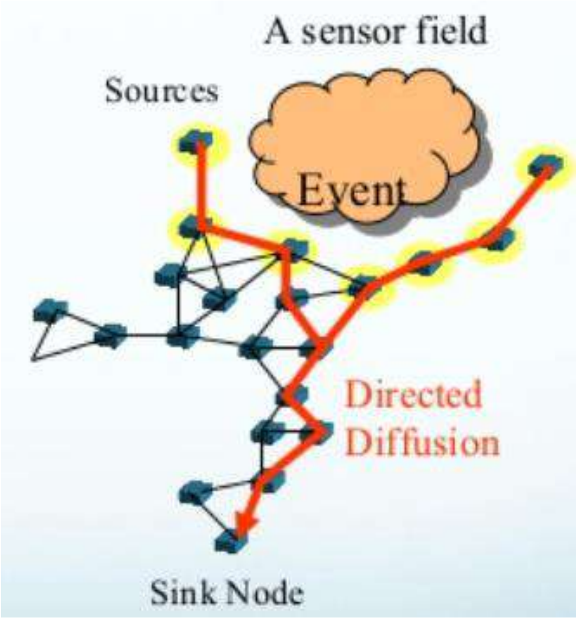
1. First, each SPIN-RL node keeps track of which advertisements it hears from which nodes, and if it does not receive the data within a reasonable period of time following a request, the node re-requests the data. It fills out the originating-advertiser field in the header of the REQ message with a destination, randomly picked from the list of neighbors that had advertised that specific piece of data.

2. Second, SPINRL nodes limit the frequency with which they will resend data. If a SPIN-RL node sends out a DATA message corresponding to a specific piece of data, it will wait a predetermined amount of time before responding to any more requests for that piece of data.

NETWORK LAYER FUNCTIONS/RESPOSIBILITIES

Directed Diffusion (DD)

- The Problem
  - Where should the data be stored?
  - How should queries be routed to the stored data?
  - How should queries for sensor networks be expressed?
  - Where and how should aggregation be performed?



NETWORK LAYER FUNCTIONS/RESPOSIBILITIES

Directed Diffusion (DD)

- Directed diffusion is data-centric.
- Useful where the sensor nodes themselves generate requests/queries for data.
- Data generated by sensor nodes is named by attribute-value pairs.
- A sink node requests data by sending interests for named data.
- Data matching the interest is then "drawn" down toward that node.
- Intermediate nodes can cache, or transform data and may direct interests based on previously cached data Section.
- The human operator's query would be transformed into an interest that is diffused (e.g., broadcasted, geographically routed) toward nodes in regions X or Y.
- When a node in that region receives an interest, it activates its sensors which begin collecting information about pedestrians.

NETWORK LAYER FUNCTIONS/RESPOSIBILITIES

- When the sensors report the presence of pedestrians, this information returns along the reverse path of interest propagation.

- Intermediate nodes might aggregate the data, e.g., more accurately pinpoint the pedestrian's location by combining reports from several sensors.

- An important feature of directed diffusion is that interest and data propagation and aggregation are determined by localized interactions (message exchanges between neighbors or nodes within some vicinity).

NETWORK LAYER FUNCTIONS/RESPOSIBILITIES

Interest and Event Naming

- Sensor nodes are monitoring animals
- Users are interested in receiving data for all 4-legged creatures seen in a rectangle
- Users specify the data rate
- **Query/interest:**
  - Type=four-legged
  - animal Interval=20 ms (event data rate)
  - Duration=10 seconds (time to cache)
  - Rect=[-100, 200, 400]
- **Reply:**
  - Type=four-legged
  - animal Instance = elephant
  - Location = [125, 220]
  - Intensity = 0. 6
  - Confidence = 0. 85
  - Timestamp = 01: 20: 40
- Attribute-Value pairs, no advanced naming scheme

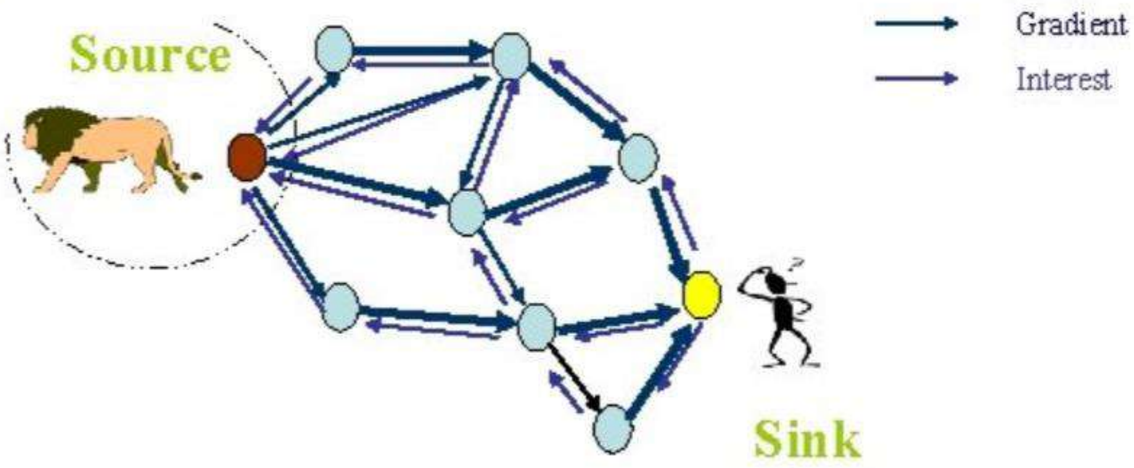
NETWORK LAYER FUNCTIONS/RESPOSIBILITIES

Directed Diffusion (DD)

- Directed diffusion consists of several elements: **interests, data messages, gradients, and reinforcements.**

- An interest message is a query or an interrogation which specifies what a user wants. Each interest contains a description of a sensing task that is supported by a sensor network for acquiring data.

- **Interest Propagation**
  - Flood interest
  - Constrained or Directional flooding based on location is possible
  - Directional propagation based on previously cached data Gradient Source Interest Sink

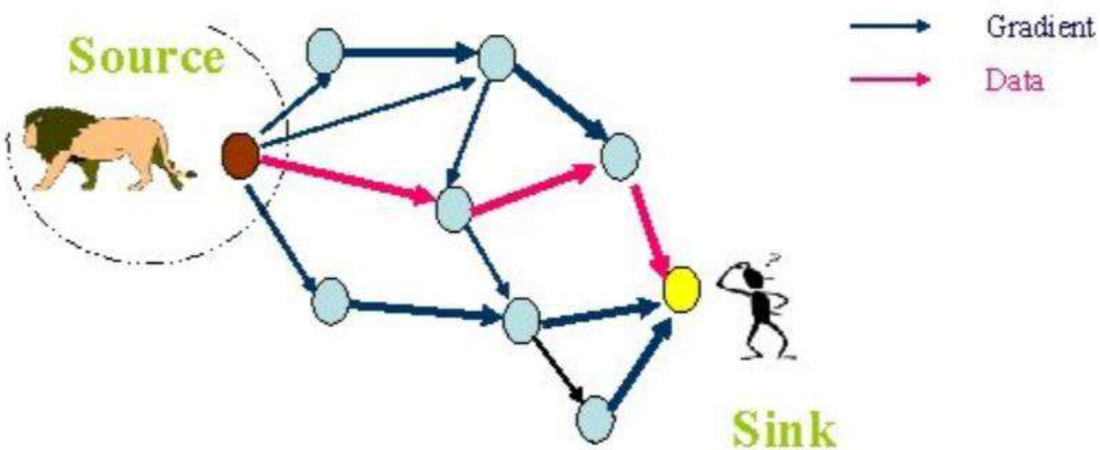


NETWORK LAYER FUNCTIONS/RESPOSIBILITIES

Directed Diffusion (DD)

- Typically, data in sensor networks is the collected or processed information of a physical phenomenon. Such data can be an event, which is a short description of the sensed phenomenon.
    - In directed diffusion, data is named using attribute-value pairs.
    - A sensing task (or a subtask thereof) is disseminated throughout the sensor network as an interest for named data.

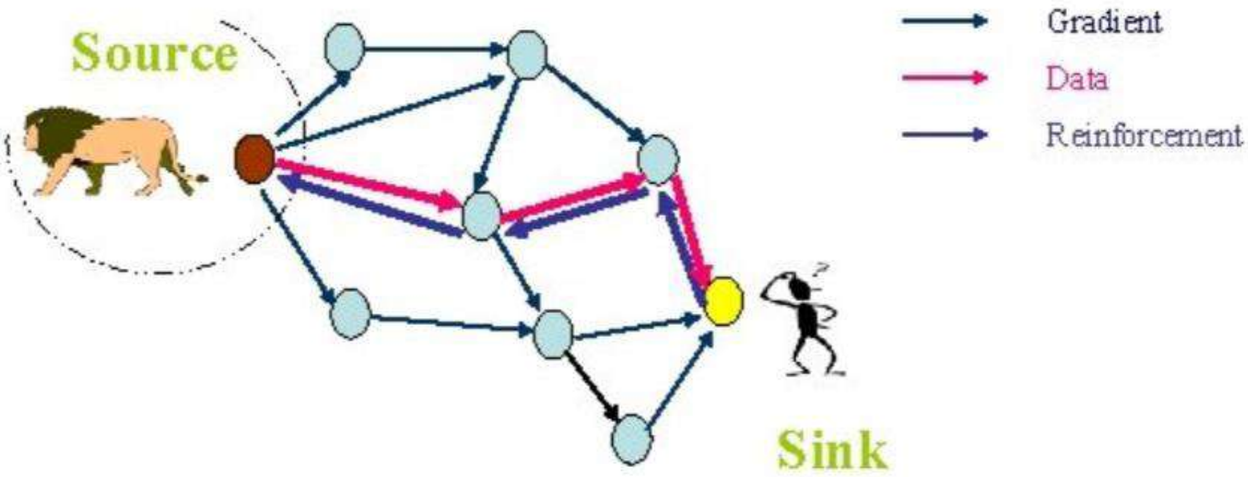
- This dissemination sets up gradients within the network designed to "draw" events (i.e., data matching the interest).
    - Specifically, a gradient is direction state created in each node that receives an interest. The gradient direction is set toward the neighboring node from which the interest is received.

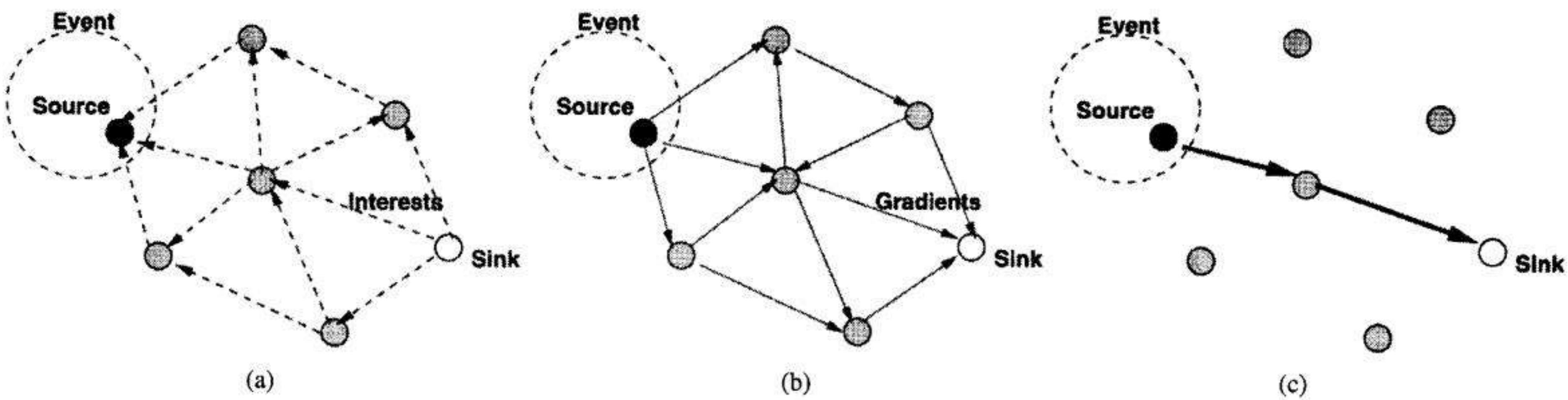


NETWORK LAYER FUNCTIONS/RESPOSIBILITIES

Directed Diffusion (DD)

- Events start flowing toward the originators of interests along multiple gradient paths. The sensor network reinforces one or a small number of these paths.





Simplified schematic for directed diffusion. (a) Interest propagation. (b) Initial gradients setup. (c) Data delivery along reinforced path.

NETWORK LAYER FUNCTIONS/RESPOSIBILITIES

Rumor Routing

- Rumor routing is a kind of directed diffusion and is used for applications where geographic routing is not feasible.

- While the classic flooding approach can be described as *event flooding* (i.e., a source propagates its sensor data via an event throughout the network), *query flooding* describes the process used to propagate queries to all nodes in the network when no localization information is available to steer the query toward the appropriate sensors.

- It combines query flooding and event flooding protocols in a random way. It has the following assumptions:
  - The network is composed of densely distributed nodes.
  - Only bi-directional links exits.
  - Only short distance transmissions are allowed.
  - It has fixed infrastructure.

- In case of directed diffusion flooding is used to inject the query to the entire network.

NETWORK LAYER FUNCTIONS/RESPOSIBILITIES

Rumor Routing

- Sometimes the requested data from the nodes are very small and thus the flooding is unnecessary, so we can use another approach which is to flood the events when the number of events is small and the number of queries is large. The queries are rooted to that particular nodes that are belongs to the interested region.
- In order to flood events through the network, the rumor routing algorithm employs long-lived packets, called agents.
- When a node detects an event, it adds such event to its local table (events table), and generates an agent.
- Agents travel the network on a random path with related event information. Then the visited nodes form a gradient towards the event.
- When a node needs to initiate a query, it routes the query to the initial source.
- If it gets some nodes lying on the gradient before its TTL expires, it will be routed to the event, else the node may need to retransmit, give up or flood the query.

NETWORK LAYER FUNCTIONS/RESPOSIBILITIES

Rumor Routing

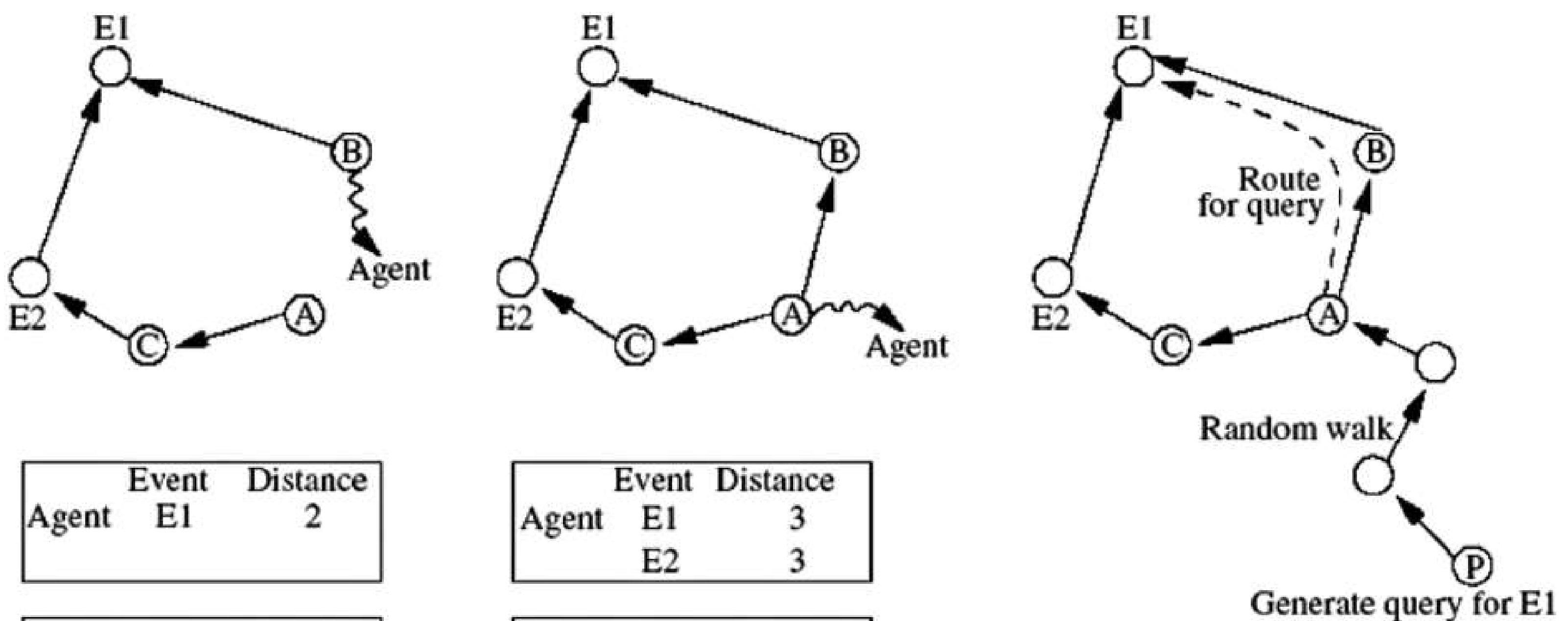
- Rumor routing is an agent-based path creation algorithm. Agents, or “ants,” are long-lived entities created at random by nodes. These are basically packets which are circulated in the network to establish shortest paths to events that they encounter. They can also perform path optimizations at nodes that they visit.

- When an agent finds a node whose path to an event is longer than its own, it updates the node’s routing table. Figure illustrates the working of the rumor routing algorithm.

NETWORK LAYER FUNCTIONS/RESPOSIBILITIES

Rumor Routing

- In Figure 12.4 (a), the agent has initially recorded a path of distance 2 to event **E**1. Node A 's table shows that it is at a distance 3 from event **E**1 and a distance 2 from **E**2. When the agent visits node **A** , it updates its own path state information to include the path to event **E**2.



	Event	Distance
Agent	E1	2

Event	Distance	Direction
E1	3	C
E2	2	C

Table at node A

(a)

	Event	Distance
Agent	E1	3
	E2	3

Event	Distance	Direction
E1	2	B
E2	2	C

Table at node A

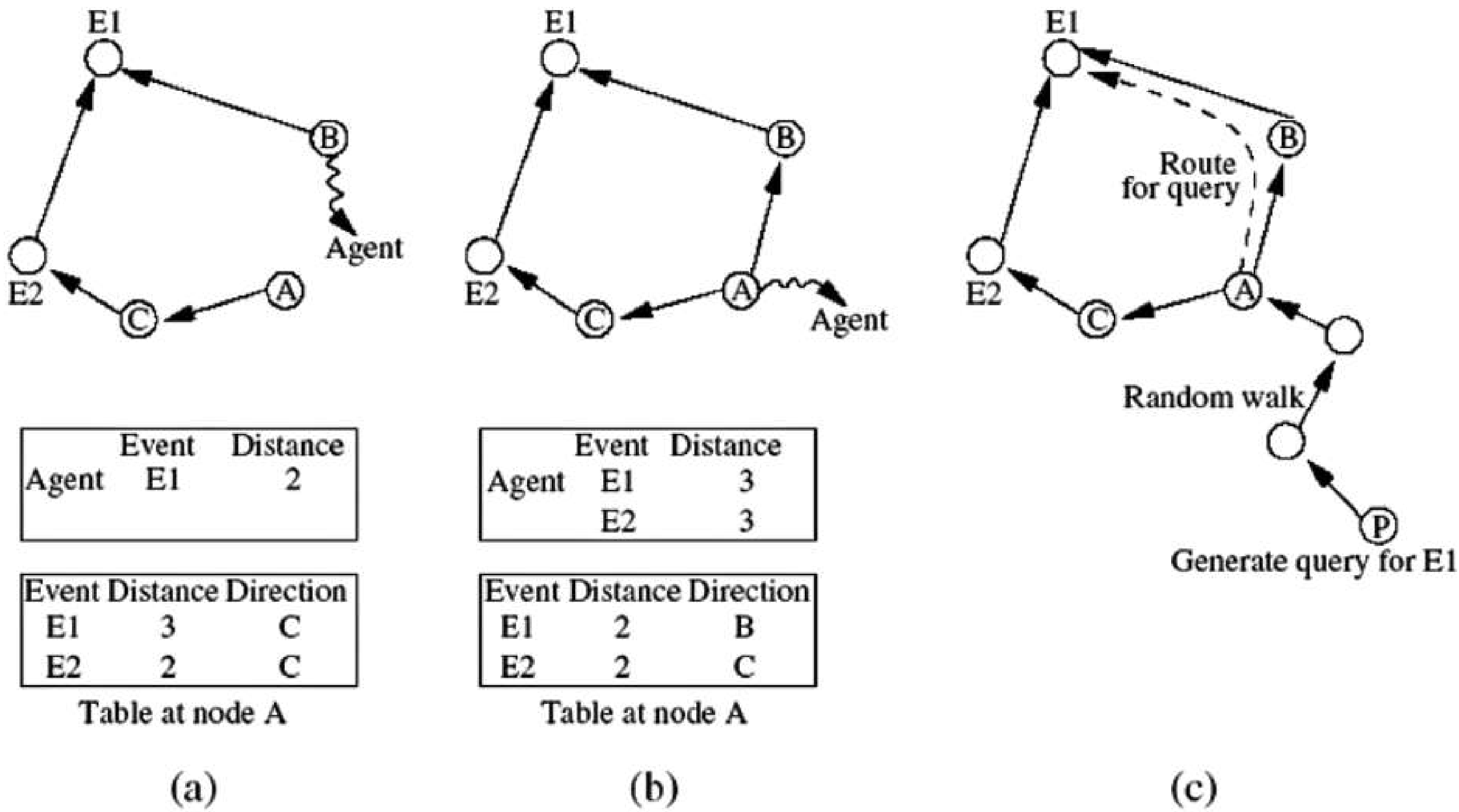
(b)

(c)

NETWORK LAYER FUNCTIONS/RESPOSIBILITIES

Rumor Routing

- The updating is with one hop greater distance than what it found in **A**, to account for the hop between any neighbor of **A** that the agent will visit next, and **A**. It also optimizes the path to **E**1 recorded at node **A** to the shorter path through node **B** . The updated status of the agent and node table is shown in Figure 12.4 (b).



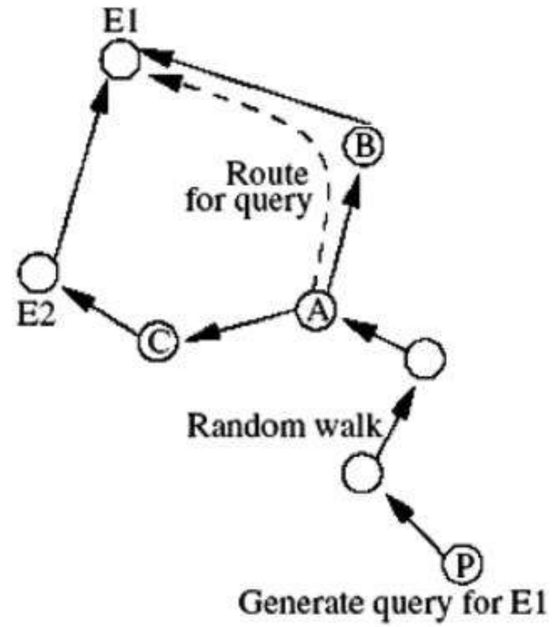
NETWORK LAYER FUNCTIONS/RESPOSIBILITIES

Rumor Routing

- When a query is generated at a sink, it is sent on a random walk with the hope that it will find a path (pre-established by an agent) leading to the required event.

- This is based on the high probability of two straight lines intersecting on a planar graph, assuming the network topology is like a planar graph, and the paths established can be approximated by straight lines owing to high density of the nodes. If a query does not find an event path, the sink times out and uses flooding as a last resort to propagate the query.

- For instance, as in Figure 12.4 (c), suppose a query for event **E1** is generated by node **P** . Through a random walk, it reaches **A** , where it finds the previously established path to **E1**. Hence, the query is directed to **E1** through node **B** , as indicated by **A** 's table.



(c)

NETWORK LAYER FUNCTIONS/RESPOSIBILITIES

Rumor Routing

- Unlike directed diffusion, where data can be routed through multiple paths at low rates, Rumor routing only maintains one path between source and destination.

- Rumor routing performs well only when the number of events is small.

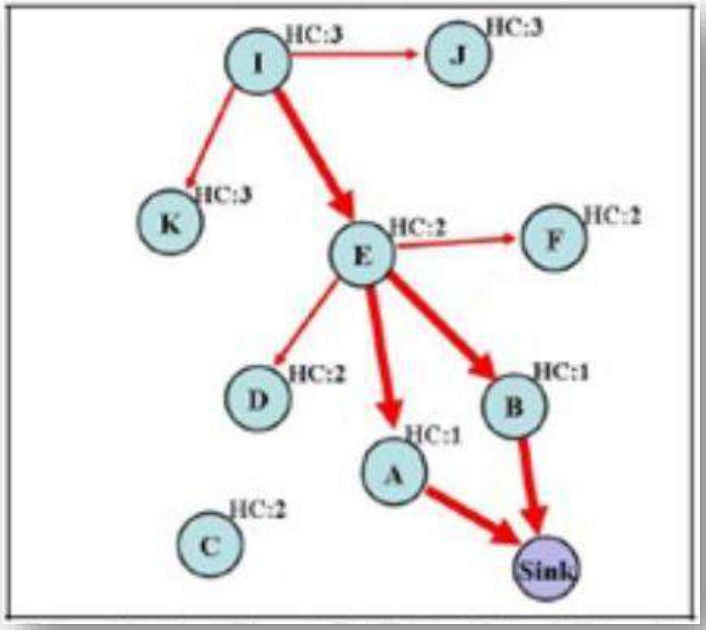
- For a large number of events, the cost of maintaining agents and event-tables in each node becomes infeasible if there is not enough interest in these events from the BS.

- Moreover, the overhead associated with rumor routing is controlled by different parameters used in the algorithm such as time-to-live (TTL) pertaining to queries and agents.

NETWORK LAYER FUNCTIONS/RESPOSIBILITIES

Gradient Based Routing

- Another variant of directed diffusion.
- Gradient is determined on the basis of the number of hops to the sink
    - A sink floods interest message
    - Each interest message records number of hops traveled
    - Therefore, a node can determine its distance (called height) to sink
    - Gradient is difference between a node's height and height of neighbor
    - Packet is forwarded on the link with the largest gradient



NETWORK LAYER FUNCTIONS/RESPOSIBILITIES

Gradient Based Routing

- Nodes can establish a Data Combining Entity (DCE), which is responsible for data compaction.

- Nodes can use a traffic spreading technique to balance traffic over network
  - ***Stochastic Scheme***: node selects next hop randomly when there are two or more with same gradient
  - ***Energy-based scheme***: node increases its height when its energy level is low
  - ***Stream-based scheme***: new streams are diverted away from nodes that already serve other streams (e.g., by reporting increased heights)

NETWORK LAYER FUNCTIONS/RESPOSIBILITIES

Proactive Routing

- Routes are established before they are actually needed
- Also called table-driven routing
- Main advantage: routes can be used immediately when needed (table look up for next-hop neighbor)
- Main disadvantages:
  - Establishing and maintaining routes that are infrequently (or never) Needed.
  - Routing tables can become very large.
  - Stale information in tables can lead to routing errors.
- ***Examples are:***
  - *DSDV (*Destination-Sequenced Distance Vector)
  - *OLSR (Optimized Link State Routing)*

NETWORK LAYER FUNCTIONS/RESPOSIBILITIES

DSDV (Destination-Sequenced Distance Vector)

- DSDV is a modified version of class Distributed Bellman-Ford algorithm.

- Concept of Distance-Vector Algorithms:

○ Node i maintains a list of distances $\{d_{ij}x\}$ for each destination x via node j.

○ Node i selects k as next-hop if $d_{ik}x = min\ \{d_{ij}x\}$.

○ Distances stored in routing table, along with sequence number for each entry (assigned by destination node)

  - ➢ purpose of sequence number is to identify stale routes

○ Nodes broadcast their routing tables to their neighbors periodically and whenever significant information is available

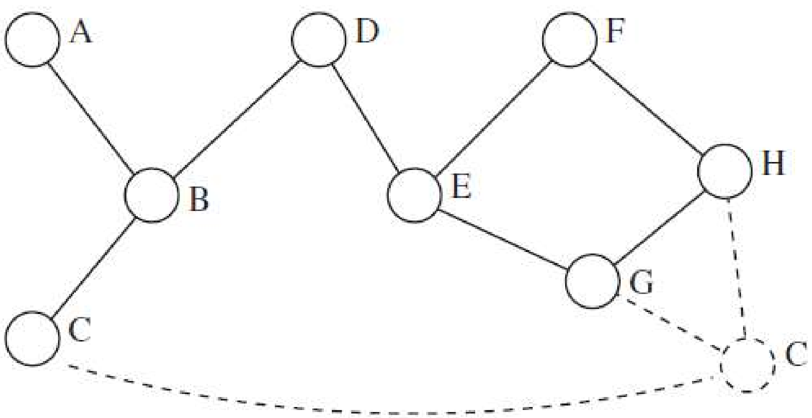
  - ➢ Full dump packet: contains entire routing table
  - ➢ Incremental packet: contains only changed table entries
  - ➢ A receiving node updates its table if the received information has a more recent sequence number or if sequence number is identical, but new route is shorter.

NETWORK LAYER FUNCTIONS/RESPOSIBILITIES

DSDV (Destination-Sequenced Distance Vector)

- Figure shows a possible network topology, indicating node locations and connectivity, including node D's routing table (first table).

- Suppose that node C moves from its current location to a new location in the vicinity of nodes H and G, which become node C's new neighbors. Update packets from D's neighbors will ultimately inform D that the route to C via B is invalid and that a new route to C via node E exists. Therefore, node D will replace C's information in its routing table to show E as the next hop neighbor and to reflect the new distance of three



Destination	NextHop	Distance
A	B	2
B	B	1
C	B	2
D	D	0
E	E	1
F	E	2
G	E	2
H	E	3

Node D's Table Before C's Move

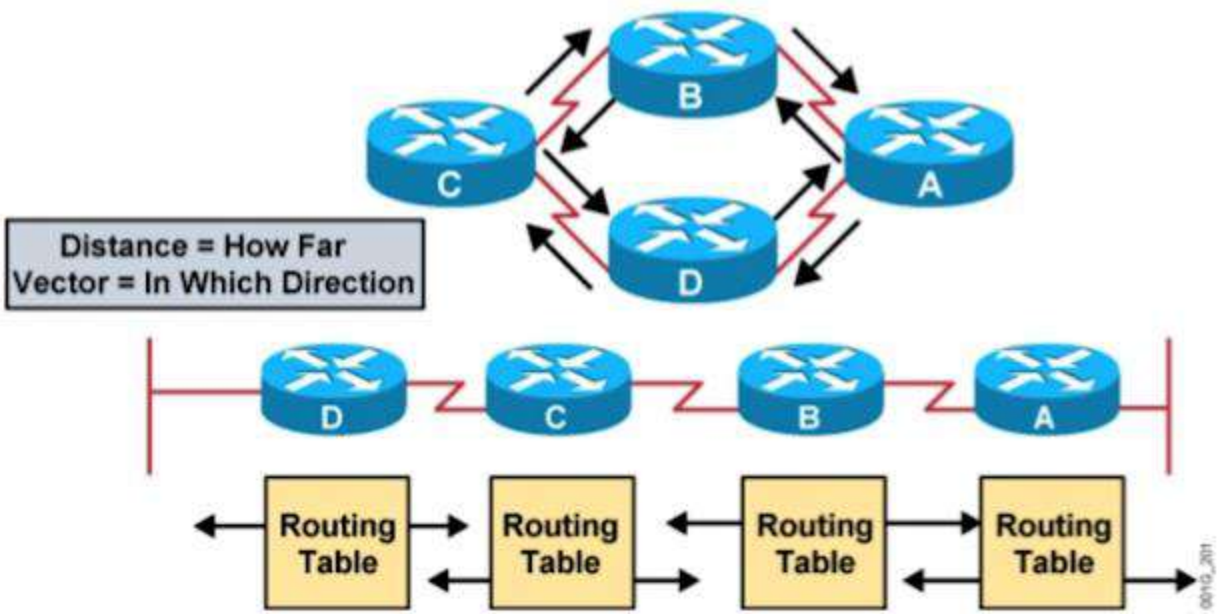
Destination	NextHop	Distance
A	B	2
B	B	1
C	E	3
D	D	0
E	E	1
F	E	2
G	E	2
H	E	3

Node D's Table After C's Move

NETWORK LAYER FUNCTIONS/RESPOSIBILITIES

Distance Vector Routing

- Each node knows the distance (=cost) to its directly connected neighbors.
- A node sends periodically a list of routing updates to its neighbors.
- If all nodes update their distances, the routing tables eventually converge.
- New nodes advertise themselves to their neighbors.

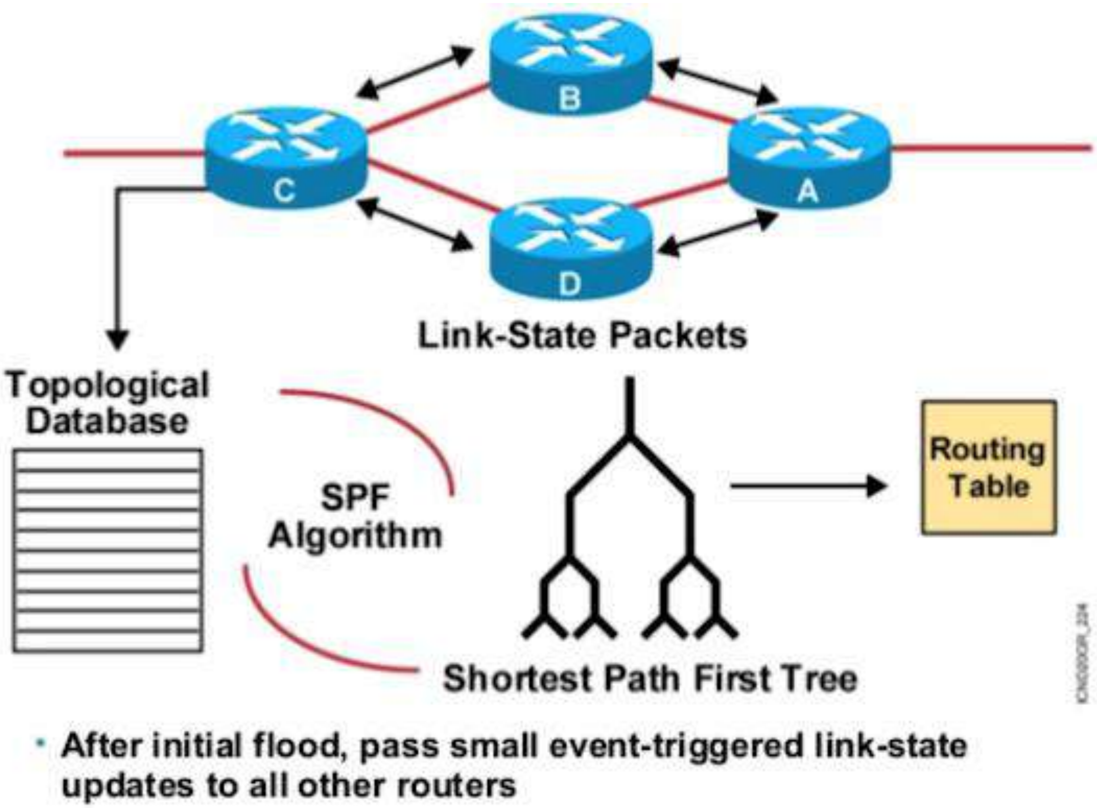


- Routers pass periodic copies of their routing table to neighboring routers and accumulate distance vectors.

NETWORK LAYER FUNCTIONS/RESPOSIBILITIES

Link State Routing

- Each node knows the distance to its neighbors.
- The distance information (=link state) is broadcast to all nodes in the network.
- Each node calculates the routing tables independently.



NETWORK LAYER FUNCTIONS/RESPOSIBILITIES

OLSR (Optimized Link State Routing)

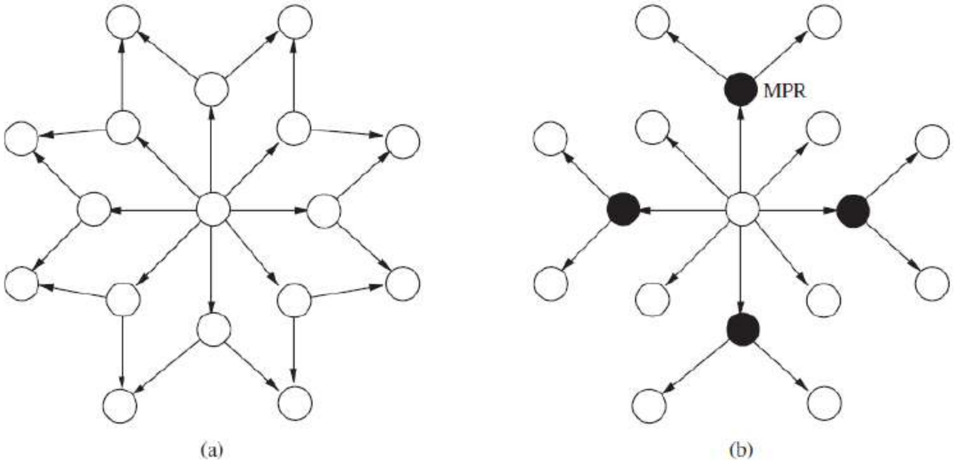
- OLSR is a protocol based on the link state algorithm.
  - Nodes periodically broadcast topological information updates to all other nodes in the network.
  - Allows nodes to obtain complete topological map and to compute paths.

- Nodes use neighbor sensing using HELLO messages:
  - Allows nodes to identify neighbors and to detect changes in neighborhood.
  - HELLO message contains node's identity (address) and list of all known neighbors.
  - If node A can receive node B's HELLO messages, but is not in node B's list of known neighbors, the link B $\rightarrow$ A is asymmetric (otherwise symmetric).
  - Instead of flooding HELLO messages to all nodes in network, OLSR uses multipoint relays (MPRs).

NETWORK LAYER FUNCTIONS/RESPOSIBILITIES

OLSR (Optimized Link State Routing)

- ***Multipoint relays:***
  - A node selects a set of symmetric neighbors as MPRs (MPR selector set); each node can use different algorithm/heuristic for the selection process
    - Example: node determines its 2-hop neighbors (using HELLO messages) and selects minimum set of 1-hop neighbors to reach all 2-hop neighbors.
  - Only MPRs forward messages to other nodes.
  - HELLO messages contain addresses of MPRs, i.e., a node advertises its reachability via MPRs only instead of all neighbors.



Comparison of (a) classic flooding and (b) MPR-based OLSR.

NETWORK LAYER FUNCTIONS/RESPOSIBILITIES

On-Demand (Reactive) Routing

- Routes are not established until actually needed.

- Instead, a source node (knowing the identity/address of a destination node) initiates a route discovery process which completes when at least one route has been found or all possible paths have been examined.

- A newly discovered route is then maintained until it either breaks or is no longer needed by the source

- ***Examples are:***
  - AODV (Ad Hoc On-Demand Distance Vector)
  - DSR (Dynamic Source Routing)

NETWORK LAYER FUNCTIONS/RESPOSIBILITIES

AODV (Ad Hoc On-Demand Distance Vector)

- AODV relies on broadcast route discovery mechanism, which is used to dynamically establish route table entries at intermediate nodes

- Path discovery process:
  - Initiated whenever a source needs to transmit data to a sink and the source does not have an entry for the sink in its routing table.
  - Source broadcasts route request (RREQ) packet containing:
    - Addresses of source and sink
    - Hop count value
    - Broadcast ID (incremented whenever source issues a new RREQ)
    - Two sequence numbers

- Upon receiving RREQ, if a node knows a route to the destination, responds by sending a unicast route reply (RREP) message back to source

- Otherwise RREQs are re-broadcast or duplicate RREQs (identified by source address and broadcast ID) are discarded

NETWORK LAYER FUNCTIONS/RESPOSIBILITIES

AODV (Ad Hoc On-Demand Distance Vector)

- Use of sequence numbers:
  - Every node maintains own sequence numbers.
  - Source's RREQ includes its own sequence number and the most recent sequence number (if known) from the destination.
  - Intermediate nodes only reply to RREQ if the sequence number of their route to the destination is greater than or equal to the destination sequence number in the RREQ.
  - When RREQ is re-broadcast, node records the address of neighbor from which RREQ came (establishing a reverse path towards source).
  - As RREP travels towards source, intermediate nodes set up forward pointers to the node from which the RREP came and records latest destination sequence number.

NETWORK LAYER FUNCTIONS/RESPOSIBILITIES

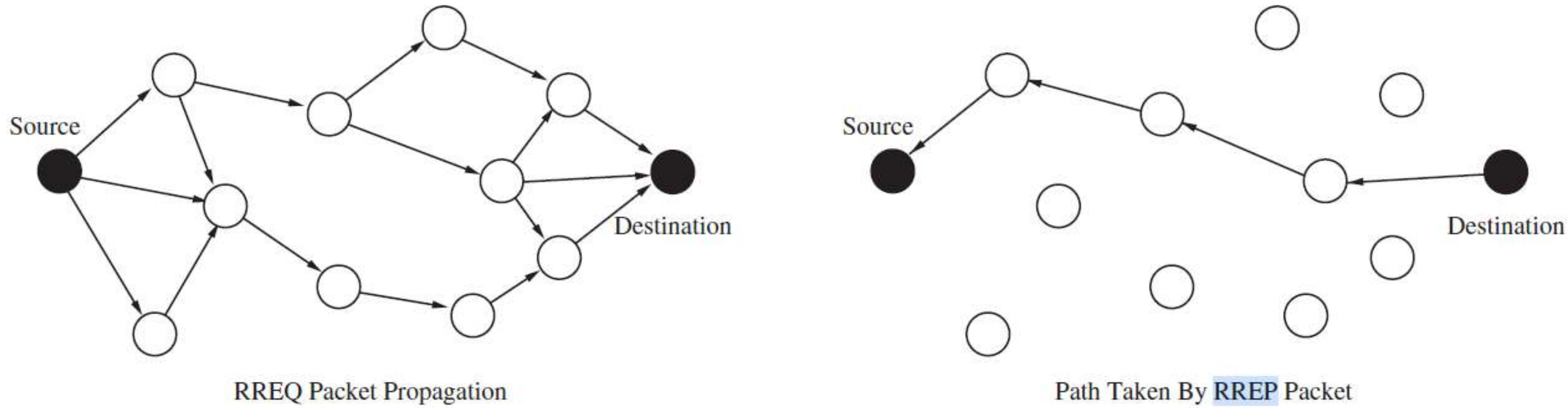
AODV (Ad Hoc On-Demand Distance Vector)

- Use of RREPs:
  - RREP contains:
    - Addresses of source and destination
    - Destination sequence number
    - Hop count
  - Intermediate node propagates RREP only if
    - This is the first copy of this RREP the node sees.
    - Or destination sequence number in RREP is greater than that of previous RREP.
    - Or destination sequence number is the same as in previous RREP, but the hop count is smaller.
  - Limits the number of RREPs in network and ensures that the RREP over the shortest route reaches the source.

NETWORK LAYER FUNCTIONS/RESPOSIBILITIES

AODV (Ad Hoc On-Demand Distance Vector)

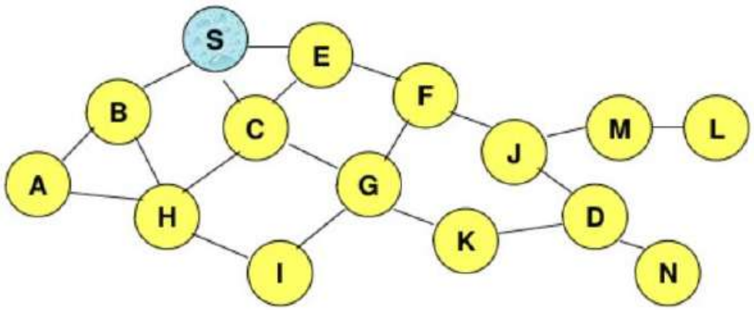
- Routes expire after certain amount of time
  - Link state monitored using HELLO messages among neighbors.
  - When a link/route breaks, the node noticing this issues a route error (RRER) packet towards the source (which can re-initiate route discovery).
- Summary:
  - Routes are established only when needed
  - No need for route table updates and exchanges for unused routes
  - Periodic HELLO messages
  - Initial transmission delay if route must be discovered first
  - RREP uses reverse path of RREQ, i.e., AODV assumes symmetric links



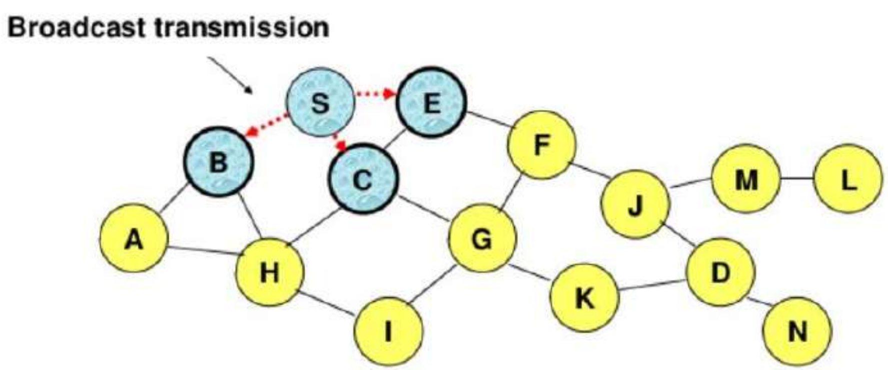
Path discovery process of AODV.

NETWORK LAYER FUNCTIONS/RESPOSIBILITIES

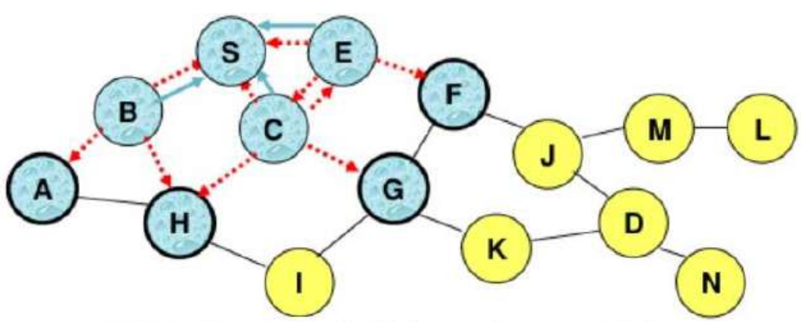
AODV (Ad Hoc On-Demand Distance Vector)



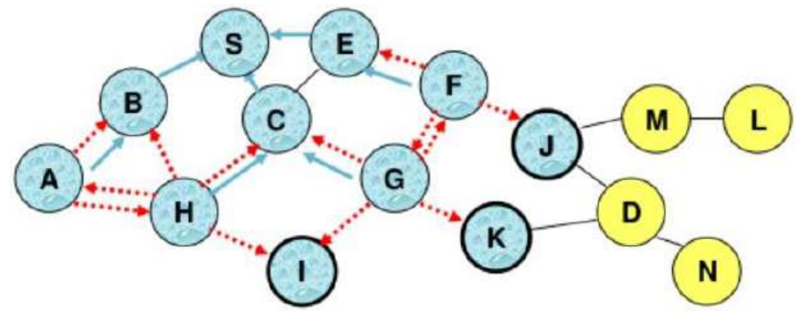
Represents a node that has received RREQ for D from S



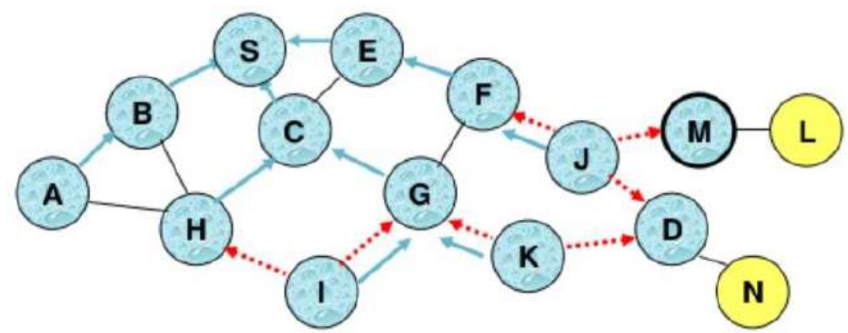
Represents transmission of RREQ

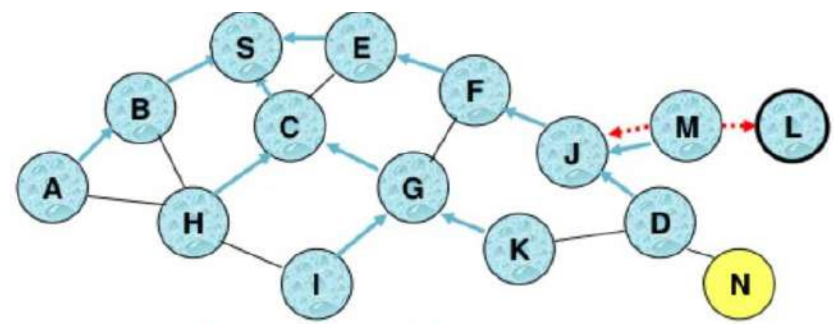


Represents links on Reverse Path



- Node C receives RREQ from G and H, but does not forward it again, because node C has already forwarded RREQ once

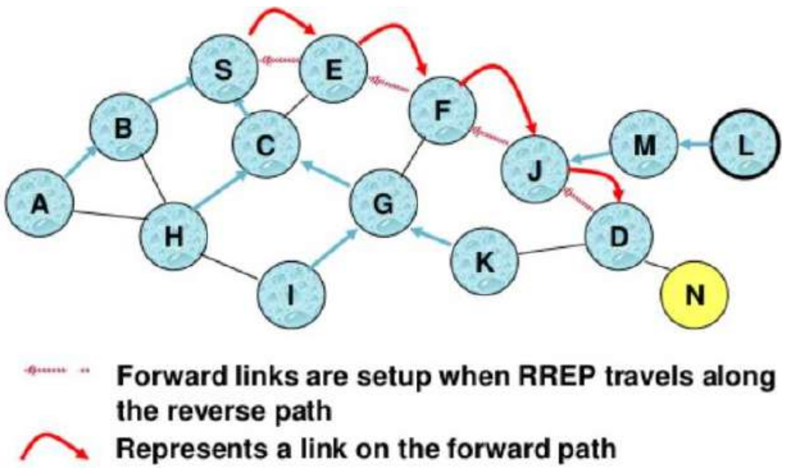


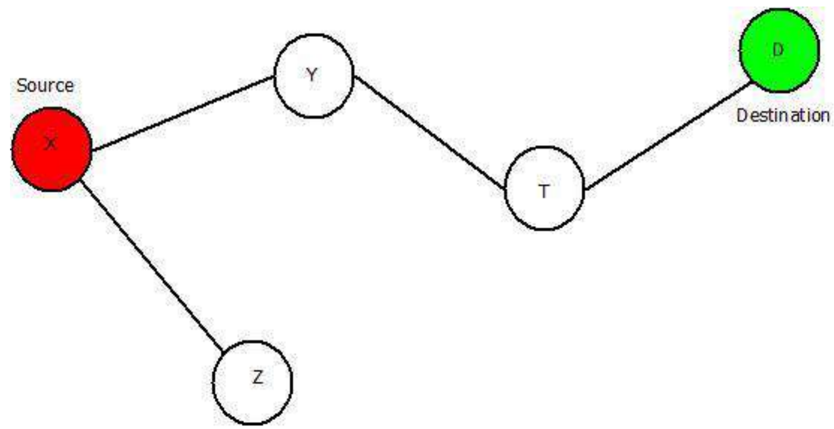


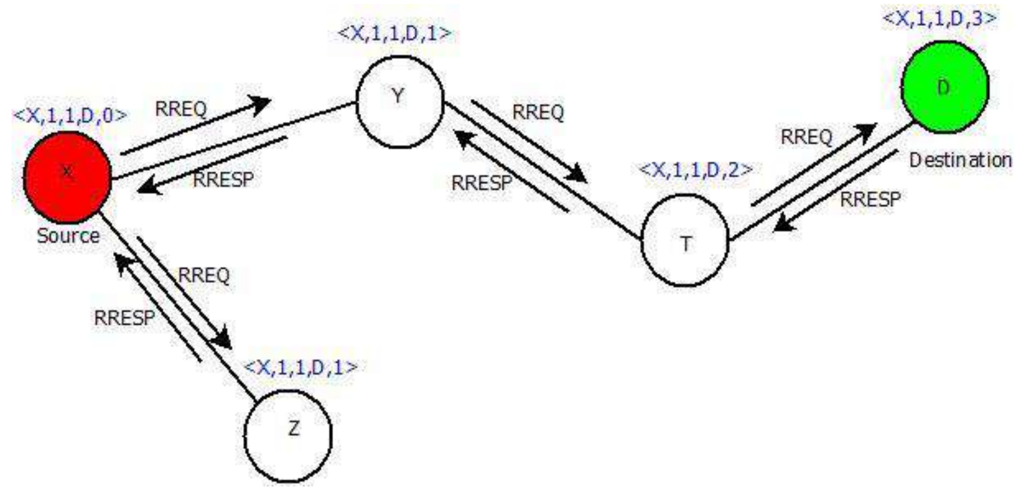
- Node D does not forward RREQ, because node D is the intended target of the RREQ

NETWORK LAYER FUNCTIONS/RESPOSIBILITIES

AODV (Ad Hoc On-Demand Distance Vector)







NETWORK LAYER FUNCTIONS/RESPOSIBILITIES

AODV (Ad Hoc On-Demand Distance Vector)

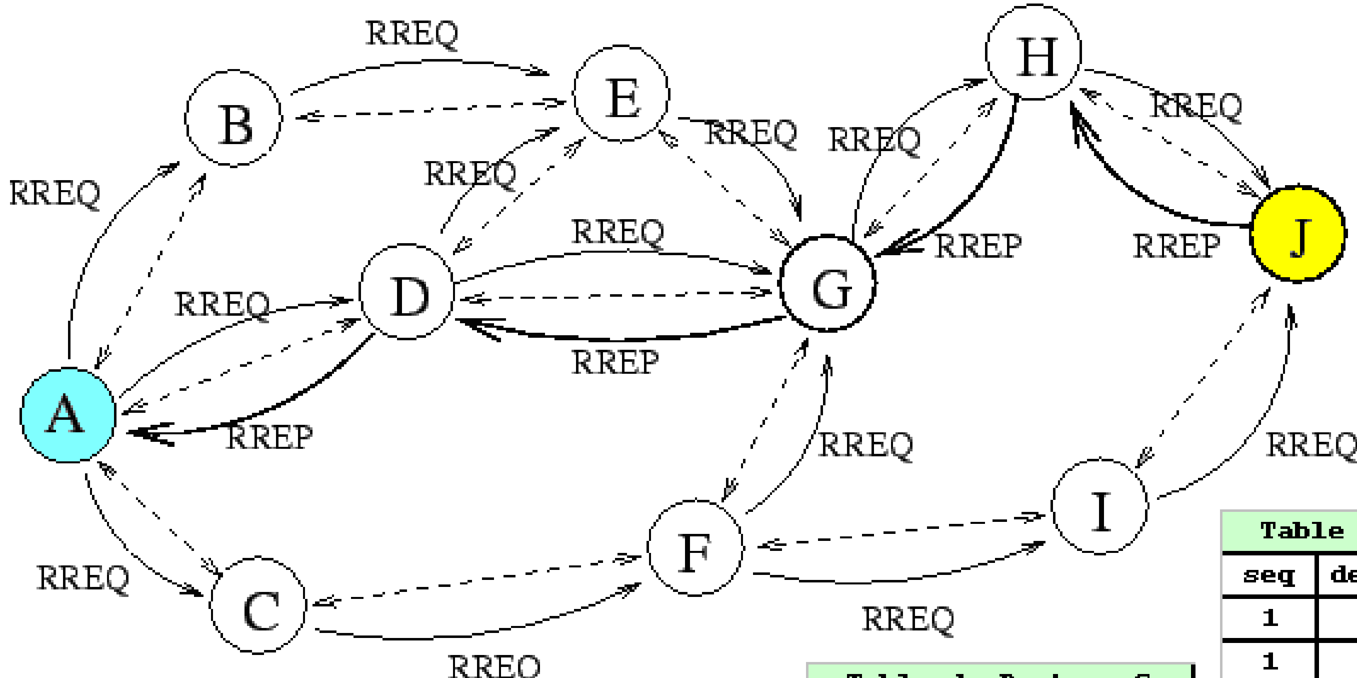


Table de Routage J			
seq	dest	next	hop
1	A	H	4
1	D	H	3
1	G	H	2
1	H	H	1

Table de Routage H			
seq	dest	next	hop
1	A	G	3
1	D	G	2
1	G	G	1
1	J	J	1

Table de Routage A			
seq	dest	next	hop
1	D	D	1
1	G	D	2
1	H	D	3
1	J	D	4

Table de Routage D			
seq	dest	next	hop
1	A	A	1
1	G	G	1
1	H	G	1
1	J	G	2

Table de Routage G			
seq	dest	next	hop
1	D	D	1
1	A	D	2
1	H	H	1
1	J	H	2

NETWORK LAYER FUNCTIONS/RESPOSIBILITIES

Hierarchical Routing

Sensor nodes communicate directly only with a cluster head

- Cluster head:
  - Responsible for propagating sensor data to sink
  - Sometimes more powerful than "regular" nodes
  - Experiences more traffic than "regular" nodes

- Challenges in cluster formation:
  - Selection (election) of cluster heads
  - Selection of cluster to join
  - Adaptation of clusters in response to topology changes, failures, etc.

- Advantages:
  - Potentially fewer collisions (compared to flat routing)
  - Easier duty cycling (energy efficiency)
  - Easier routing process (though routes may be longer)
  - Easier in-network data aggregation

- **Examples are:**
  - LR, LANMAR, LEACH, PEGASIS, SAFARI etc.

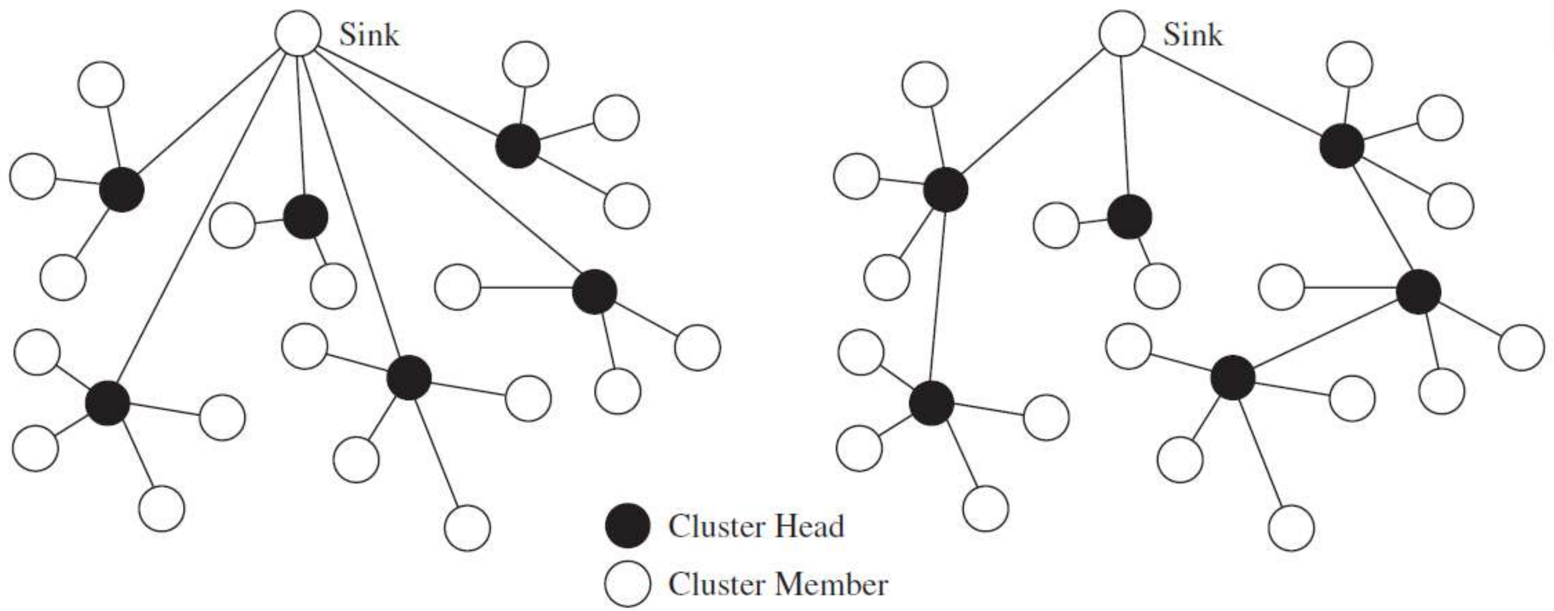
NETWORK LAYER FUNCTIONS/RESPOSIBILITIES

Hierarchical Routing

Variations of Hierarchical Routing

- When all cluster heads communicate directly with the sink node (left graph), the routing challenge is reduced to the ***cluster formation problem***.

- When cluster heads do not directly communicate with the sink (right graph), a ***cluster-based routing protocol*** must also establish multi-hop routes from all cluster heads to the sink.



Clustering with single-hop connections to the sink (left) and clustering with multi-hop connections to the sink (right).

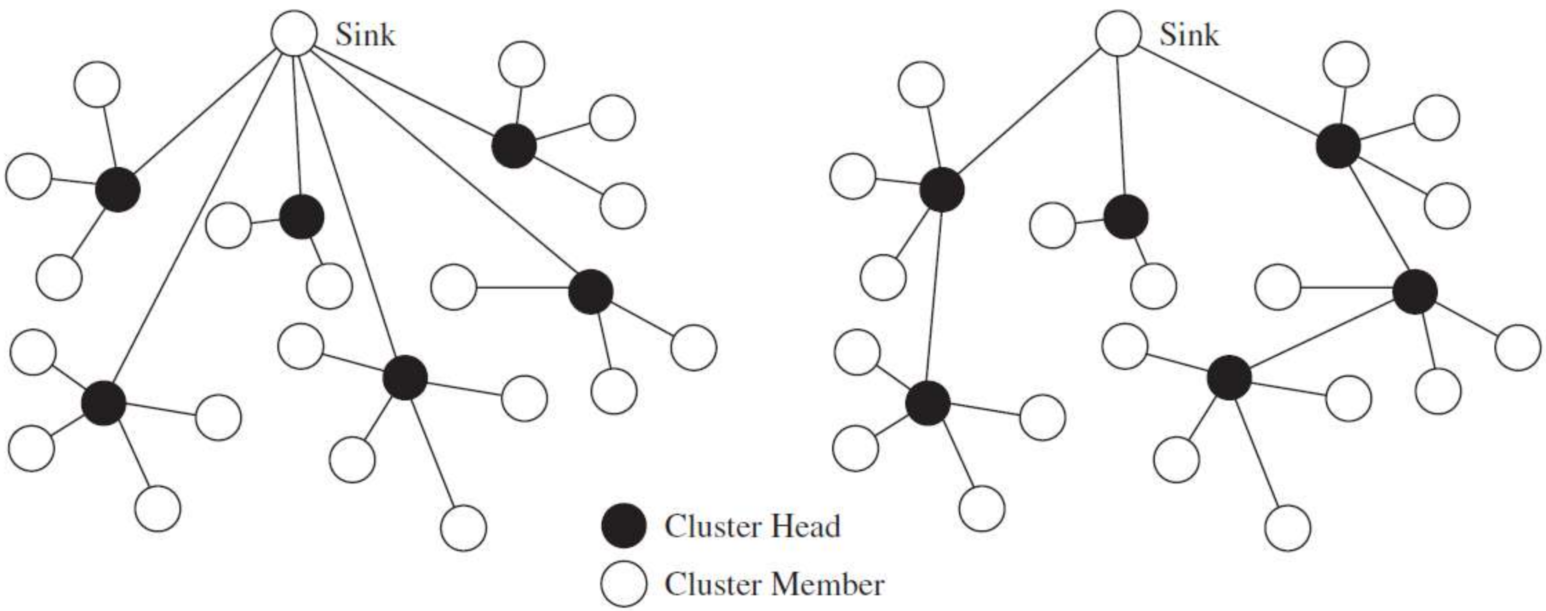
NETWORK LAYER FUNCTIONS/RESPOSIBILITIES

Hierarchical Routing

Variations of Hierarchical Routing

- When all cluster heads communicate directly with the sink node (left graph), the routing challenge is reduced to the ***cluster formation problem***.

- When cluster heads do not directly communicate with the sink (right graph), a ***cluster-based routing protocol*** must also establish multi-hop routes from all cluster heads to the sink.



Clustering with single-hop connections to the sink (left) and clustering with multi-hop connections to the sink (right).

NETWORK LAYER FUNCTIONS/RESPOSIBILITIES

Location-based Routing

- Also referred to as Geographic Routing
- Used when nodes are able to determine their (approximate) positions
- Nodes use location information to make routing decisions
  - Sender must know the locations of itself, the destination, and its neighbors
  - Location information can be queried or obtained from a location broker

- Types of geographic routing:
  - Unicast: single destination
  - Multicast: multiple destinations
  - Geocast: data is propagated to nodes within certain geographic area

- Examples are:
  - **Unicast:** GPSR (Greedy Perimeter Stateless Routing), Forwarding Strategy, Geographic Adaptive Fidelity,
  - **Multicast:** SPBM (Scalable Position-Based Multicast), Geographic Multicast Routing (GMR), Receiver Based Multicast (RBMulticast).
  - **Geocast:** Geographic and Energy Aware Routing (GEAR), Geographic-Forwarding-Perimeter-Geocast.

NETWORK LAYER FUNCTIONS/RESPOSIBILITIES

QoS-Based Routing Protocols

- Protocols that explicitly address one or more Quality-of-Service (QoS) metrics

- Examples of QoS metrics:
  - low (end-to-end) latency/delay
  - low jitter
  - low energy consumption
  - low bandwidth requirements
  - high reliability

- *Examples are:*
  - Sequential Assignment Routing (SAR)
  - SPEED
  - Multipath Multi-SPEED

BASIC ARCHITECTURAL FRAMEWORK

PHYSICAL LAYER

OUTLINES

- Basic Components
- Source Encoding
  - The Efficiency of a Source Encode
  - Pulse Code Modulation and Delta Modulation
- Channel Encoding
  - Types of Channels
  - Information Transmission over a Channel
  - Error Recognition and Correction
- Modulation
  - Modulation Types
  - Quadratic Amplitude Modulation
  - Summary
- Signal Propagation

PHYSICAL LAYER - OVERVIEW

- One of the desirable aspects of WSNs is their ability to communicate over a wireless link, so
  - mobile applications can be supported
  - flexible deployment of nodes is possible
  - the nodes can be placed in areas that are inaccessible to wired nodes

- Once the deployment is carried out, it is possible to
  - rearrange node placement - optimal coverage and connectivity
  - the rearrangement can be made without disrupting the normal operation

- Some formidable *challenges*:
  - limited bandwidth
  - limited transmission range
  - poor packet delivery performance because of interference, attenuation, and multi-path scattering

- therefore, it is vital to understand their properties and some of the mitigation strategies

- In this Chapter, we'll discuss a fundamental introduction to *point-to-point* wireless digital communication

COMMUNICATION SYSTEM – BASIC COMPONENTS

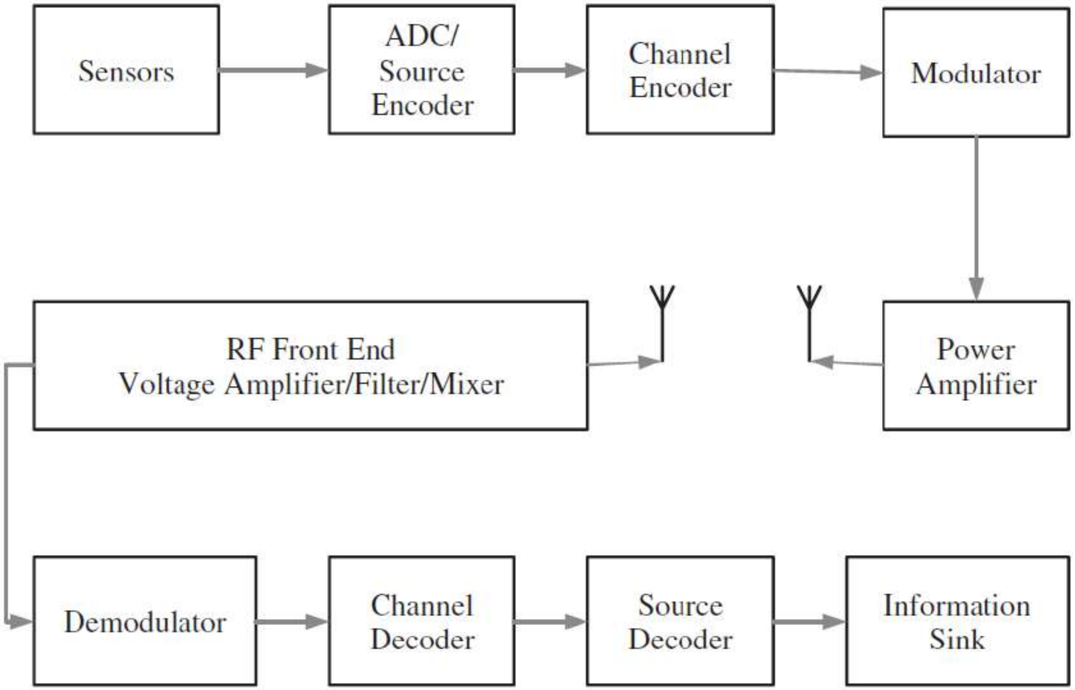
- Communication Systems deals with the flow of some sort of information in some network.

- Information my be electrical signals, words, pictures, music etc.

- The *basic components* of a digital communication system:
  - Transmitter or Source
  - Channel or Transmission Networks, which transfer the communique from transmitter to receiver
  - Receiver or Destination

- In WSN communication, we study the *short range communication* - because nodes are placed close to each other.

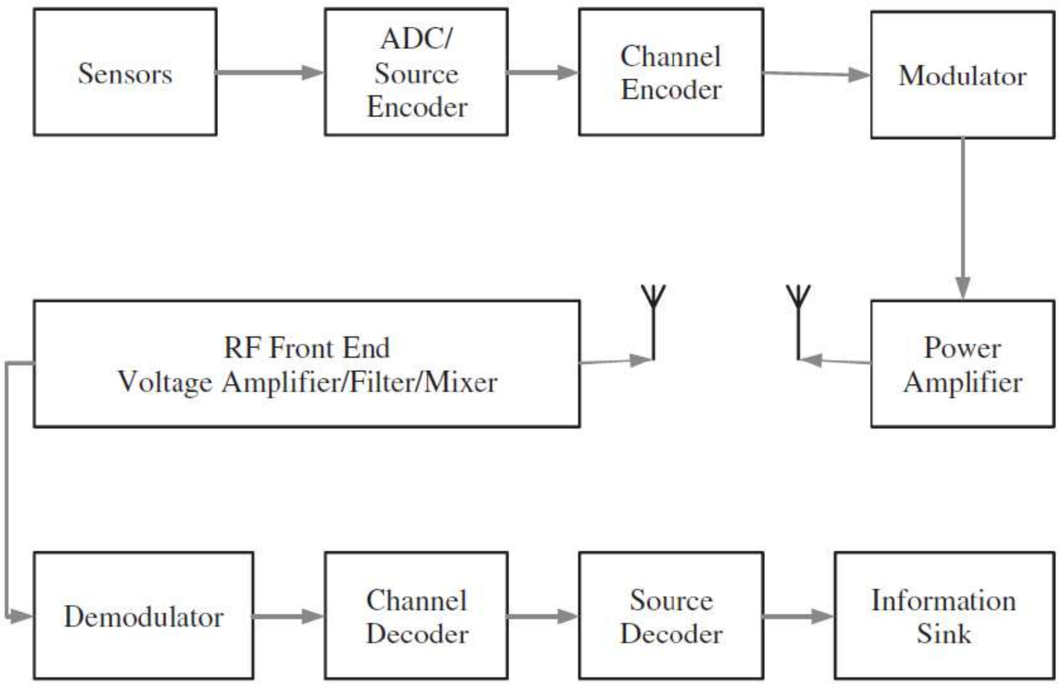
PHYSICAL LAYER – BASIC COMPONENTS



Components of a digital communication system.

- *The* *communication source* represents one or more sensors and produces *a message signal* - *an analog signal*
  - the signal is a *baseband* signal having dominant frequency components *near zero*
  - the message signal has to be converted to a discrete signal (*discrete* both in *time* and *amplitude*)

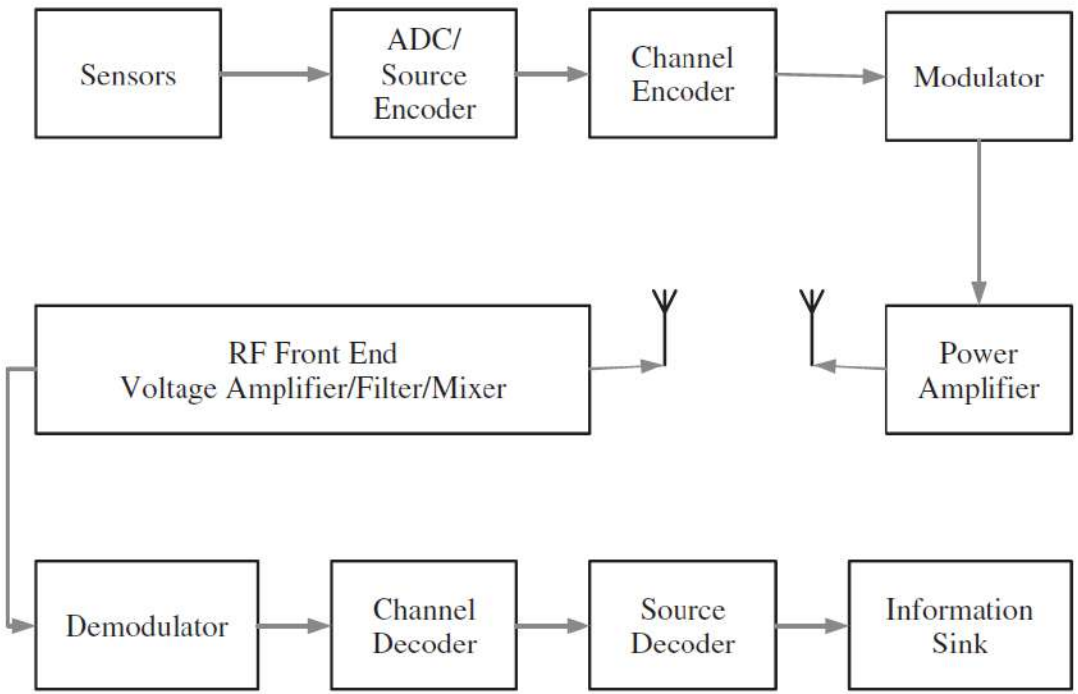
PHYSICAL LAYER – BASIC COMPONENTS



Components of a digital communication system.

- This conversion requires sampling the signal at least at *Nyquist Rate* - no information will be lost
  - The Nyquist rate sets a lower bound on the sampling frequency
  - Hence, the minimum sampling rate should be twice the bandwidth of the signal

PHYSICAL LAYER – BASIC COMPONENTS



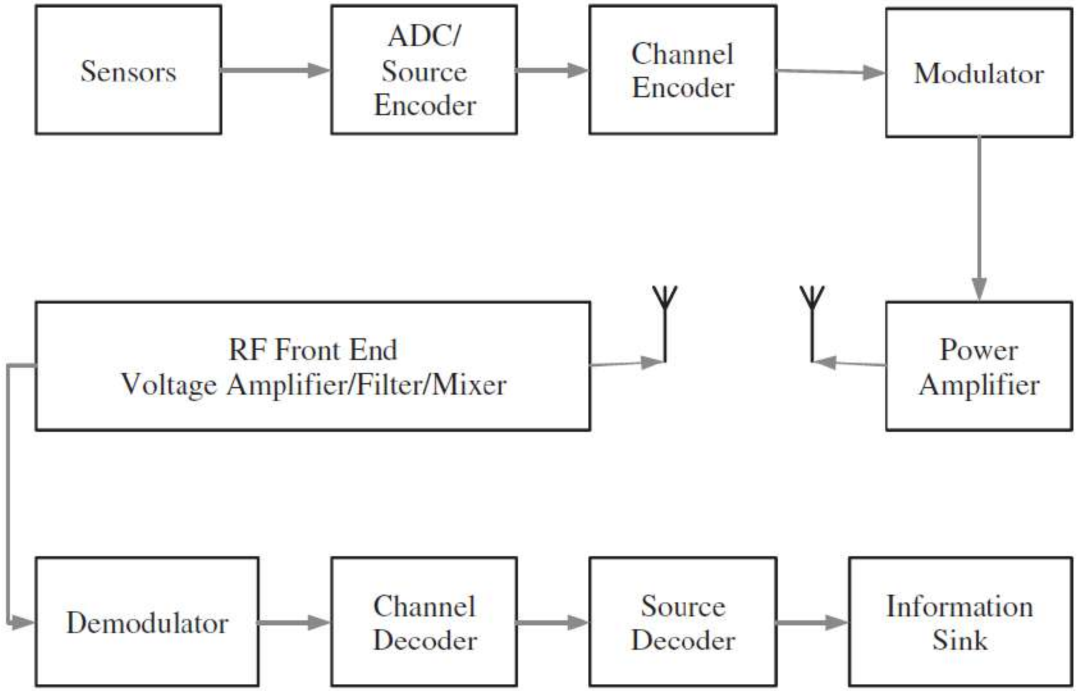
Components of a digital communication system.

Source Encoding: the discrete signal is converted to a binary stream after sampling.

- An efficient source-coding technique can satisfy the channel's bandwidth and signal power requirements.

- It can be achieved by defining a probability model of the information source.
  - The length of each information symbol depends on its probability of occurrence.

PHYSICAL LAYER – BASIC COMPONENTS

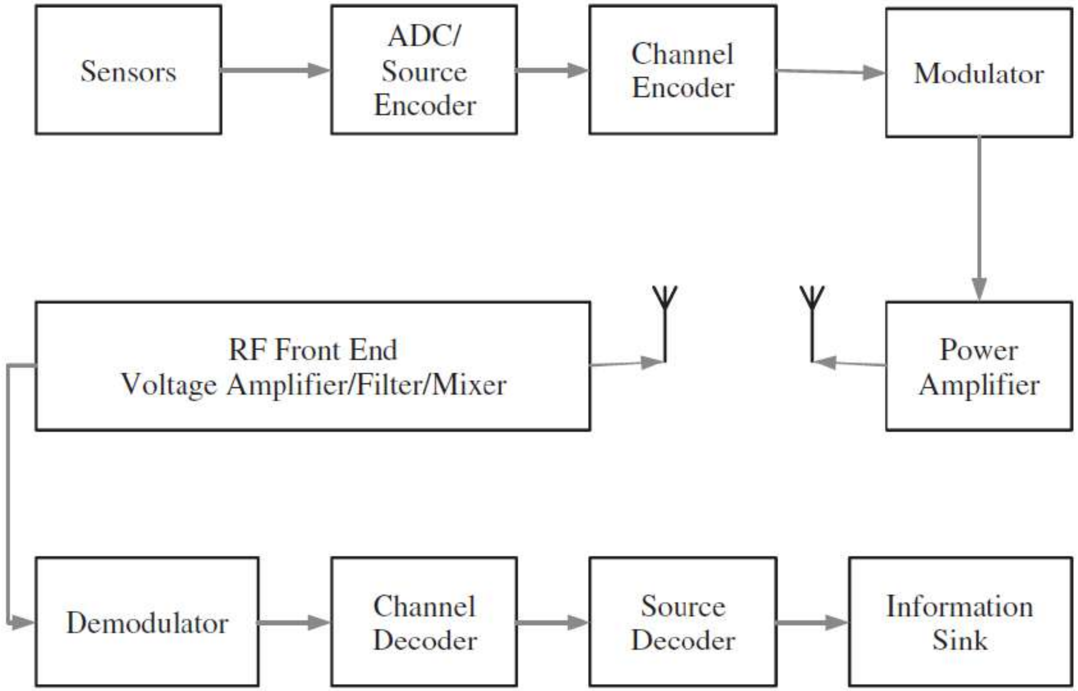


Components of a digital communication system.

Channel Encoding –

- Its aim is to make the transmitted signal robust to noise and interference.
- Moreover, in case of signal corruption, it enables an error to be recognized and the original data to be recovered.
- There are two essential approaches:
  - to transmit symbols from a predetermined codebook, and
  - to transmit redundant symbols.

PHYSICAL LAYER – BASIC COMPONENTS

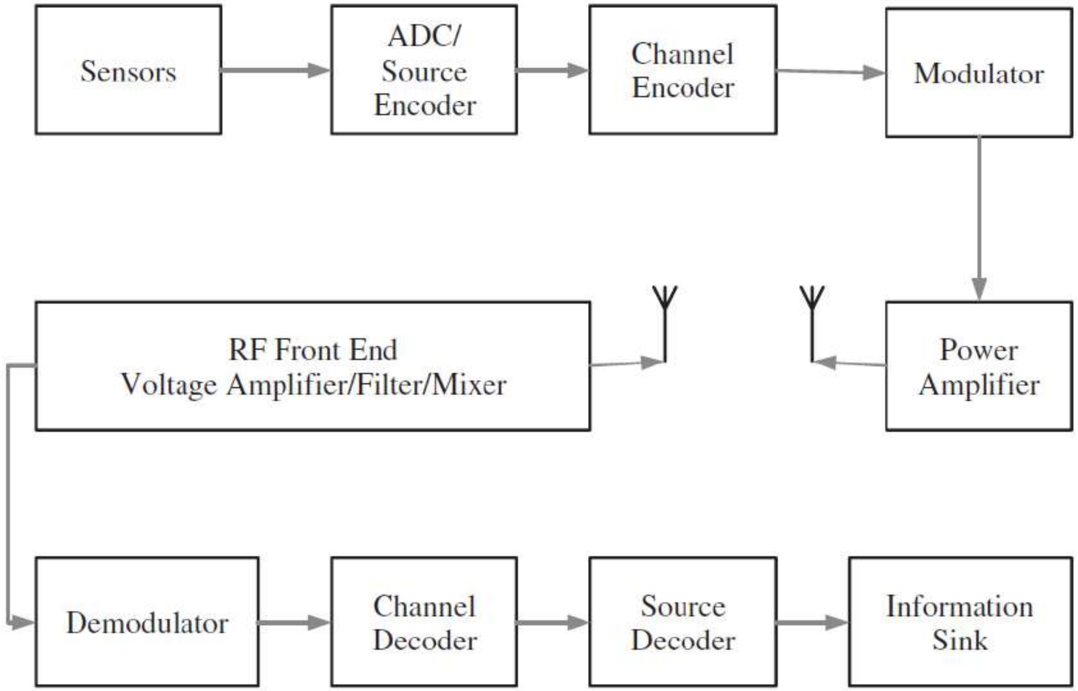


Components of a digital communication system.

Modulation–

- This is a process by which the baseband signal is transformed into a bandpass signal.
- Modulation is useful for various reasons, but the main reason is to transmit and receive signals with short antennas.
- In general, the shorter the wavelength of the transmitted signal, the shorter is the length of the antenna.

PHYSICAL LAYER – BASIC COMPONENTS



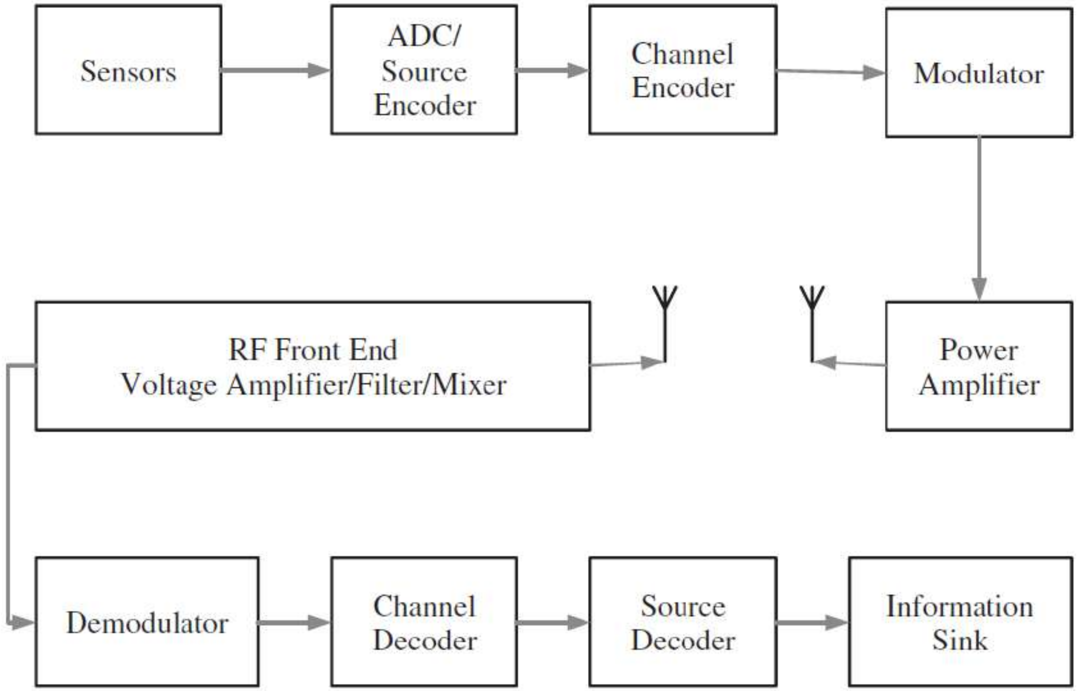
Components of a digital communication system.

Power Amplifier –

- The modulated signal has to be amplified and the electrical energy is converted into electromagnetic energy (electromagnetic radiation) by the transmitter's antenna, and

- The signal is propagated over a wireless link to the desired destination.

PHYSICAL LAYER – BASIC COMPONENTS



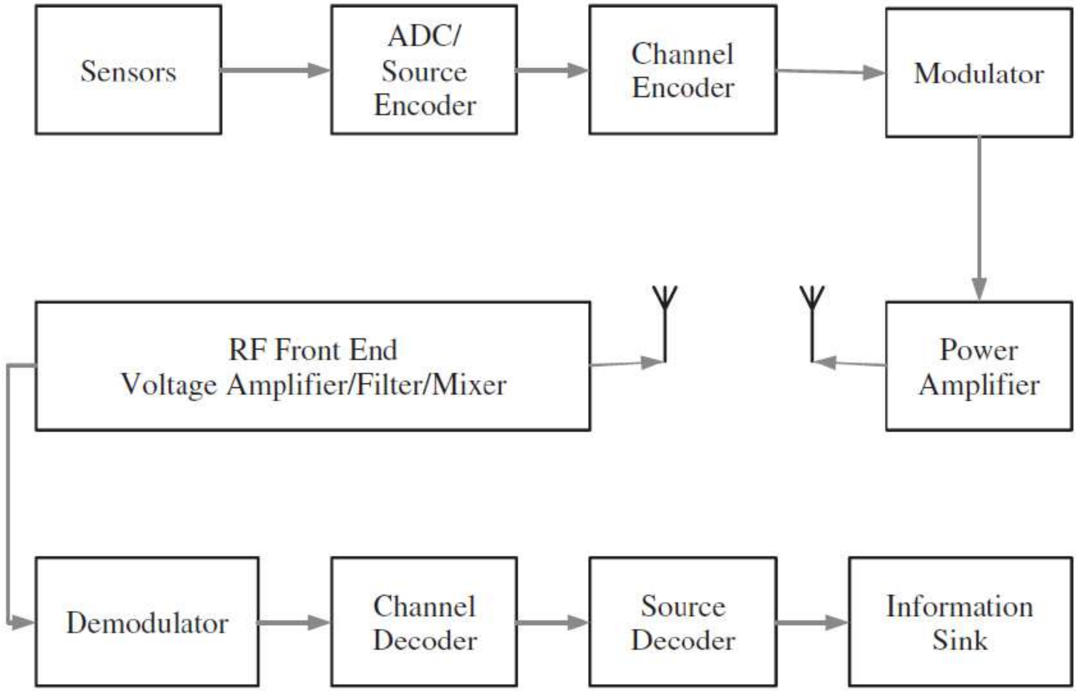
Components of a digital communication system.

Receiver End

- The components of the receiver block carry out the reverse process to retrieve the message signal from the electromagnetic waves. The receiver antenna induces a voltage that is, ideally, similar in shape, frequency, and phase with the modulated signal.

- The magnitude and shape of the signal are changed because of losses and interferences

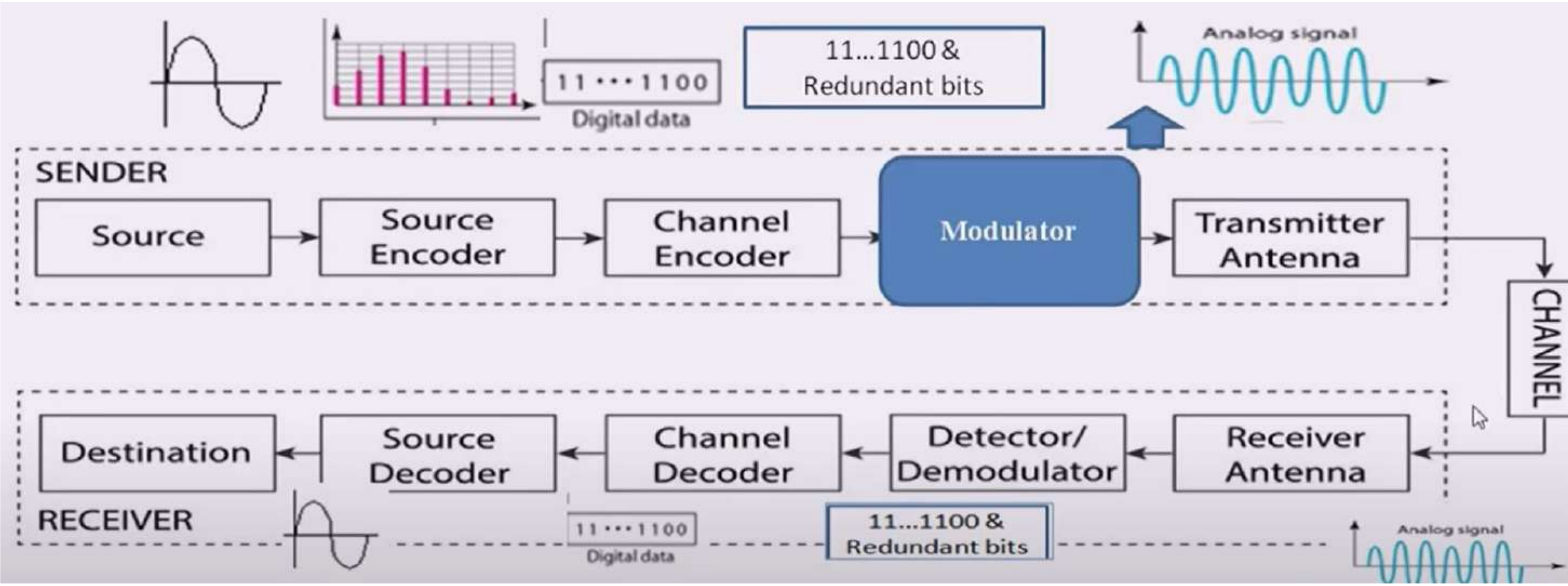
PHYSICAL LAYER – BASIC COMPONENTS



Components of a digital communication system.

- The signal has to pass through a series of *amplification* and *filtering processes*
- It is then transformed back to a baseband signal through the process of *demodulation* and *detection*
- Finally, the baseband signal undergoes *a pulse-shaping process* and *two* stages of *decoding* (channel and source)
  - extract the sequence of symbols - the original analog signal (the message)

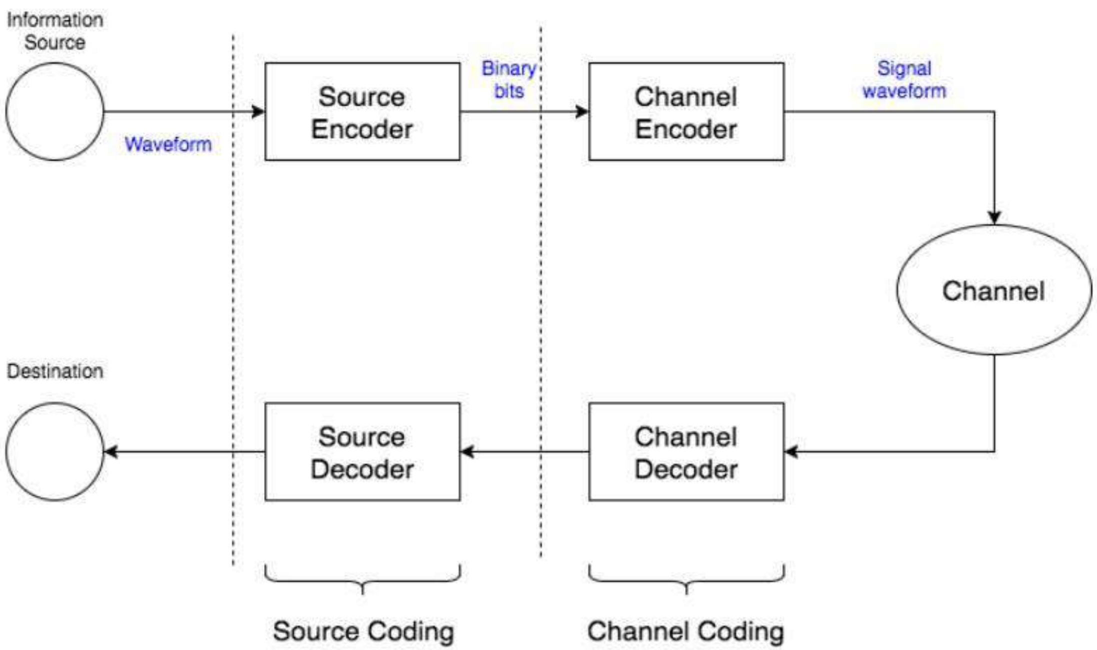
BLOCK DIAGRAM OF DIGITAL COMMUNICATION SYSTEM



BLOCK DIAGRAM OF DIGITAL COMMUNICATION SYSTEM

Digital communication systems are communication systems that use digital sequence (typically binary digits, that is, bits) as an interface between the source coding part and the channel coding part.

Source coding: The source encoder converts information waveforms to bits, while the decoder converts bits back to waveforms. The block diagram of source coding is depicted in Figure.



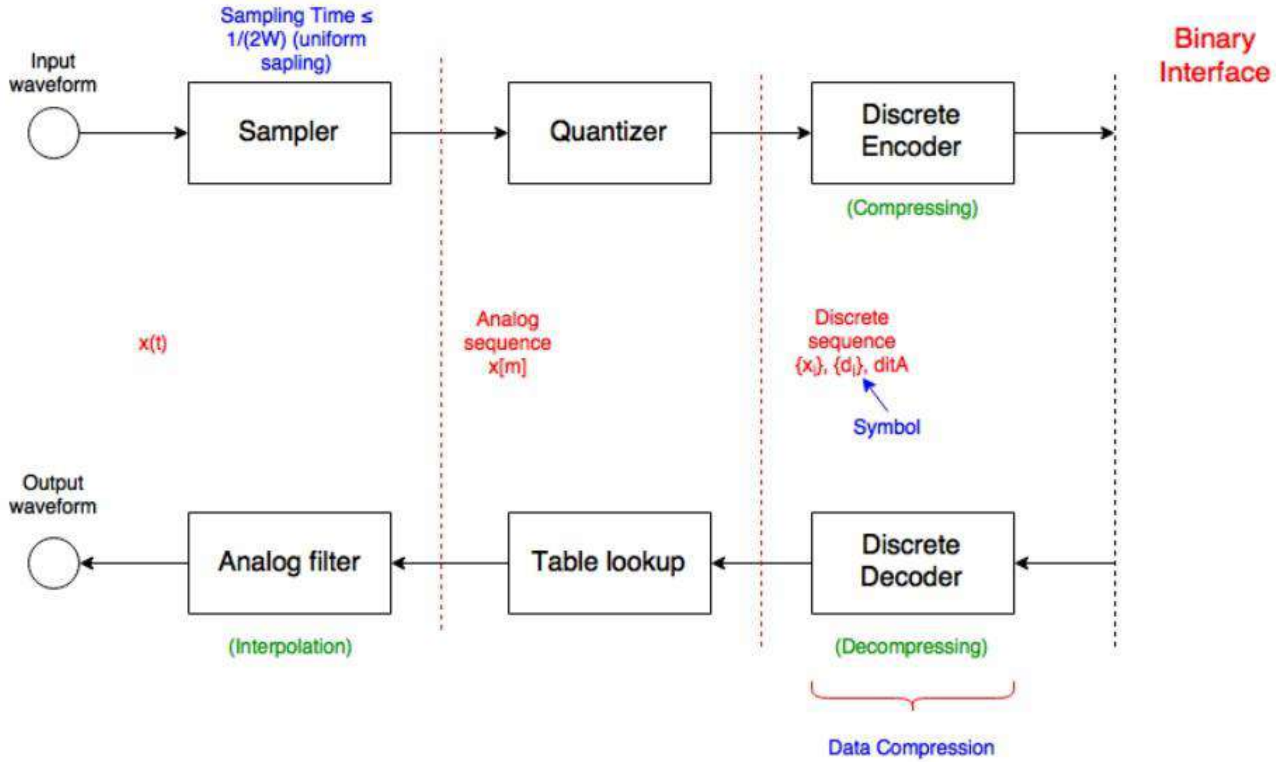
BLOCK DIAGRAM OF DIGITAL COMMUNICATION SYSTEM

Source Coding Source encoding aims to convert information waveforms (text, audio, image, video, etc.) into bits, the universal currency of information in the digital world. The three major steps are:

Sampling: convert the continuous-time analog waveform to discrete-time sequence (but still continuous-valued).

Quantization: convert each continuous-valued symbol to discrete-valued representatives.

Data compression: remove the redundancy in the data and generate roughly i.e. uniformly distributed bits.



PHYSICAL LAYER – SOURCES ENCODING

Source Encoding

- A source encoder transforms an *analog signal* into a *digital sequence*.

- The process consists of: *sampling, quantizing, encoding*

- Suppose a sensor produces an analog signal *s(t)*
- *s(t)* will be sampled and quantized by the analog-to-digital converter (ADC) that has a resolution of *Q* distinct values
- as a result, a sequence of samples, *S =(s[1], s[2], ..., s[n])* are produced
- the difference between the sampled *s[j]* and its corresponding analog value at time *tj* is the *quantization error*
- as the signal varies over time, the quantization error also varies and can be modeled as a random variable with a probability density function, *Ps (t)*

PHYSICAL LAYER – SOURCES ENCODING

Source Encoding

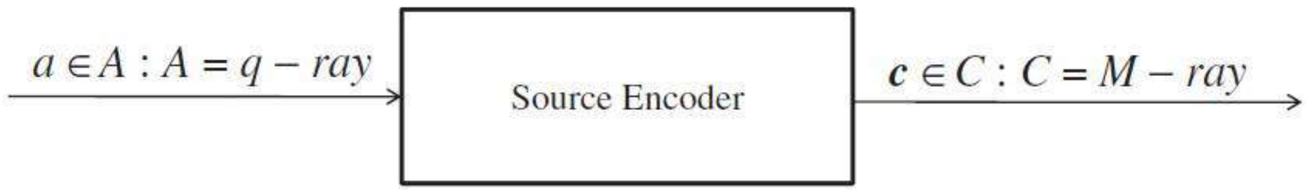
- The *aim* of the source encoder is to map each quantized element, *s[j]* into a corresponding binary symbol of length *r* from a codebook, *C*

- *Block code:* if all the binary symbols (code words) in the codebook are of *equal* length

- Often, the symbol length and the sampling rate are *not uniform*

- It is customary to assign:
    - *short-sized symbols* and *high* sampling rates to the most probable sample values
    - *long-sized symbols* and *low* sampling rates to less probable sample values

PHYSICAL LAYER – SOURCES ENCODING



Input – output relationship of a source encoder.

- A codebook, *C*, can be uniquely decoded, if each sequence of symbols, *(C(1), C(2), …)* can be mapped back to a corresponding value in *S =(*s[1], s[2], ..., s[n]*)*

- A binary codebook has to satisfy the following equation to be uniquely decoded

$$\sum_{i=1}^{u} \left(\frac{1}{r}\right)^{l_i} \leq 1$$

  - where *u* is the size of the codebook
  - *li* is the size of the codeword *C(i)*
  - *r is the Binary Coding System i.e. 2.*

PHYSICAL LAYER – SOURCES ENCODING

- A codebook can be instantaneously decoded
  - if each symbol sequence can be extracted (decoded) from a stream of symbols *without* taking into consideration previously decoded symbols

- This will be possible
  - *if* there does *not exist* a symbol in the codebook, such that the symbol a = $(a_1, a_2, ..., a_m)$ is not a prefix of the symbol b $= (b_1, b_2, ..., b_n)$, where $m < n$ and $a_i = b_i$, $\forall i = 1, 2, ..., m$ within the same codebook

PHYSICAL LAYER – SOURCES ENCODING

Source-encoding techniques

	C^1	C^2	C^3	C^4	C^5	C^6
s_1	0	00	0	0	0	0
s_2	10	01	100	10	01	10
s_3	00	10	110	110	011	110
s_4	01	11	11	1110	111	111
Block code	No	Yes	No	No	No	No
Uniquely decoded	No	Yes	No	Yes	Yes	Yes
$\sum_{i=1}^{n} \left(\frac{1}{2}\right)^{l_i}$	$1\frac{1}{4}$	1	1	$\frac{15}{16} < 1$	1	1
Instantly decoded	No	Yes (block code)	No	Yes (comma code)	No	Yes

INFORMATION THEORY

Why Information Theory?

When the Communication thing is readily measurable, such as an electric current, the study of the communication system is relatively easy. But, when the communication thing is information, the study becomes rather difficult.

- Information Theory answers the following questions:
  - Measure for an amount of information.
  - Measure to improve the communication of information.

- *Unit of information*
  - Communication systems are of *statistical* nature; i.e. the performance of the system can never be described in a deterministic sense.

  - Communication systems are *unpredictable* or *uncertain*.

  - An amount of information is calculated by statistical parameter associated with a *probability scheme*. The parameter should indicate a relative measure of uncertainty relevant to the occurrence of each message in the message ensemble.

INFORMATION THEORY

The amount of information is inversely proportional to the probability of an event. The more the probability of an event, the less is the amount of information associated with it, and vice versa.

$$I(x_j) = f\left[\frac{1}{p(x_j)}\right] \qquad \text{..........Eq. 1}$$

Where x_j is an event with a probability $p(xj)$ and the information associated with it is $I(x_j)$.

Now, let there be another event y_k such that x_j and y_k are independent. Hence the probability of the joint event is $p(x_{j\,,}\, y_k) = p(xj)\ p(y_k)$ with associated information content

$$I(x_{j\,,}\, y_k) = f\left[\frac{1}{p\,(x_{j\,,}\, y_k)}\right] = f\left[\frac{1}{p(xj)\ p(y_k)}\right] \quad \text{..............Eq. 2}$$

The total information $I(x_{j\,,}\, y_k)$ must be equal to the sum of individual information $I(x_j)$ and $I(y_k)$. Where,

$$I(y_k) = f\left[\frac{1}{p(y_k)}\right]$$

Thus, it can be seen that the function on RHS of Eq. 2 must be a function which converts multiplication into addition. Logarithm is one such function. Thus,

INFORMATION THEORY

Thus, it can be seen that the function on RHS of Eq. 2 must be a function which converts multiplication into addition. Logarithm is one such function. Thus,

$$I(x_j, y_k) = \log\left[\frac{1}{p(xj)\ p(y_k)}\right]$$

$$= \log\left[\frac{1}{p(xj)}\right] + \log\left[\frac{1}{p(y_k)}\right]$$

$$= I(x_j) + I(y_k)$$

Hence, the basic equation defining the amount of information (or self information) is

$$I(x_j) = \log\left[\frac{1}{p(xj)}\right] = -\log[p(xj)]$$

Different units of information can be defined for different bases of logarithms.

Base 2 :*Unit is bit*

Base e :*Unit is nat*

Base 10 :*Unit is decit*

In Information theory, unit of information is bit. So, it is assumed to take base 2.

INFORMATION THEORY

An intuitive and meaningful measure of information should have the following properties:

- Self information should decrease with increasing probability.
- Self information of two independent events should be their sum.
- Self information should be a continuous function of the probability.

The only function satisfying the above conditions is the -log of the probability.

Entropy

When we observe the possibilities of the occurrence of an event, how surprising or uncertain it would be, it means that we are trying to have an idea on the average content of the information from the source of the event.

"Entropy is a measure of uncertainty."

Entropy can be defined as a *measure of the average information content per source symbol*. Claude Shannon, the "father of the Information Theory", provided a formula for it as –

$$H = \sum_{i=1}^{M} p_i \log_2 \frac{1}{p_i}$$

INFORMATION THEORY

If someone is sending us a stream symbols from a source (such as those above), we generally have some uncertainty about the sequence of symbols we will be receiving. If not, there would no need for the transmitter to even bother sending us the data.

A useful and common way to model this uncertainty is to assume that the data is coming randomly according to some probability distribution.

The simplest model is to assume that each the symbols $s_1, \ldots, s_M$ have associated probabilities of occurrence $p_1, \ldots, p_M$, and we see a string of symbols drawn independently according to these probabilities. Since the p_i are probabilities, we have $0 \leq p_i \leq 1$ for all i.

Also, we assume the only possible symbols in use are the $s_1, \ldots, s_M$, and so

$$p_1 + \cdots + p_M = 1.$$

INFORMATION THEORY

How much information does a source provide?

Consider just one symbol from the source. The probability that we observe the symbol s_i is p_i , so that if we do indeed observe s_i then we get $\log \frac{1}{p_i}$ bits of information. Therefore, the average number of bits of information we get based on observing one symbol is

$$p_1 \log_2 \frac{1}{p_1} + \cdots + p_M \log_2 \frac{1}{p_M} = \sum_{i=1}^{M} p_i \log_2 \frac{1}{p_i}$$

This is an important quantity called the entropy of the source and is denoted by H.

Definition

Given a source that outputs symbols $s_1, \ldots, s_M$ with probabilities $p_1, \ldots, p_M$, respectively, the entropy of the source, denoted H, is defined as

$$H = \sum_{i=1}^{M} p_i \log_2 \frac{1}{p_i}$$

INFORMATION THEORY

Case I:

If there is only one possible message, i.e. M=1 and p_k =P =1, then

$$H = p1 \log[1/p_1] = 1 \log[1/1] = 0$$

In this case of single possible message, the reception of total message conveys no information.

Case 2:

Only one message out of M having a probability 1 and others have 0.

$$H = \sum_{k=1}^{M} p_k \log_2 \frac{1}{p_k}$$

$$H = p_1 \log \frac{1}{p1} + \lim_{p \to 0} \left[p \log \frac{1}{p} + p \log \frac{1}{p} + \cdots \ldots \ldots .. \right]$$

$$H = 1 \log[1/1] + 0$$

$$H = 0$$

In this case, entropy is zero.

Case 3:

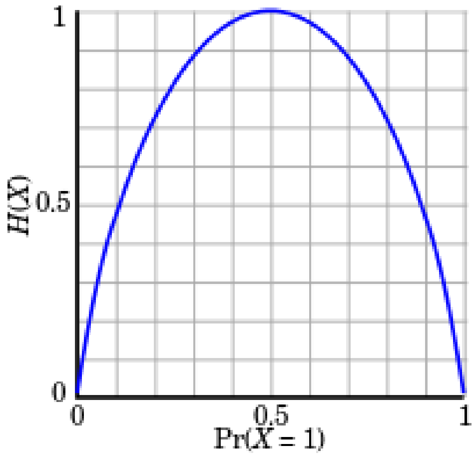
For Binary System (M=2), the entropy is

$$H = p_1 \log\frac{1}{p1} + p_2 \log\frac{1}{p2}$$

Let p1 =p then p2 =1− p1 =1− p =q

$$H = p \log\frac{1}{p} + (1-p) \log\frac{1}{(1-p)}$$

$$H = p \log\frac{1}{p} + q \log\frac{1}{q}$$



The maximum value of H can be found by putting p =1/2

$$H_{max} = 1/2 \log\frac{1}{1/2} + 1/2 \log\frac{1}{1/2} = 1 \text{ bit/message}$$

In this case, we have seen that the entropy is maximum when p =1/2, i.e. both the message are equally probable. In this case, the maximum entropy is then

$$H_{max} = \sum_{k=1}^{M} p_k \log_2\frac{1}{p_k} = \sum_{k=1}^{M} \frac{1}{M} \log M = \log M \text{ bits/message}$$

INFORMATION THEORY

Rate of Information

If a message source generates messages at the rate of r messages per second, the rate of information R is defined as the average number of bits of information per second. Now, H is the average number of bits of information per message. Hence

$$R = r\,H \text{ bits/sec}$$

INFORMATION THEORY

- The importance of the entropy of a source lies in its operational significance concerning coding the source. Since *H* represents *the average number of bits of information per symbol from the source,*

- We might expect that we need to use at least *H* bits per symbol to represent the source with a uniquely decodable code.

- This is in fact the case, and moreover, if we wish to code longer and longer strings of symbols, we can find codes whose performance (average number of bits per symbol) gets closer to H. This result is called the *Source Coding Theorem* and was discovered by *Shannon* in 1948.

- *Source Coding Theorem*
  a) The average number of bits/symbol of any uniquely decodable source must be greater than or equal to the entropy H of the source.

  a) If the string of symbols is sufficiently large, there exists a uniquely decodable code for the source such that the average number of bits/symbol of the code is as close to H as desired.

PHYSICAL LAYER – SOURCES ENCODING

The Efficiency of a Source Encoder

- Quantity that expresses the average length
- Sampled analog signal: $L(C) = E[l_i(C)]$
- Suppose the probability of a m-ary source
  - i.e., it has *q* distinct symbols
  - producing the symbol s_i is P_i and the symbol C_i in a codebook is used to encode s_i
  - the expected length of the codebook is given by:

$$L(C) = \sum_{i=1}^{m} P_i * l_i(C)$$

The Efficiency of a Source Encoder

- To express efficiency in terms of the information entropy or *Shannon's entropy*
  - defined as *the minimum message length* necessary to communicate information

  - related to the *uncertainty* associated with the information

  - if the symbol *si* can be expressed by a binary symbol of *n* bits, the information content of *si* is:

$$I(s_i) = -\log_2 P_i = \log_2 \frac{1}{P_i}$$

  - Logarithmic Measure of information possesses the desired additive property when a number of source outputs is considered as a block.

  - the entropy (in bits) of a *m*-ary memoryless source encoder is expressed as:

$$H = \sum_{i=1}^{M} p_i \log_2 \frac{1}{p_i}$$

The Efficiency of a Source Encoder

- The efficiency of a source encoder in terms of entropy reveals the unnecessary redundancy in the encoding process. This can be expressed by
- Average information per message/information actually transmitted per message

$$\eta = \frac{\textbf{Actual Transinformation}}{\textbf{Maximum Transinformation}}$$

$$\eta = \frac{I(X;Y)}{C} = \frac{I(X;Y)}{Max\ I(X;Y)}$$

$$\eta(C) = \frac{H(S)}{L(C)}$$

Where,
H(S) is the Entropy of the Samples (Actual Transinformation)
L(C) is the Entropy of the Code Block (Maximum Transinformation)

- Transinformation is a measure of the average information per symbol transmitted in the system. i.e. the rate of transmission of information.

- The redundancy of the encoder is: $\frac{L - H(S)}{L} = 1 - \eta$

PHYSICAL LAYER – SOURCES ENCODING

Example 1

Message	Probability	Code	Length of Code
m1	1/2	C1 = 00	l1=2
m2	1/4	C2= 01	l2=2
m3	1/8	C3= 10	l3=2
m4	1/8	C4=11	l4=2

- It is quantized into four distinct values, 0, *1, 2, 3* .
- if the *probability* of occurrence of these values is *P(1)* = 0.5, *P(2)* = 0.25, *P(3)* = 0.*125*, *P(4)* = 0.*125*, then, it is possible to compute the efficiency of the codebooks given in Table
- *li has been taken equal for* each symbol. Hence:

$$L(C) = \sum_{i=1}^{4} l_i * P_i = (2 * 1/2) + (2 * 1/4) + (2 * 1/8) + (2 * 1/8) = 2 \text{ bits/message}$$

Example 1

- Using Equation, the entropy of *Codebook* is calculated as:

$$H(C) = \frac{1}{2}\log\frac{1}{2} + \frac{1}{4}\log\frac{1}{4} + \frac{1}{8}\log\frac{1}{8} + \frac{1}{8}\log\frac{1}{8} = 7/4 = 1.75$$

$$\eta(C) = \frac{H(C)}{L(C)} = \frac{1.75}{2} = 87.5\ \%$$

- Therefore, the encoding efficiency of the codebook, *C2* (see Table) is:

$$\eta(C) = 87.5\ \%$$

- The redundancy in *C2* is:

$$R(C) = 1 - \eta = 1 - 0.875 = 0.125 = 12.5\%$$

- In terms of energy efficiency, this implies that 12.5*% of the transmitted bits are unnecessarily redundant*, because *C* is not compact enough.

PHYSICAL LAYER – SOURCES ENCODING

Example 1

Message	Probability	Code	Length of Code
m1	1/2	C1 = 0	l1=1
m2	1/4	C2= 10	l2=2
m3	1/8	C3= 110	l3=3
m4	1/8	C4=111	l4=3

- It is quantized into four distinct values, 0, *1*, *2*, *3* .

- if the *probability* of occurrence of these values is *P(1)* = 0.5, *P(2)* = 0.25, *P(3)* = 0.*125*, *P(4)* = 0.*125*, then, it is possible to compute the efficiency of the codebooks given in Table

- *li has been taken different for* each symbol. Hence:

$L(C) = \sum_{i=1}^{4} l_i * P_i = (1 * 1/2) + (2 * 1/4) + (3 * 1/8) + (3 * 1/8) = 1.75$ bits/message

Example 2

- Using Equation, the entropy of *Codebook* is calculated as:

$$H\,(C) = \frac{1}{2}\log\frac{1}{2} + \frac{1}{4}\log\frac{1}{4} + \frac{1}{8}\log\frac{1}{8} + \frac{1}{8}\log\frac{1}{8} = 7/4 = 1.75$$

$$\eta\,(C) = \frac{H(C)}{L(C)} = \frac{1.75}{1.75} = 100\,\%$$

- Therefore, the encoding efficiency of the codebook, *C2* (see Table) is:

$$\eta\,(C) = 87.5\,\%$$

- The redundancy in *C2* is:

$$R\,(C) = 1 - \eta = 1 - 1 = 0 = 0\%$$

- In terms of energy efficiency, this implies that no bits are *transmitted as redundant bits*, because *C* is perfect enough and efficient codebook.

PHYSICAL LAYER – SOURCES ENCODING

Source Coding Techniques

1. Huffman Code
2. Lempel-Ziv Code
3. Shannon-Fano Code
4. Arithmetic Code

Pulse Code Modulation & Delta Modulation

PCM and DM are the two predominantly employed *source encoding* techniques

- *In digital pulse code modulation*
  - the signal is quantized first
  - each sample is represented by a binary word from a finite set of words

- The resolution of a PCM technique and the source encoder bit rate are determined by
  - the size of the individual words
  - the number of words in the set

PHYSICAL LAYER – SOURCES ENCODING

Pulse Code Modulation

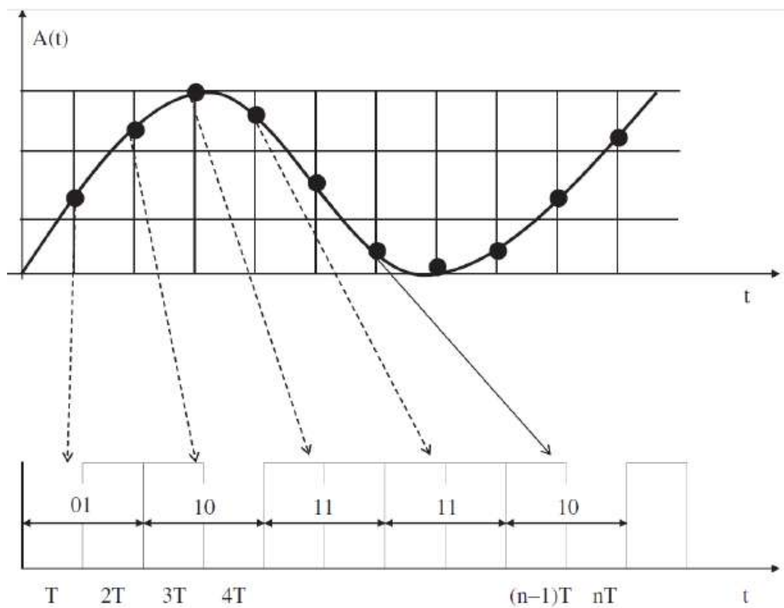
In PCM, information is conveyed in the presence or absence of *pulses*

- greatly enhances the transmission and regeneration of *binary words*

- the associated cost with this form of source encoding is
  - the quantization error, the energy and bandwidth required to transmit the multiple bits for each sampled output!

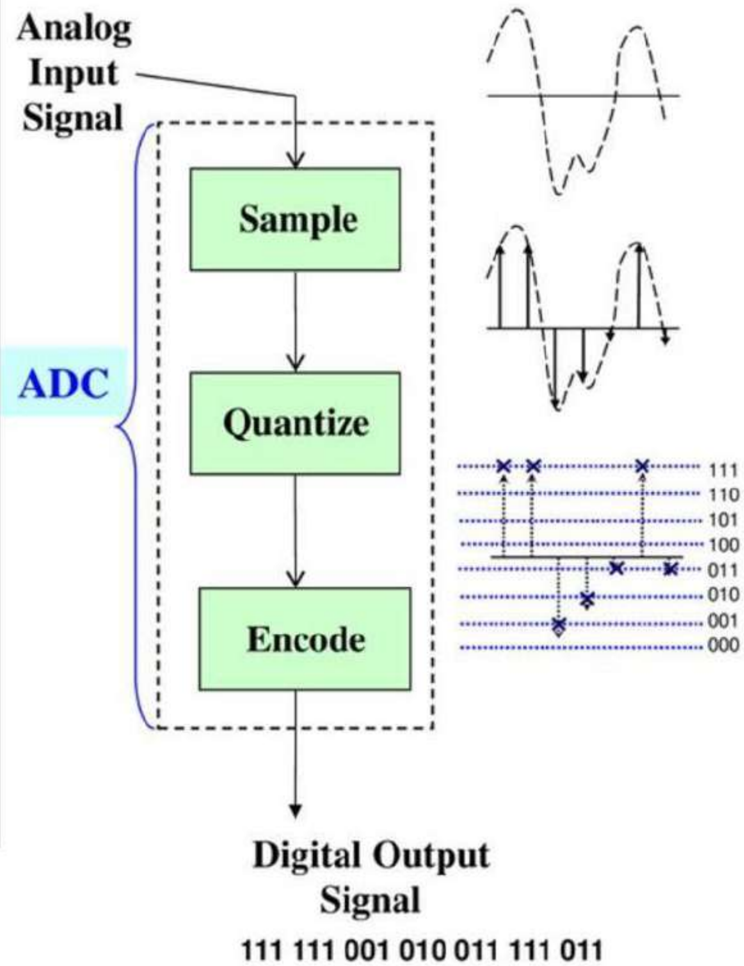
- Figure illustrates a PCM technique that uses two bits to encode a single sample

- Four distinct levels are permissible during sampling



A PCM based source encoding.

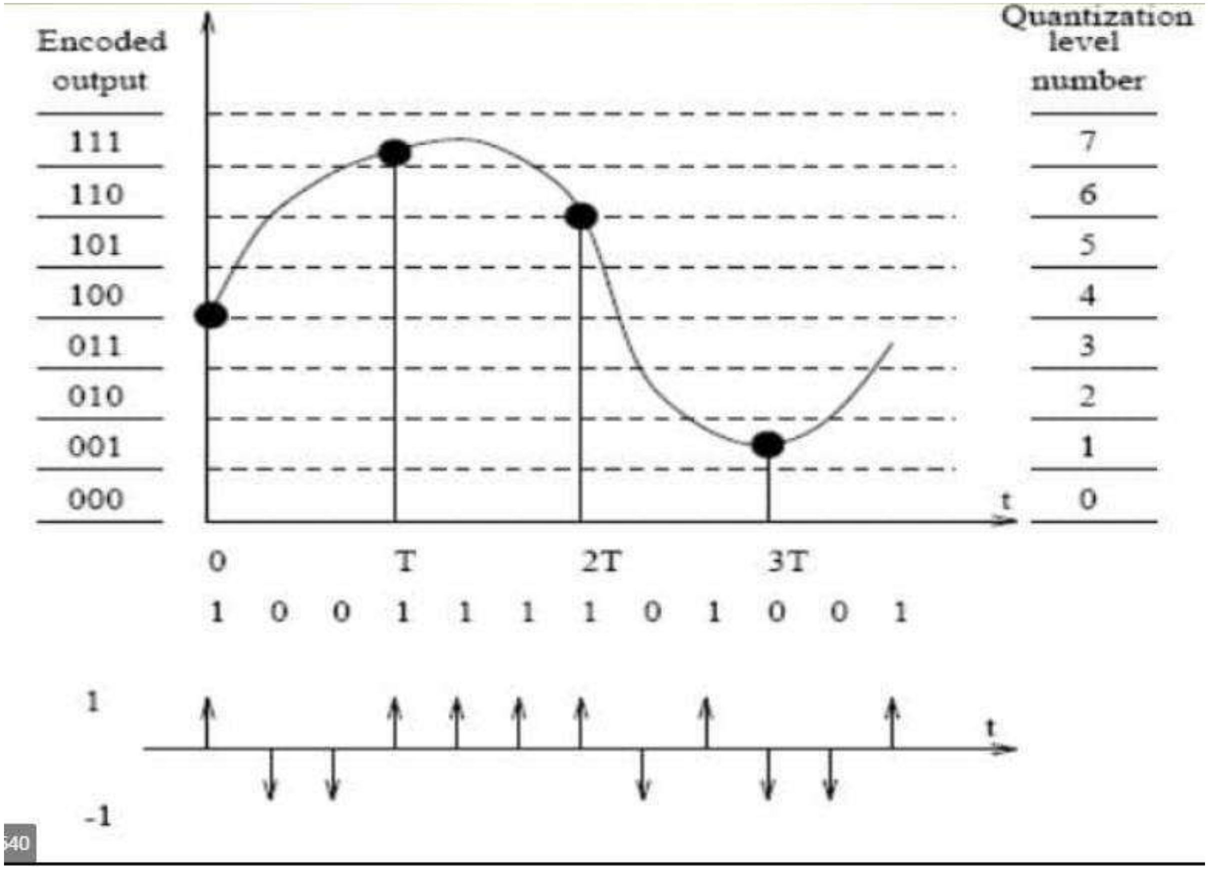
Pulse Code Modulation



- The ***Analog-to-digital Converter*** **(ADC)** performs three functions:
  - **Sampling**
    - Makes the signal discrete in time.
    - If the analog input has a bandwidth of W Hz, then the ***minimum sample frequency*** such that the signal can be reconstructed without distortion is 2W or more.
  - **Quantization**
    - Makes the signal discrete in amplitude.
    - Round off to one of q discrete levels.
  - **Encode**
    - Maps the quantized values to digital words that are ν bits long.
- If the (Nyquist) ***Sampling Theorem*** is satisfied, then only quantization introduces distortion to the system.

PHYSICAL LAYER – SOURCES ENCODING

Pulse Code Modulation



PHYSICAL LAYER – SOURCES ENCODING

Pulse Code Modulation

Level	Code word
0	000
1	001
2	010
3	011
4	100
5	101
6	110
7	111

Levels are encoded using this table

Table: Quantization levels with belonging code words

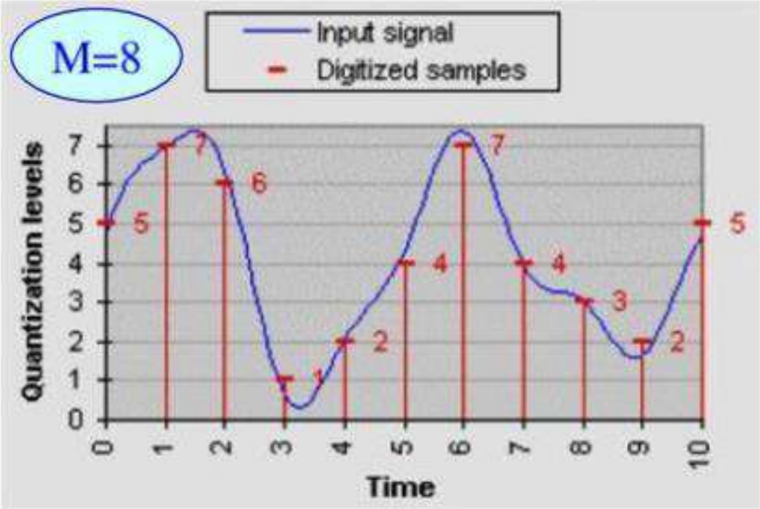


Chart 1. Quantization and digitalization of a signal.

Signal is quantized in 11 time points & 8 quantization segments.

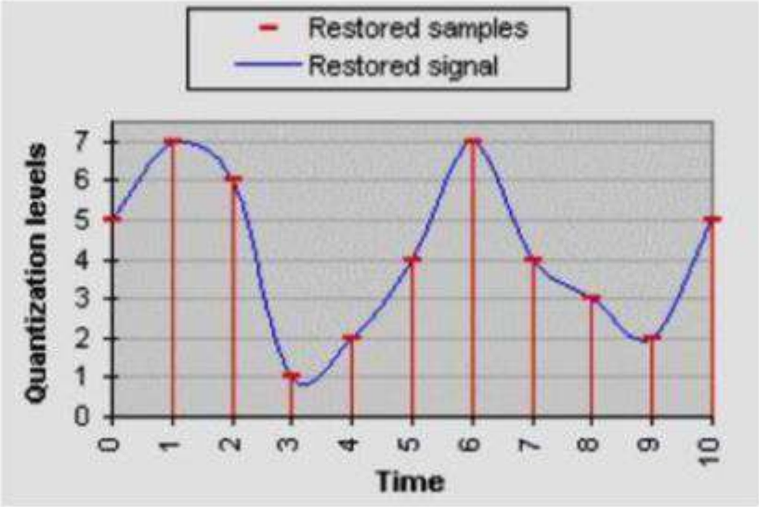


Chart 2. Process of restoring a signal.

PCM encoded signal in binary form:

101 111 110 001 010 100 111 100 011 010 101

Total of **33** bits were used to encode a signal

Delta Modulation

- Linear PCM (LPCM) is a linear quantization PCM.

- Differential PCM (DPCM) encodes PCM values as the difference between the current value and the expected value. The algorithm predicts the next sample based on the previous samples, and the encoder only stores the differentiation between this projection and the actual value. If the projection is reasonable, less bits can be used to represent the same information. For audio, this type of encoding decreases the number of bits required per sample by about 25% compared to PCM.

- Adaptive DPCM (ADPCM) is a variation of DPCM that changes the size of the quantization steps to allow further reduction in the bandwidth required for a given signal-to-noise ratio.

- Delta modulation is a form of DPCM which uses one bit per sample to indicate whether the signal is increasing or decreasing from the previous sample.

PHYSICAL LAYER – SOURCES ENCODING

Delta Modulation

Delta modulation is a *digital pulse modulation* technique

- it has found widespread acceptance *in low bit rate* digital systems
- it is a *differential* encoder and transmits bits of information
- the information describes *the difference between successive* signal values, as opposed to the actual values of a time-series sequence
- the difference signal, *V*d*(t)*, is produced by first estimating the signal's magnitude based on previous samples (*Vi (t*0*)*) and comparing this value with the actual input signal, *V*in*(t*0*)*
- The polarity of the difference value indicates the polarity of the pulse transmitted

PHYSICAL LAYER – SOURCES ENCODING

Delta Modulation

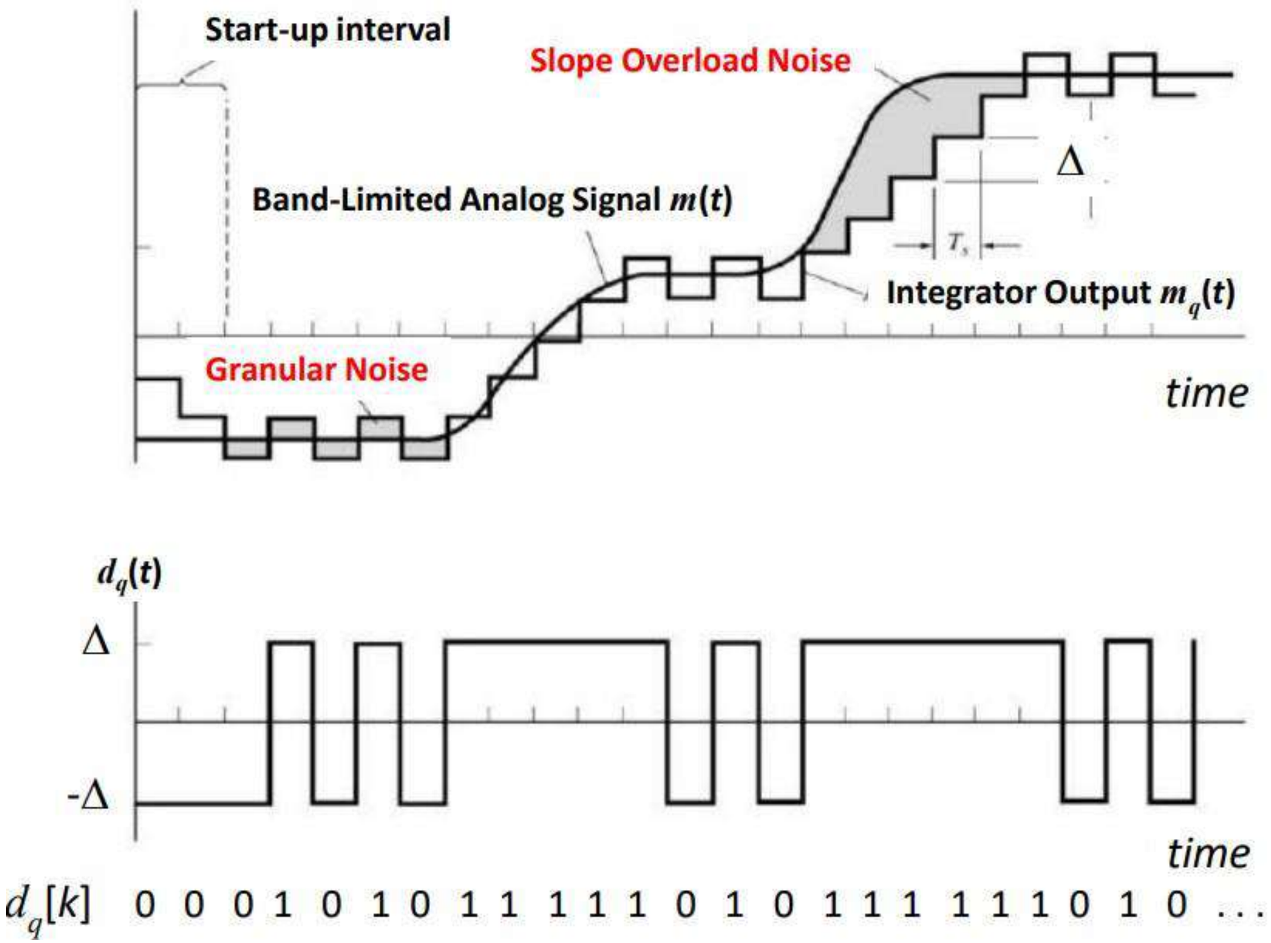
Delta modulation is a *digital pulse modulation* technique

- Delta modulation (DM) is the simplest method for analog-to-digital conversion (ADC).
- Delta modulation uses 1-bit per sampling period (TS) – it is a 1-bit ADC.
-
- Delta modulation requires a sampling rate much greater than the Nyquist rate (commonly four or five times the Nyquist rate).
- DM is closely related to DPCM.
- In DM we use a first-order predictor (one time delay TS is the predictor).
- DM uses very simple hardware and is low cost for that reason.
- The transmitted output is a binary stream of $\pm\Delta$ pulses at f_S. It gives a stepwise approximation m_q (t) to m(t).

PHYSICAL LAYER – SOURCES ENCODING

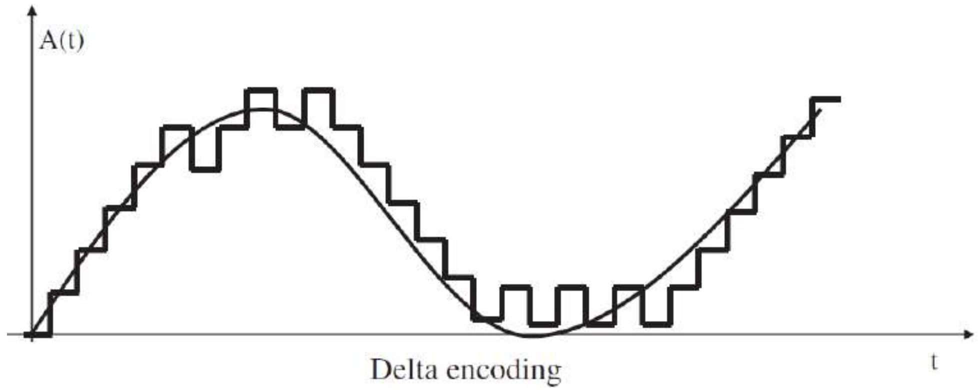
Delta Modulation

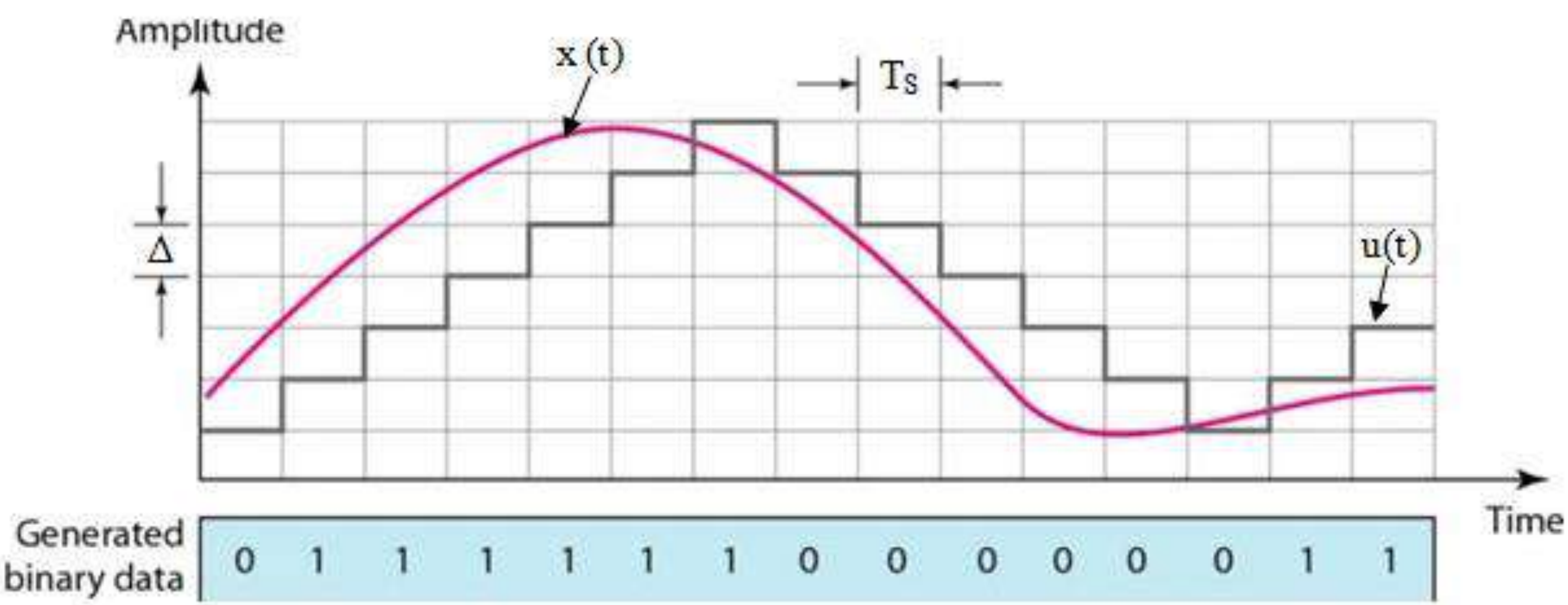
- The difference signal is a *measure* of the *slope of the signal*
  - first, *sampling* the analog signal
  - then, *varying* the amplitude, width, or the position of the digital signal in accordance with the amplitude of the sampled signal



PHYSICAL LAYER – SOURCES ENCODING

Delta Modulation





PHYSICAL LAYER – CHANNEL ENCODING

Channel Encoding

- To produce a sequence of data that is robust to noise.

- To provide error detection and forward error correction mechanisms.

- In simple and cheap transceivers, forward error correction is costly and, therefore, the task of channel encoding is limited to the detection of errors in packet transmission.

- The noise present in a channel creates unwanted errors between the input and the output sequences of a digital communication system. The error probability should be very low, nearly $\leq 10^{-6}$ for a reliable communication.

- The channel coding in a communication system, introduces redundancy with a control, so as to improve the reliability of the system.

- The source coding reduces redundancy to improve the efficiency of the system.

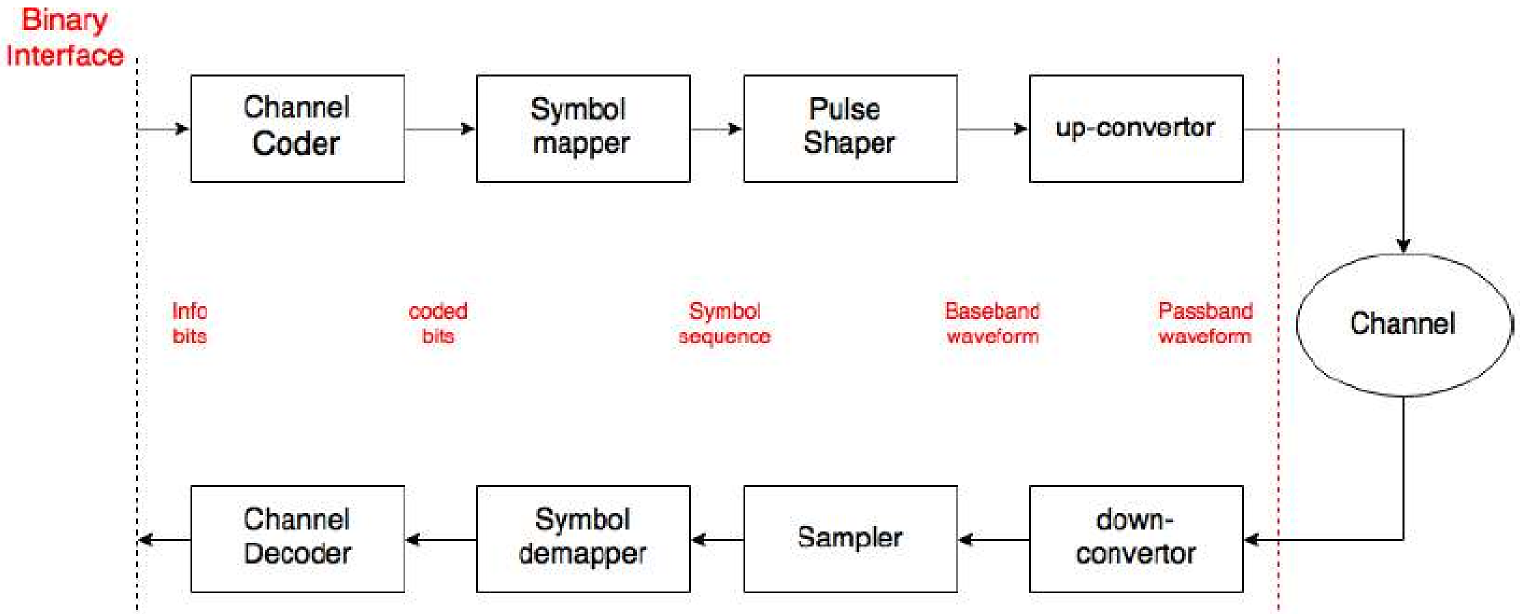
PHYSICAL LAYER – CHANNEL ENCODING

Channel Encoding

- Channel coding consists of two parts of action.
  1. Mapping incoming data sequence into a channel input sequence. The mapping is done by the transmitter, with the help of an encoder.

  2. Inverse Mapping the channel output sequence into an output data sequence. The inverse mapping is done by the decoder in the receiver.

- The final target is that the overall effect of the channel noise should be minimized.



PHYSICAL LAYER – CHANNEL ENCODING

Channel Encoding

- The four major steps are:
  1. Error correcting codes: introduce redundancy into the information bits and produce longer coded bits.
  2. Symbol mapping: map the coded bits to *constellation points*, each of which is a complex symbol.
  3. Pulse shaping: modulate the symbol to suitable *baseband* waveforms.
  4. Up conversion: convert the baseband waveform to *passband* waveform, so that the effective frequency band follows the constraints from the physical world.
- Channel decoding does the reverse of encoding.
- These Error Correcting Techniques are also called Error-Control Coding:
  1. Block Code (Parity Check Code, Binary Code Space, Linear Block Code)
  2. Hamming Code
  3. Cyclic Codes
  4. Convolutional Codes

PHYSICAL LAYER – CHANNEL ENCODING

Shannon Channel Coding Theorem

- It is related with the rate of transmission over a communication channel.

- The communication channel may be noisy or limited bandwidth.

- Shannon's Coding Theorem says that

  "It is possible to device a mean whereby a communication system will transmit information with an arbitrarily small possibility of error provided that the information rate R is less than equal to C, Channel Capacity."

- The statements of Shannon's Theorem is:
  - Given a source of M equally likely messages, with M>>1, which is generating information at a rate R. Given a Channel Capacity C. Then, if R<<C, there exists a coding technique such that the output of the source may be transmitted over the channel with a probability of error of receiving the message which may be made arbitrarily small.

  - It indicates that for R ≪ C. error free transmission is possible in the presence of noise.

PHYSICAL LAYER – CHANNEL ENCODING

Shannon Hartley Theorem

- According to the Shannon–Hartley theorem, the capacity of a channel to transmit a message without an error is given as:

$$C = B \cdot \log_2 \left(1 + \frac{S}{N} \right)$$

where *C* is the channel capacity in bits per second;
B is the bandwidth of the channel in hertz;
S is the average signal power over the entire bandwidth, measured in watts; and
N is the average noise power over the entire bandwidth, measured in watts.

- This equation states that
  - For data transmission with free of errors, its transmission rate should be below the channel's capacity.
  - It also indicates how the signal-to-noise (SNR) ratio, can improve the channel's capacity.

PHYSICAL LAYER – CHANNEL ENCODING

Shannon Hartley Theorem

- The equation reveals two independent reasons why errors can be introduced during transmission:
    1. Information will be lost if the message is transmitted at a rate higher than the channel's capacity. This type of error is called *equivocation* in information theory. It is characterized as a subtractive error.

    2. Information will be lost because of noise, which adds irrelevant information into the signal.

- It is possible to describe the impact of irrelevance and equivocation as well as the percentage of information that can be transmitted over the channel without an error, which is also called Transinformation or Mutual Information.

- Question is How?
  Answer: Entropy, Joint Entropy & Conditional Entropy

PHYSICAL LAYER – CHANNEL ENCODING

Joint Entropy & Conditional Entropy

- In a single probability scheme, we may study the behavior of either transmitter or receiver by calculating the associated Entropy.

- To study of Communication System, we must simultaneously study the behavior of transmitter and receiver. This will rise the concept of two dimensional probability scheme.
- Let there be two finite discrete sample space S_1 and S_2 and we can assume their product as $S = S_1 S_2$, Let

$$[X] = [x_1\ x_2\ x_3\ \ldots\ldots x_m]$$
$$[Y] = [y_1\ y_2\ y_3\ \ldots\ldots y_n]$$

- Be the sets of events in S_1 and S_2 respectively. Each event x_j of S_1 may occur in conjunction with any event y_k in S_2. Hence, the complete set of events in $S = S_1 S_2$ is

$$\lfloor XY \rfloor = \begin{bmatrix} x_1 y_1 & \cdots & x_1 y_n \\ \vdots & \ddots & \vdots \\ x_m y_1 & \cdots & x_m y_n \end{bmatrix}$$

Thus, we have three sets of complete probability schemes

$$P(X) = [P(x_j)]$$
$$P(Y) = [P(y_k)]$$
$$P(XY) = [P\ (x_j, y_k)]$$

PHYSICAL LAYER – CHANNEL ENCODING

Joint Entropy & Conditional Entropy

- We have three complete probability schemes and naturally there will be three associative entropies.

$$H(X) = -\sum_{j=1}^{M} p(xj)\log_2 p(xj)$$

$$P(x_j) = \sum_{k=1}^{n} p(xj, yk)$$

$$H(Y) = -\sum_{k=1}^{n} p(y_k)\log_2 p(y_k)$$

$$P(y_k) = \sum_{j=1}^{m} p(xj, yk)$$

$$H(XY) = -\sum_{j=1}^{m}\sum_{k=1}^{n} p(xj, yk)\log_2 p(xj, yk)$$

H(X) and H(Y) are marginal entropies of X & Y respectively, and H(XY) is the joint entropy of X & Y.

The conditional probability p(X/Y) is given by

$$P(X/Y) = \frac{P(X,Y)}{P(Y)}$$

We know that the y_K may occur in conjunction with $x_1, x_2, \ldots, x_m$.

PHYSICAL LAYER – CHANNEL ENCODING

Joint Entropy & Conditional Entropy

- The conditional entropy may be defined as

$$H(X/Y) = -\sum_{j=1}^{m}\sum_{k=1}^{n} p(xj, yk)\, log_2\, p(xj/yk)$$

- Similarly, it can be shown that

$$H(Y/X) = -\sum_{j=1}^{m}\sum_{k=1}^{n} p(xj, yk)\, log_2\, p(yk/xj)$$

H(X/Y) and H(Y/X) are average conditional entropies or simply conditional entropies.

PHYSICAL LAYER – CHANNEL ENCODING

Joint Entropy & Conditional Entropy

- There are five Entropies associated with a two-dimensional probability scheme. They are: H(X), H(Y), H(X,Y), H(X/Y), and H(Y/X). Where X represents a transmitter and Y a receiver. Therefore, the following interpretations of the different entropies for a communication system can be derived:

- H(X): Average information per character at the transmitter, or entropy of the transmitter. *It is the probabilistic nature of the transmitter.*

$$H(X) = -\sum_{j=1}^{M} p(xj)\log_2 p(xj)$$

- H(Y): Average information per character at the receiver, or entropy of the receiver. *It is the probabilistic nature of the receiver.*

$$H(Y) = -\sum_{k=1}^{n} p(yk)\log_2 p(yk)$$

- H(X, Y): Average information per pair of the transmitted and received characters or, *the average uncertainty of the communication system as a whole.*

$$H(XY) = -\sum_{j=1}^{m}\sum_{k=1}^{n} p(xj, yk)\log_2 p(xj, yk)$$

PHYSICAL LAYER – CHANNEL ENCODING

Joint Entropy & Conditional Entropy

- H(X/Y): A received character y_k may be the result of the transmission of one of the x_j's with a given probability. The entropy associated with this probability scheme, when y_k covers all received symbols. The conditional entropy is a measure of information about the transmitter, where it is known that Y is received.
- The content of information that can be lost because of the channel's inherent constraints can be quantified by observing the input *x* given that the output *y* is known:

$$H(X/Y) = -\sum_{j=1}^{m}\sum_{k=1}^{n} p(xj, yk)\, log_2\, p(xj/yk)$$

A good channel encoding scheme is one that has a high inference probability. This can be achieved by introducing redundancy during channel encoding.

H(X/Y) indicates how well one can recover the transmitted symbols, from the received symbols; i.e. It gives the information lost in the channel. This is also known as equivocation.

Joint Entropy & Conditional Entropy

- H(Y/X): A transmitted character x_j may be the result of the reception of one of the y_k's with a given probability. The entropy associated with this probability scheme, when x_j covers all transmitted symbols. The conditional entropy is a measure of information about the receiver, where it is known that X is transmitted.

  The content of information that can be introduced into the channel due to noise is described as the conditional information content, I*(Y/X)*. It is the information content of *y* that can be observed provided that *x* is known. The conditional entropy is given as:

$$H(Y/X) = -\sum_{j=1}^{m}\sum_{k=1}^{n} p(xj, yk)\, log_2\, p(yk/xj)$$

  A good channel encoder is one that reduces the irrelevance entropy.

  H(Y/X) indicates how well one can recover the received symbols, from the transmitted symbols; i.e. It can be considered as the noise entropy added in the channel. Thus, it is a measure of an error or noise due to channel.

Joint Entropy & Conditional Entropy

- The relationship between the different entropies is found as follows:

$$H(XY) = H(X/Y) + H(Y)$$

Similarly, it can be shown that

$$H(XY) = H(Y/X) + H(X)$$

Mutual Information

- The information content $I\ (X;Y)$ that overcomes the channel's constraints to reach the destination (the receiver) is called transinformation. The mutual information I(X; Y) indicates a measure of average information per symbol transmitted in the system. Given the input entropy, $H(X)$, and equivocation, $H(X|Y)$, the transinformation is computed as:

$$I(X;Y) = H(X) - H(X/Y) \quad \text{-------}\ 1$$

$$I(X;Y) = H(Y) - H(Y/X) \quad \text{-------}\ 2$$

$$I(X;Y) = H(X) + H(Y) - H(X,Y) \quad \text{------}\ 3$$

Since H(X) represents the uncertainty of the channel input (or source) before the channel output is observed, and

H(X/Y) is the uncertainty of the channel input (or Source) after the channel output is observed.

The mutual information I(X; Y) represents the uncertainty of the channel input (or Source) that is resolved by observing the channel output. *Since, I(X; Y) indicates a measure of the information transferred through the channel. It is also known as transferred information and Transinformation.*

Mutual Information

$$I(X;Y) = H(X) - H(X/Y) \quad \text{-------}\ 1$$

The transinformation is equal to the average source information minus the uncertainty that still remains about the message.

In other words, H(X/Y) is the additional information needed at the receiver after the reception in order to completely specify the message sent.

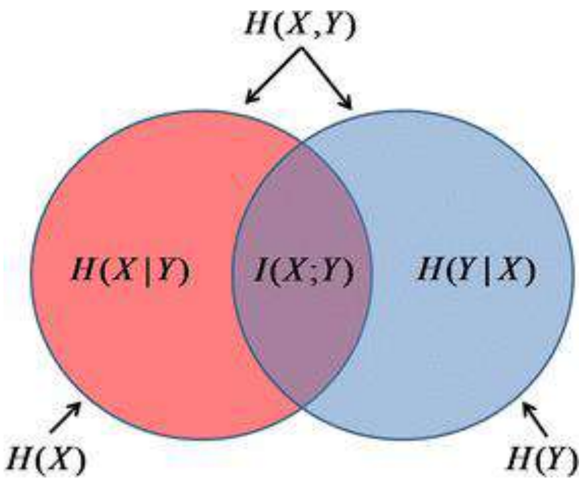
Thus, H(X/Y) gives the information lost in the channel. This is also known as equivocation.

$$I(X;Y) = H(Y) - H(Y/X) \quad \text{-------}\ 2$$

The transinformation is equal to the receiver entropy minus that part of the receiver entropy which is not the information about the source.

Thus, H(Y/X) is considered as the noise entropy added in the channel. Thus, it is a measure of an error or noise due to channel.

Equivocation, Irrelevance & Transinformation



Channel Capacity

The mutual information I(X; Y) indicates a measure of average information per symbol transmitted in the system. A suitable measure for the efficiency of transmission of information may be introduced by comparing the actual rate and the upper bound of the rate of the information transmission of the channel.

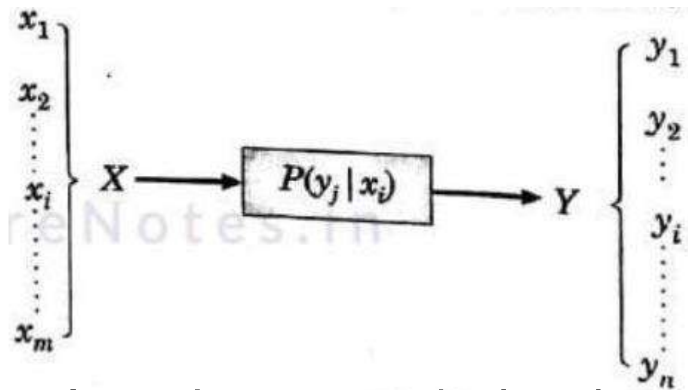
Shannon's has introduced the concept of the Channel Capacity:

Channel Capacity is defined as the maximum of mutual information. Thus the channel capacity C is given by

$$C = \max I(X;Y) = \max [H(X) - H(X/Y)]$$

Channel Representation

A communication channel may be defined as the path or medium through which the symbols flow to the receiver end. A discrete memory channel (DMC) is a statistical model with an input X and an output Y as shown in figure:



During each unit of time (signaling interval, the channel accepts an input symbol from X, and in response it generates an output symbol from Y. The channel is said to be "discrete" when the alphabets of X and Y are both finite.

Also, it is said to be "memoryless" when the current output depends on only the current input and not on any of the previous inputs.

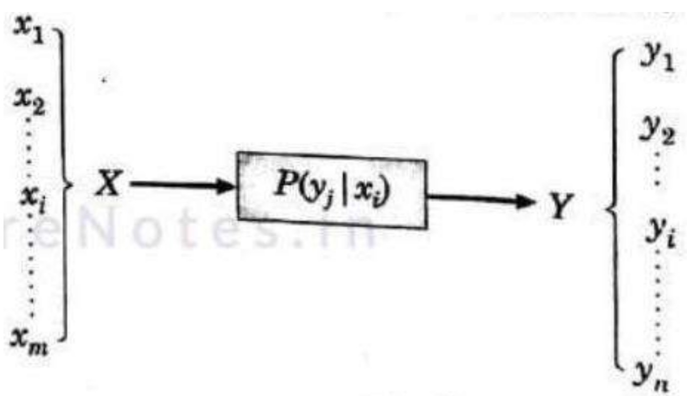
PHYSICAL LAYER – SOURCES ENCODING

Channel Representation

A diagram of DMC with m inputs and n outputs has been given in the figure. The input X consists of input symbols $x_1, x_2, \ldots, x_m$.

The a priori probabilities of these source symbols $P(x_i)$ are assumed to be known. The output Y consists of output symbols $y_1, y_2, \ldots, y_n$.

Each possible input-output path is indicated along with probability $P(y_j/x_i)$, where $P(y_j/x_i)$ is called probability of obtaining output y_j given that input x_i and is called a channel transition probability.



PHYSICAL LAYER – SOURCES ENCODING

Channel Matrix

A Channel is completely specified by the complete set of transition probabilities. Accordingly, the channel is often specified by the matrix of transition probabilities [P(Y/X)]. This matrix is given by

$$[P(Y|X)] = \begin{bmatrix} P(y_1|x_1) & P(y_2|x_1) & \cdots & P(y_n|x_1) \\ P(y_1|x_2) & P(y_2|x_2) & \cdots & P(y_n|x_2) \\ \cdots & \cdots & \cdots & \cdots \\ P(y_1|x_m) & P(y_2|x_m) & \cdots & P(y_n|x_m) \end{bmatrix}$$

This matrix is called a Channel Matrix. Since each input to the channel results some output, each row of the channel matrix must be sum to unity. This means that

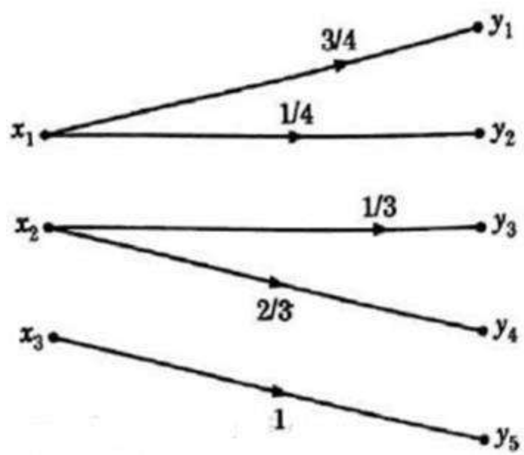
$$\sum_{j=1}^{n} P(y_j|x_i) = 1 \text{ for all } i$$

Lossless Channel

A Channel is described by a channel matrix with only one non-zero element in each column is called a lossless channel. An example of lossless channel has been shown in figure, and the corresponding channel matrix is given in the matrix below:

$$P[Y/X] = \begin{pmatrix} 3/4 & 1/4 & 0 & 0 & 0 \\ 0 & 0 & 1/3 & 2/3 & 0 \\ 0 & 0 & 0 & 0 & 1 \end{pmatrix}$$

It can be shown in the lossless channel, no source information is lost in the transmission.



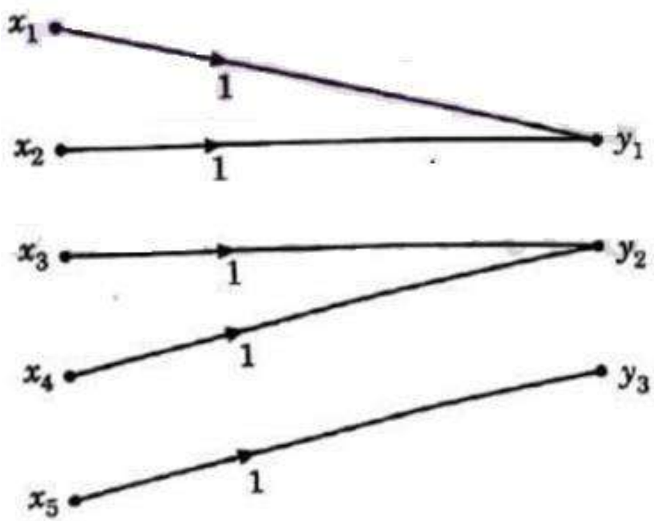
PHYSICAL LAYER – SOURCES ENCODING

Deterministic Channel

A Channel is described by a channel matrix with only one non-zero element in each row is called a deterministic channel. An example of deterministic channel has been shown and the corresponding channel matrix is given below:

It may be noted that since each row has only one non-zero element, therefore, this element must be unity by equation given below. Thus, when a given source symbols is sent in the deterministic channel, it is clear which output symbol will be received.

$$\sum_{j=1}^{n} P(y_j \mid x_i) = 1 \text{ for all } i$$

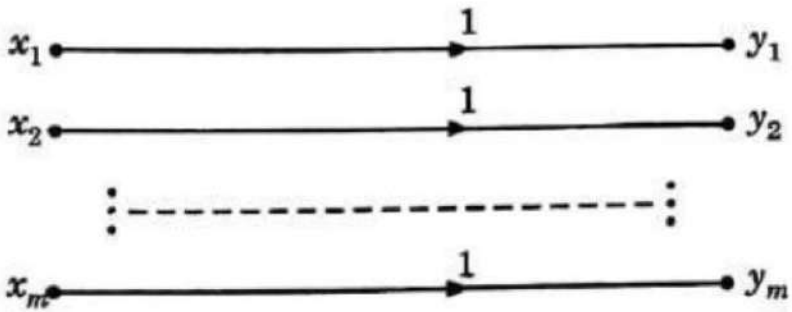


$$[P(Y \mid X)] = \begin{bmatrix} 1 & 0 & 0 \\ 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{bmatrix}$$

PHYSICAL LAYER – SOURCES ENCODING

Noiseless Channel

A Channel is called noiseless if it is both lossless and deterministic. A noiseless channel has been shown in the figure. The channel matrix has only one element in each row and each column, and this element is unity. Note that the input and output symbols are of the same size, i.e., m=n for the noiseless channel.



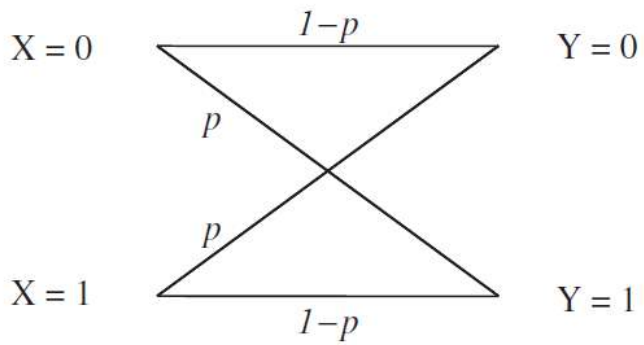
$$\begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

PHYSICAL LAYER – SOURCES ENCODING

Binary Symmetry Channel (BSC)

A Binary Symmetry Channel is defined by the channel diagram and its channel matrix given below:

$$P_{\text{BSC}} = \begin{bmatrix} (1-p) & p \\ p & (1-p) \end{bmatrix}$$



A BSC channel has two inputs (x1=0, x2=1) and two outputs (y1=0, y2=1). This channel is symmetry because the probability of receiving a 1 if a O is sent is the same as the probability of receiving a 0 if a 1 is sent,

BASIC ARCHITECTURAL FRAMEWORK

PHYSICAL LAYER

OUTLINES

- Basic Components
- Source Encoding
  - The Efficiency of a Source Encode
  - Pulse Code Modulation and Delta Modulation
- Channel Encoding
  - Types of Channels
  - Information Transmission over a Channel
  - Error Recognition and Correction
- Modulation
  - Modulation Types
  - Quadratic Amplitude Modulation
  - Summary
- Signal Propagation

PHYSICAL LAYER - OVERVIEW

- One of the desirable aspects of WSNs is their ability to communicate over a wireless link, so
  - mobile applications can be supported
  - flexible deployment of nodes is possible
  - the nodes can be placed in areas that are inaccessible to wired nodes

- Once the deployment is carried out, it is possible to
  - rearrange node placement - optimal coverage and connectivity
  - the rearrangement can be made without disrupting the normal operation

- Some formidable *challenges*:
  - limited bandwidth
  - limited transmission range
  - poor packet delivery performance because of interference, attenuation, and multi-path scattering

- therefore, it is vital to understand their properties and some of the mitigation strategies

- In this Chapter, we'll discuss a fundamental introduction to *point-to-point* wireless digital communication

COMMUNICATION SYSTEM – BASIC COMPONENTS

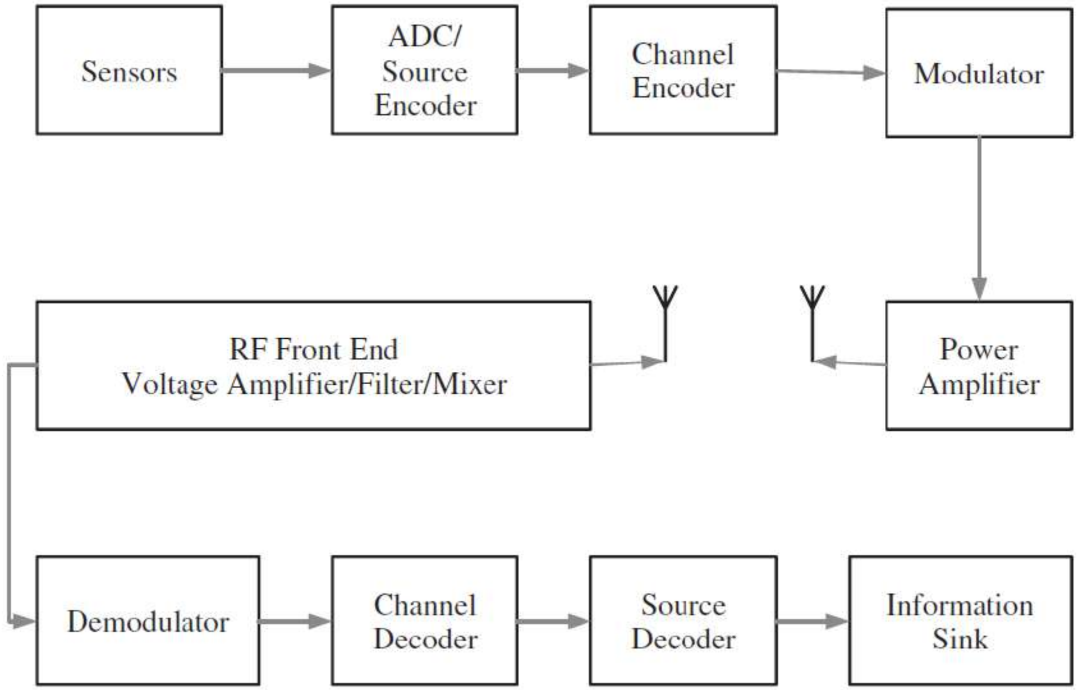
- Communication Systems deals with the flow of some sort of information in some network.

- Information my be electrical signals, words, pictures, music etc.

- The *basic components* of a digital communication system:
  - Transmitter or Source
  - Channel or Transmission Networks, which transfer the communique from transmitter to receiver
  - Receiver or Destination

- In WSN communication, we study the *short range communication* - because nodes are placed close to each other.

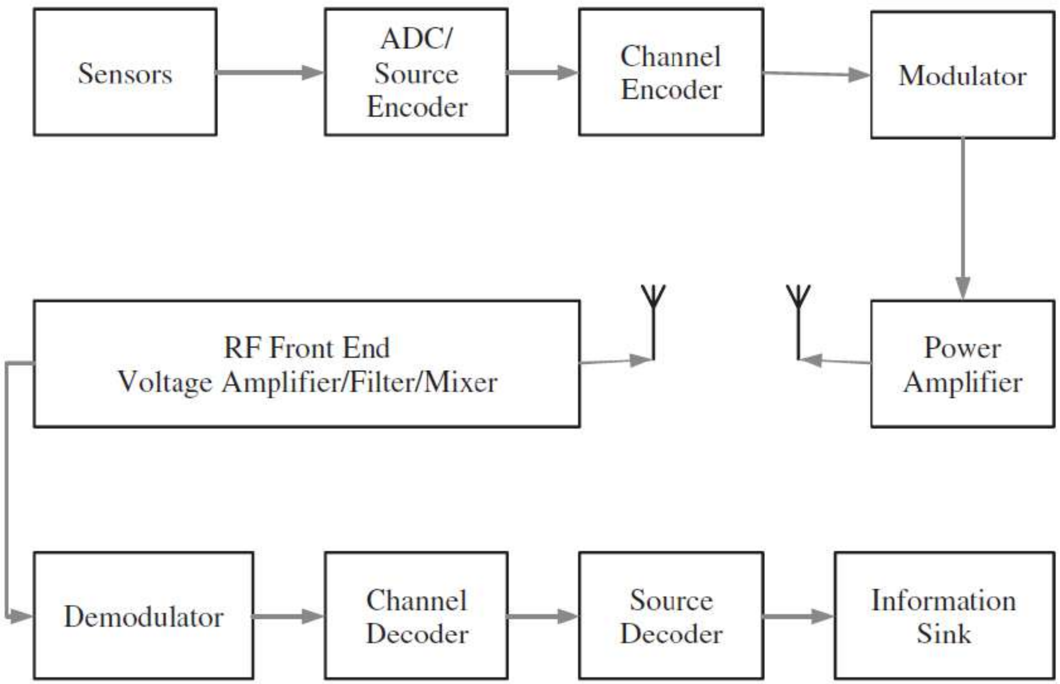
PHYSICAL LAYER – BASIC COMPONENTS



Components of a digital communication system.

- *The communication source* represents one or more sensors and produces *a message signal - an analog signal*
  - the signal is a *baseband* signal having dominant frequency components *near zero*
  - the message signal has to be converted to a discrete signal (*discrete* both in *time* and *amplitude*)

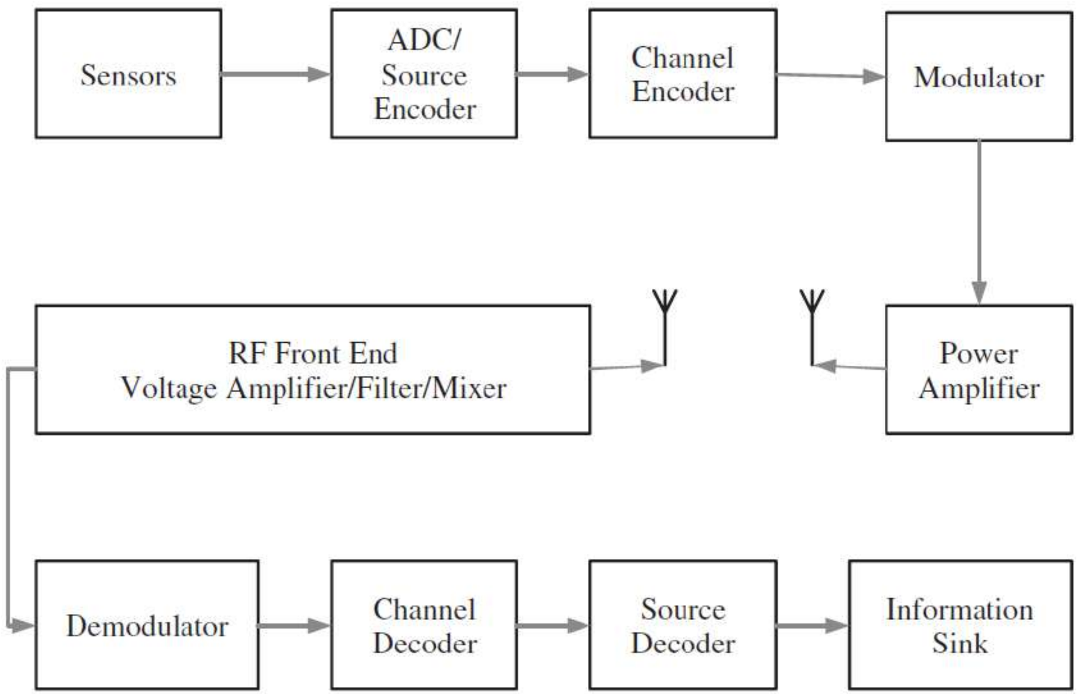
PHYSICAL LAYER – BASIC COMPONENTS



Components of a digital communication system.

- This conversion requires sampling the signal at least at *Nyquist Rate* - no information will be lost
  - The Nyquist rate sets a lower bound on the sampling frequency
  - Hence, the minimum sampling rate should be twice the bandwidth of the signal

PHYSICAL LAYER – BASIC COMPONENTS



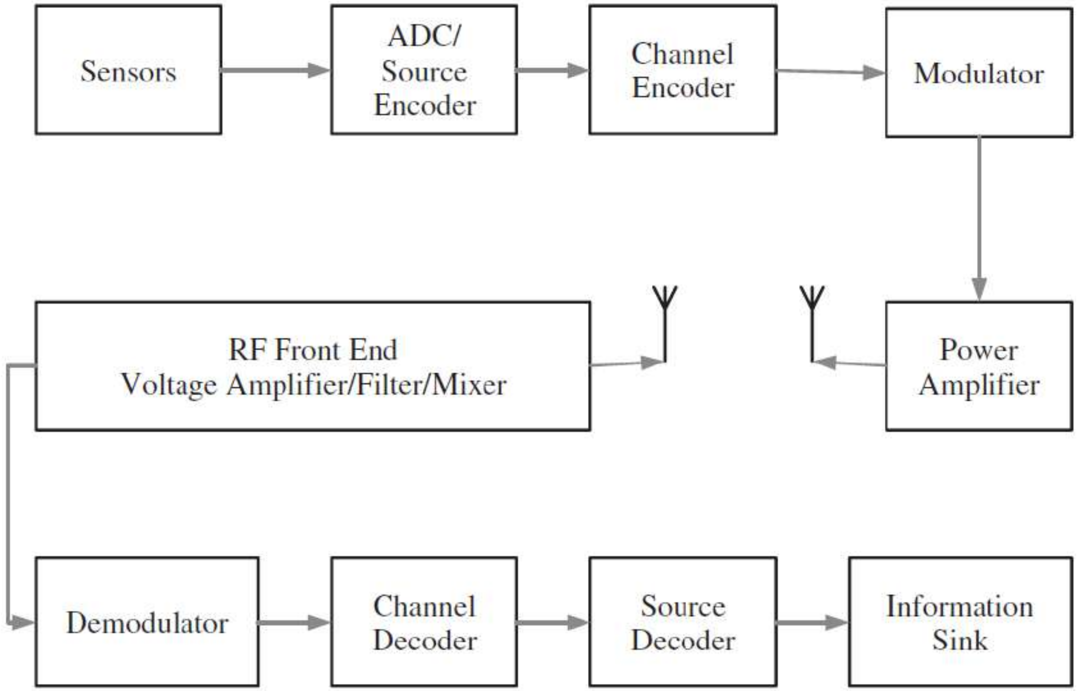
Components of a digital communication system.

***Source Encoding*:** the discrete signal is converted to a binary stream after sampling.

- An efficient source-coding technique can satisfy the channel's bandwidth and signal power requirements.

- It can be achieved by defining a probability model of the information source.
  - The length of each information symbol depends on its probability of occurrence.

PHYSICAL LAYER – BASIC COMPONENTS

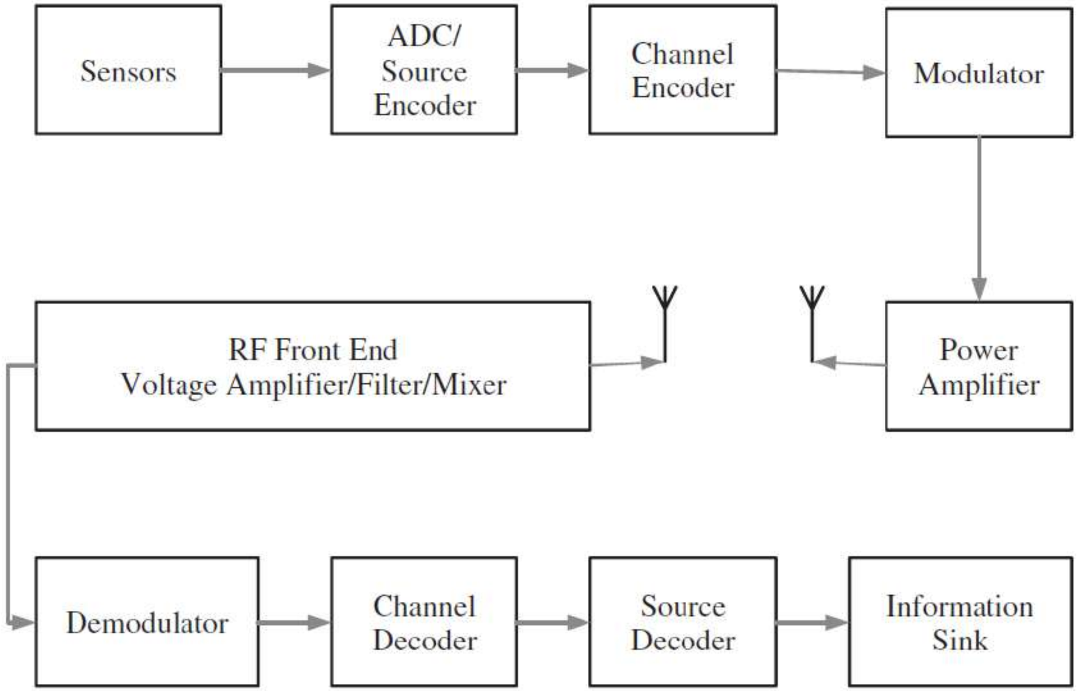


Components of a digital communication system.

Channel Encoding –

- Its aim is to make the transmitted signal robust to noise and interference.
- Moreover, in case of signal corruption, it enables an error to be recognized and the original data to be recovered.
- There are two essential approaches:
  - to transmit symbols from a predetermined codebook, and
  - to transmit redundant symbols.

PHYSICAL LAYER – BASIC COMPONENTS

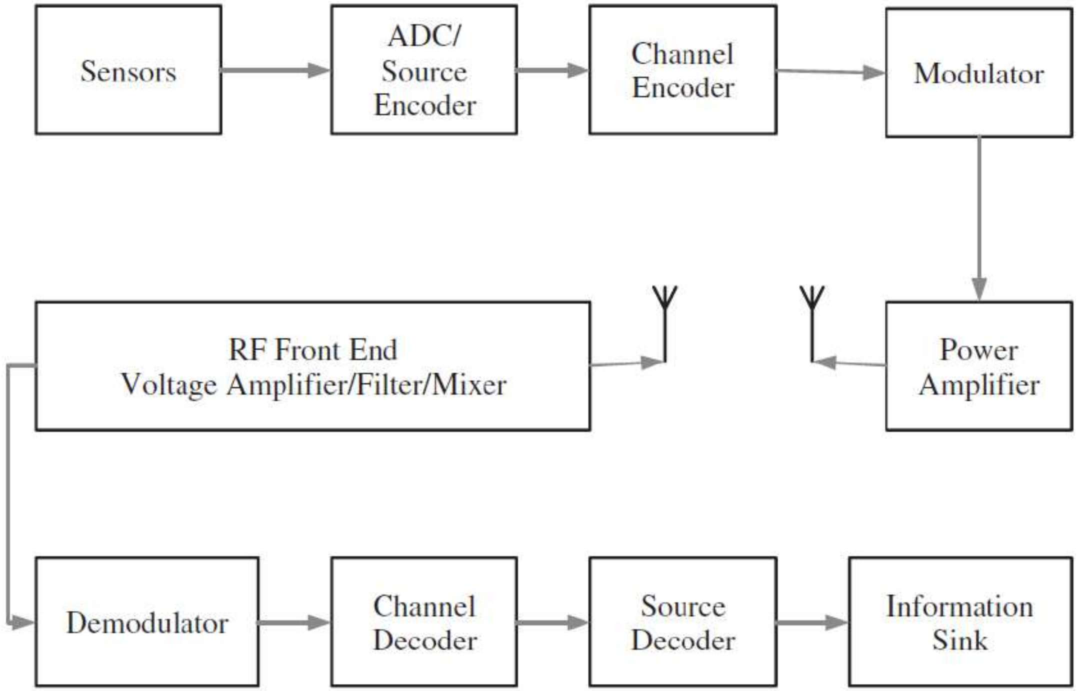


Components of a digital communication system.

Modulation–

- This is a process by which the baseband signal is transformed into a bandpass signal.
- Modulation is useful for various reasons, but the main reason is to transmit and receive signals with short antennas.
- In general, the shorter the wavelength of the transmitted signal, the shorter is the length of the antenna.

PHYSICAL LAYER – BASIC COMPONENTS



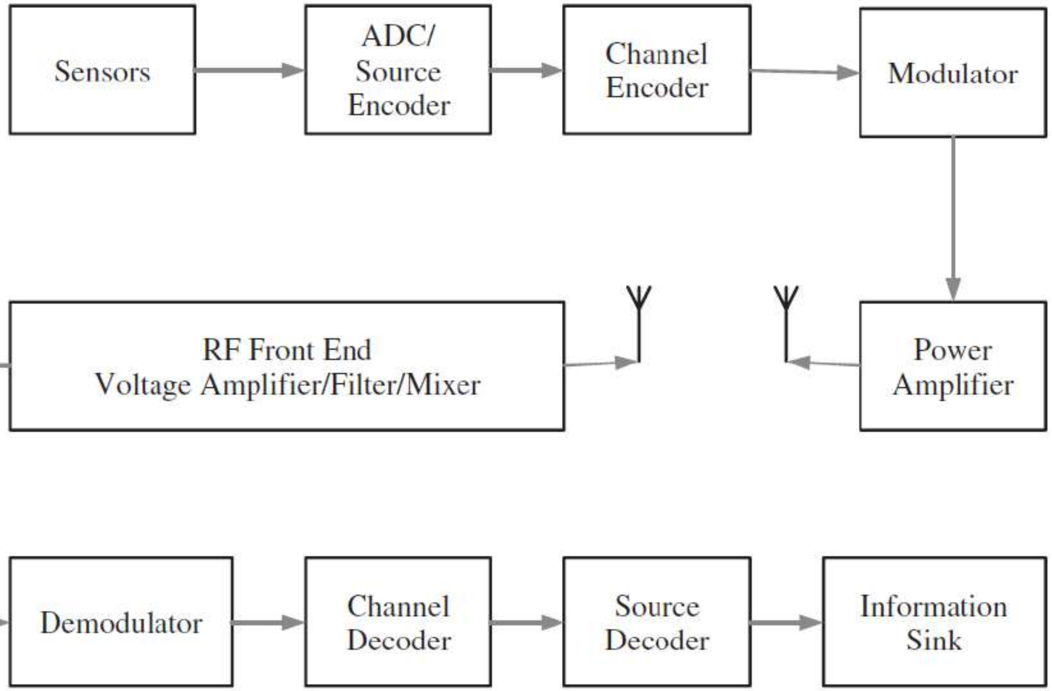
Components of a digital communication system.

Power Amplifier–

- The modulated signal has to be amplified and the electrical energy is converted into electromagnetic energy (electromagnetic radiation) by the transmitter's antenna, and

- The signal is propagated over a wireless link to the desired destination.

PHYSICAL LAYER – BASIC COMPONENTS



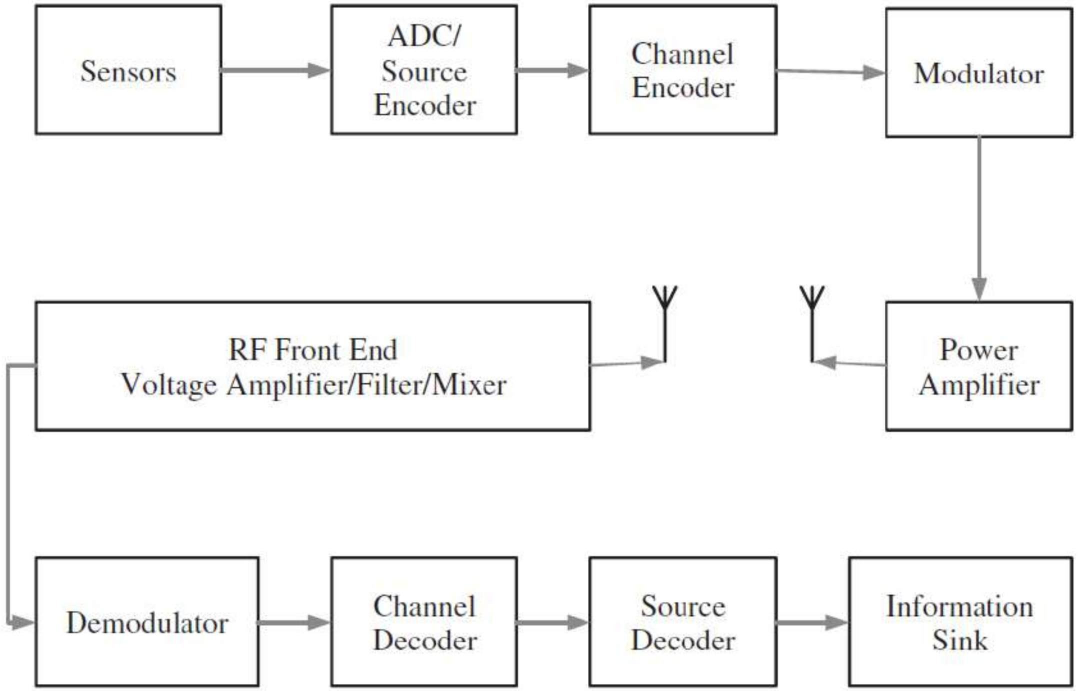
Components of a digital communication system.

Receiver End

- The components of the receiver block carry out the reverse process to retrieve the message signal from the electromagnetic waves. The receiver antenna induces a voltage that is, ideally, similar in shape, frequency, and phase with the modulated signal.

- The magnitude and shape of the signal are changed because of losses and interferences

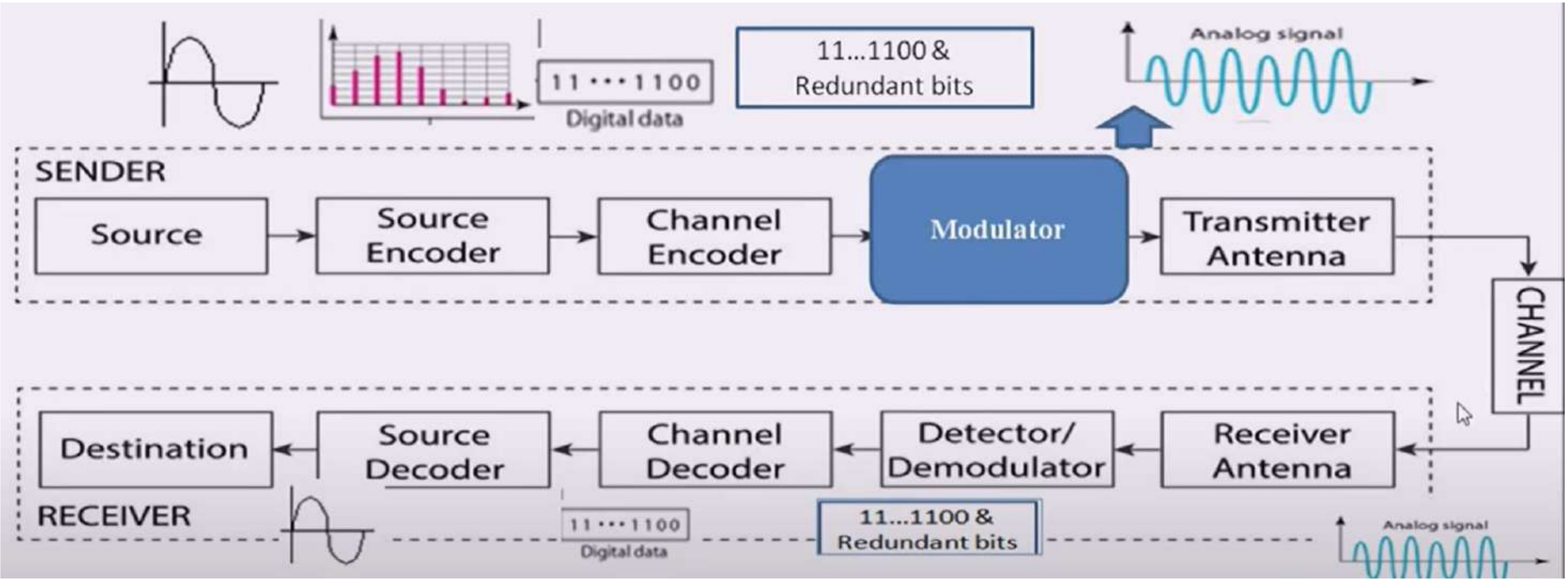
PHYSICAL LAYER – BASIC COMPONENTS



Components of a digital communication system.

- The signal has to pass through a series of *amplification* and *filtering processes*
- It is then transformed back to a baseband signal through the process of *demodulation* and *detection*
- Finally, the baseband signal undergoes *a pulse-shaping process* and *two* stages of *decoding* (channel and source)
  - extract the sequence of symbols - the original analog signal (the message)

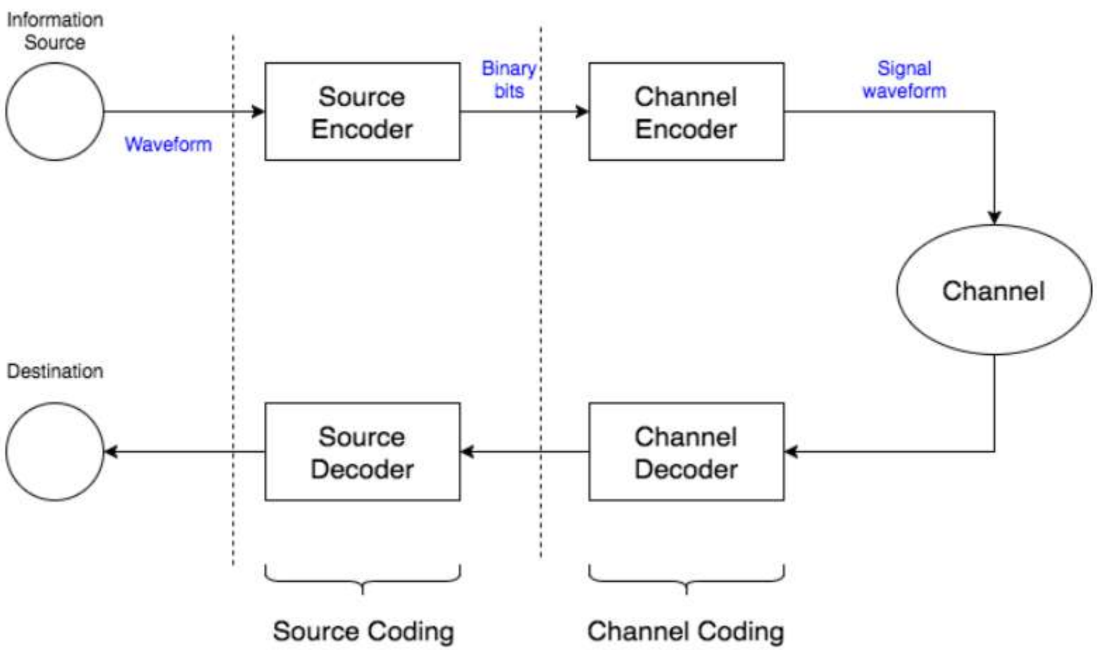
BLOCK DIAGRAM OF DIGITAL COMMUNICATION SYSTEM



BLOCK DIAGRAM OF DIGITAL COMMUNICATION SYSTEM

Digital communication systems are communication systems that use digital sequence (typically binary digits, that is, bits) as an interface between the source coding part and the channel coding part.

Source coding: The source encoder converts information waveforms to bits, while the decoder converts bits back to waveforms. The block diagram of source coding is depicted in Figure.



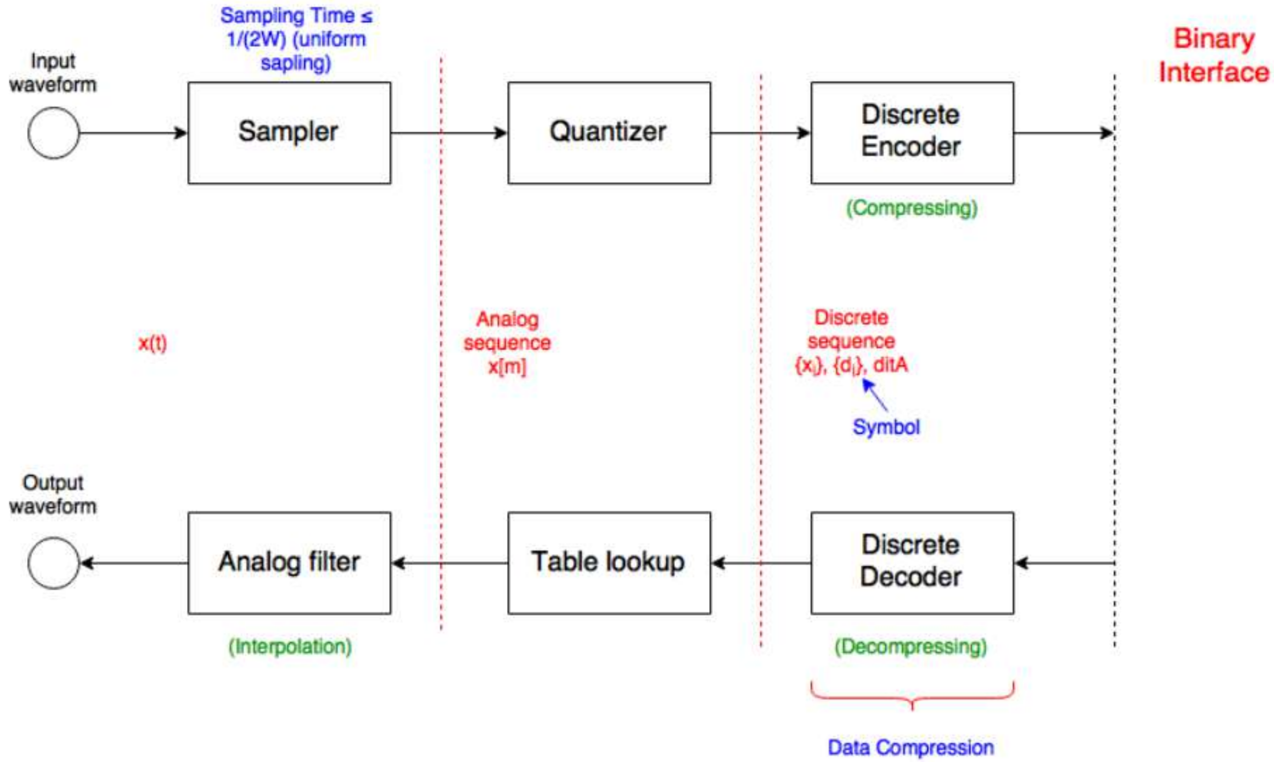
BLOCK DIAGRAM OF DIGITAL COMMUNICATION SYSTEM

Source Coding Source encoding aims to convert information waveforms (text, audio, image, video, etc.) into bits, the universal currency of information in the digital world. The three major steps are:

Sampling: convert the continuous-time analog waveform to discrete-time sequence (but still continuous-valued).

Quantization: convert each continuous-valued symbol to discrete-valued representatives.

Data compression: remove the redundancy in the data and generate roughly i.e. uniformly distributed bits.



PHYSICAL LAYER – SOURCES ENCODING

Source Encoding

- A source encoder transforms an *analog signal* into a *digital sequence*.

- The process consists of: *sampling, quantizing, encoding*

- Suppose a sensor produces an analog signal $s(t)$
- $s(t)$ will be sampled and quantized by the analog-to-digital converter (ADC) that has a resolution of Q distinct values
- as a result, a sequence of samples, $S = (s[1], s[2], \ldots, s[n])$ are produced
- the difference between the sampled $s[j]$ and its corresponding analog value at time t_j is the *quantization error*
- as the signal varies over time, the quantization error also varies and can be modeled as a random variable with a probability density function, $P_s(t)$

PHYSICAL LAYER – SOURCES ENCODING

Source Encoding

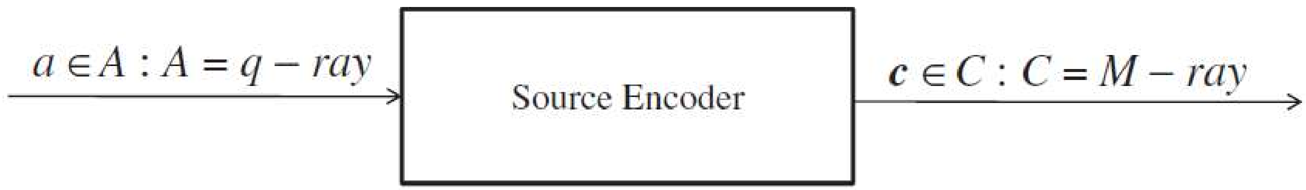
- The *aim* of the source encoder is to map each quantized element, *s[j]* into a corresponding binary symbol of length *r* from a codebook, *C*

- *Block code:* if all the binary symbols (code words) in the codebook are of *equal* length

- Often, the symbol length and the sampling rate are *not uniform*

- It is customary to assign:
    - *short-sized symbols* and *high* sampling rates to the most probable sample values
    - *long-sized symbols* and *low* sampling rates to less probable sample values

PHYSICAL LAYER – SOURCES ENCODING



Input – output relationship of a source encoder.

- A codebook, *C*, can be uniquely decoded, if each sequence of symbols, *(C(1), C(2), ...)* can be mapped back to a corresponding value in *S* = *(*s[1], s[2], ..., s[n]*)*

- A binary codebook has to satisfy the following equation to be uniquely decoded

$$\sum_{i=1}^{u}\left(\frac{1}{r}\right)^{l_i} \leq 1$$

  - where *u* is the size of the codebook
  - *li* is the size of the codeword *C(i)*
  - *r is the Binary Coding System i.e. 2.*

PHYSICAL LAYER – SOURCES ENCODING

- A codebook can be instantaneously decoded
  - if each symbol sequence can be extracted (decoded) from a stream of symbols *without* taking into consideration previously decoded symbols

- This will be possible
  - *if* there does *not exist* a symbol in the codebook, such that the symbol **a** = $(a_1, a_2, \ldots, a_m)$ is not a prefix of the symbol **b** = $(b_1, b_2, \ldots, b_n)$, where $m < n$ and $a_i = b_i$, $\forall\, i = 1, 2, \ldots, m$ within the same codebook

PHYSICAL LAYER – SOURCES ENCODING

Source-encoding techniques

	C^1	C^2	C^3	C^4	C^5	C^6
s_1	0	00	0	0	0	0
s_2	10	01	100	10	01	10
s_3	00	10	110	110	011	110
s_4	01	11	11	1110	111	111
Block code	No	Yes	No	No	No	No
Uniquely decoded	No	Yes	No	Yes	Yes	Yes
$\sum_{i=1}^{n} \left(\frac{1}{2}\right)^{l_i}$	$1\frac{1}{4}$	1	1	$\frac{15}{16} < 1$	1	1
Instantly decoded	No	Yes (block code)	No	Yes (comma code)	No	Yes

INFORMATION THEORY

Why Information Theory?

When the Communication thing is readily measurable, such as an electric current, the study of the communication system is relatively easy. But, when the communication thing is information, the study becomes rather difficult.

- Information Theory answers the following questions:
  - Measure for an amount of information.
  - Measure to improve the communication of information.

- ***Unit of information***
  - Communication systems are of *statistical* nature; i.e. the performance of the system can never be described in a deterministic sense.

  - Communication systems are *unpredictable* or *uncertain*.

  - An amount of information is calculated by statistical parameter associated with a *probability scheme*. The parameter should indicate a relative measure of uncertainty relevant to the occurrence of each message in the message ensemble.

INFORMATION THEORY

The amount of information is inversely proportional to the probability of an event. The more the probability of an event, the less is the amount of information associated with it, and vice versa.

$$I(x_j) = f\left[\frac{1}{p(x_j)}\right] \qquad \text{.............}Eq.\ 1$$

Where x_j is an event with a probability $p(xj)$ and the information associated with it is $I(x_j)$.

Now, let there be another event y_k such that x_j and y_k are independent. Hence the probability of the joint event is $p(x_j, y_k) = p(xj)\ p(y_k)$ with associated information content

$$I(x_j, y_k) = f\left[\frac{1}{p(x_j, y_k)}\right] = f\left[\frac{1}{p(xj)\ p(y_k)}\right] \qquad \text{..................}Eq.\ 2$$

The total information $I(x_j, y_k)$ must be equal to the sum of individual information $I(x_j)$ and $I(y_k)$. Where,

$$I(y_k) = f\left[\frac{1}{p(y_k)}\right]$$

Thus, it can be seen that the function on RHS of Eq. 2 must be a function which converts multiplication into addition. Logarithm is one such function. Thus,

INFORMATION THEORY

Thus, it can be seen that the function on RHS of Eq. 2 must be a function which converts multiplication into addition. Logarithm is one such function. Thus,

$$I(x_j, y_k) = log\left[\frac{1}{p(xj)\ p(y_k)}\right]$$

$$= log\left[\frac{1}{p(xj)}\right] + log\left[\frac{1}{p(y_k)}\right]$$

$$= I(x_j) + I(y_k)$$

Hence, the basic equation defining the amount of information (or self information) is

$$I(x_j) = log\left[\frac{1}{p(xj)}\right] = -\ log[p(xj)]$$

Different units of information can be defined for different bases of logarithms.

Base 2 :*Unit is bit*

Base e :*Unit is nat*

Base 10 :*Unit is decit*

In Information theory, unit of information is bit. So, it is assumed to take base 2.

INFORMATION THEORY

An intuitive and meaningful measure of information should have the following properties:

- Self information should decrease with increasing probability.
- Self information of two independent events should be their sum.
- Self information should be a continuous function of the probability.

The only function satisfying the above conditions is the -log of the probability.

Entropy

When we observe the possibilities of the occurrence of an event, how surprising or uncertain it would be, it means that we are trying to have an idea on the average content of the information from the source of the event.

"Entropy is a measure of uncertainty."

Entropy can be defined as a ***measure of the average information content per source symbol***. **Claude Shannon**, the "father of the Information Theory", provided a formula for it as –

$$H = \sum_{i=1}^{M} p_i \log_2 \frac{1}{p_i}$$

INFORMATION THEORY

If someone is sending us a stream symbols from a source (such as those above), we generally have some uncertainty about the sequence of symbols we will be receiving. If not, there would no need for the transmitter to even bother sending us the data.

A useful and common way to model this uncertainty is to assume that the data is coming randomly according to some probability distribution.

The simplest model is to assume that each the symbols $s_1, \ldots, s_M$ have associated probabilities of occurrence $p_1, \ldots, p_M$, and we see a string of symbols drawn independently according to these probabilities. Since the p_i are probabilities, we have $0 \leq p_i \leq 1$ for all i.

Also, we assume the only possible symbols in use are the $s_1, \ldots, s_M$, and so

$$p_1 + \cdots + p_M = 1.$$

INFORMATION THEORY

How much information does a source provide?

Consider just one symbol from the source. The probability that we observe the symbol s_i is p_i , so that if we do indeed observe s_i then we get $\log \frac{1}{p_i}$ bits of information. Therefore, the average number of bits of information we get based on observing one symbol is

$$p_1 \log_2 \frac{1}{p_1} + \cdots + p_M \log_2 \frac{1}{p_M} = \sum_{i=1}^{M} p_i \log_2 \frac{1}{p_i}$$

This is an important quantity called the entropy of the source and is denoted by H.

Definition

Given a source that outputs symbols $s_1, \ldots, s_M$ with probabilities $p_1, \ldots, p_M$, respectively, the entropy of the source, denoted H, is defined as

$$H = \sum_{i=1}^{M} p_i \log_2 \frac{1}{p_i}$$

Case I:

If there is only one possible message, i.e. M=1 and p_k = P = 1, then

$$H = p1 \log[1/p_1] = 1 \log[1/1] = 0$$

In this case of single possible message, the reception of total message conveys no information.

Case 2:

Only one message out of M having a probability 1 and others have 0.

$$H = \sum_{k=1}^{M} p_k \log_2 \frac{1}{p_k}$$

$$H = p_1 \log \frac{1}{p1} + \lim_{p \to 0} \left[p \log \frac{1}{p} + p \log \frac{1}{p} + \cdots \ldots \ldots . . \right]$$

$$H = 1 \log [1/1] + 0$$

$$H = 0$$

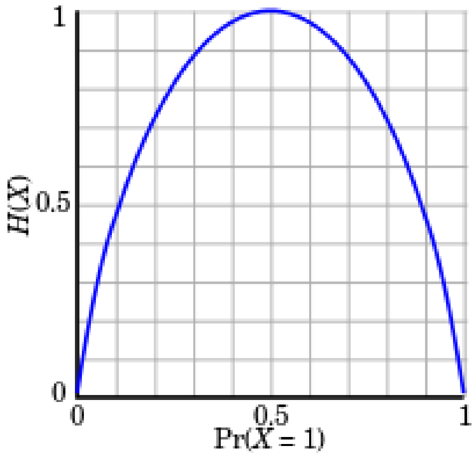
In this case, entropy is zero.

Case 3:

For Binary System (M=2), the entropy is

$$H = p_1 \log\frac{1}{p1} + p_2 \log\frac{1}{p2}$$

Let p1 = p then p2 = 1 – p1 = 1 – p = q



$$H = p \log\frac{1}{p} + (1-p) \log\frac{1}{(1-p)}$$

$$H = p \log\frac{1}{p} + q \log\frac{1}{q}$$

The maximum value of H can be found by putting p = 1/2

$$H_{max} = 1/2 \log\frac{1}{1/2} + 1/2 \log\frac{1}{1/2} = 1 \text{ bit/message}$$

In this case, we have seen that the entropy is maximum when p = 1/2, i.e. both the message are equally probable. In this case, the maximum entropy is then

$$H_{max} = \sum_{k=1}^{M} p_k \log_2\frac{1}{p_k} = \sum_{k=1}^{M} \frac{1}{M} \log M = \log M \text{ bits/message}$$

INFORMATION THEORY

Rate of Information

If a message source generates messages at the rate of r messages per second, the rate of information R is defined as the average number of bits of information per second. Now, H is the average number of bits of information per message. Hence

$$R = r\,H \ \text{bits/sec}$$

INFORMATION THEORY

- The importance of the entropy of a source lies in its operational significance concerning coding the source. Since *H* represents ***the average number of bits of information per symbol from the source,***

- We might expect that we need to use at least *H* bits per symbol to represent the source with a uniquely decodable code.

- This is in fact the case, and moreover, if we wish to code longer and longer strings of symbols, we can find codes whose performance (average number of bits per symbol) gets closer to H. This result is called the ***Source Coding Theorem*** and was discovered by ***Shannon*** in 1948.

- ***Source Coding Theorem***
  a) The average number of bits/symbol of any uniquely decodable source must be greater than or equal to the entropy H of the source.

  a) If the string of symbols is sufficiently large, there exists a uniquely decodable code for the source such that the average number of bits/symbol of the code is as close to H as desired.

PHYSICAL LAYER – SOURCES ENCODING

The Efficiency of a Source Encoder

- Quantity that expresses the average length
- Sampled analog signal: $L(C) = E[li(C)]$
- Suppose the probability of a m-ary source
  - i.e., it has q distinct symbols
  - producing the symbol si is Pi and the symbol Ci in a codebook is used to encode si
  - the expected length of the codebook is given by:

$$L(C) = \sum_{i=1}^{m} P_i * l_i(C)$$

The Efficiency of a Source Encoder

- To express efficiency in terms of the information entropy or *Shannon's entropy*
  - defined as *the minimum message length* necessary to communicate information
  - related to the *uncertainty* associated with the information
  - if the symbol *si* can be expressed by a binary symbol of *n* bits, the information content of *si* is:

$$I(s_i) = -\log_2 P_i = \log_2 \frac{1}{P_i}$$

  - Logarithmic Measure of information possesses the desired additive property when a number of source outputs is considered as a block.
  - the entropy (in bits) of a *m*-ary memoryless source encoder is expressed as:

$$H = \sum_{i=1}^{M} p_i \log_2 \frac{1}{p_i}$$

The Efficiency of a Source Encoder

- The efficiency of a source encoder in terms of entropy reveals the unnecessary redundancy in the encoding process. This can be expressed by
- Average information per message/information actually transmitted per message

$$\eta = \frac{\text{Actual Transinformation}}{\text{Maximum Transinformation}}$$

$$\eta = \frac{I(X;Y)}{C} = \frac{I(X;Y)}{Max\ I(X;Y)}$$

$$\eta(C) = \frac{H(S)}{L(C)}$$

Where,
H(S) is the Entropy of the Samples (Actual Transinformation)
L(C) is the Entropy of the Code Block (Maximum Transinformation)

- Transinformation is a measure of the average information per symbol transmitted in the system. i.e. the rate of transmission of information.

- The redundancy of the encoder is: $\frac{L - H(S)}{L} = 1 - \eta$

Example 1

Message	Probability	Code	Length of Code
m1	1/2	C1 = 00	l1=2
m2	1/4	C2= 01	l2=2
m3	1/8	C3= 10	l3=2
m4	1/8	C4=11	l4=2

- It is quantized into four distinct values, 0*, 1, 2, 3* .
- if the *probability* of occurrence of these values is *P(1)* = 0.5, *P(2)* = 0.25, *P(3)* = 0.*125*, *P(4)* = 0.*125*, then, it is possible to compute the efficiency of the codebooks given in Table
- *li has been taken equal for* each symbol. Hence:

$$L(C) = \sum_{i=1}^{4} l_i * P_i = (2 * 1/2) + (2 * 1/4) + (2 * 1/8) + (2 * 1/8) = 2 \text{ bits/message}$$

Example 1

- Using Equation, the entropy of *Codebook* is calculated as:

$$H(C) = \frac{1}{2}\log\frac{1}{2} + \frac{1}{4}\log\frac{1}{4} + \frac{1}{8}\log\frac{1}{8} + \frac{1}{8}\log\frac{1}{8} = 7/4 = 1.75$$

$$\eta(C) = \frac{H(C)}{L(C)} = \frac{1.75}{2} = 87.5\ \%$$

- Therefore, the encoding efficiency of the codebook, *C2* (see Table) is:

$$\eta(C) = 87.5\ \%$$

- The redundancy in *C2* is:

$$R(C) = 1 - \eta = 1 - 0.875 = 0.125 = 12.5\%$$

- In terms of energy efficiency, this implies that 12.5*% of the transmitted bits are unnecessarily redundant*, because *C* is not compact enough.

Example 1

Message	Probability	Code	Length of Code
m1	1/2	C1 = 0	l1=1
m2	1/4	C2= 10	l2=2
m3	1/8	C3= 110	l3=3
m4	1/8	C4=111	l4=3

- It is quantized into four distinct values, 0, *1, 2, 3* .
- if the *probability* of occurrence of these values is *P(1)* = 0.5, *P(2)* = 0.25, *P(3)* = 0.*125*, *P(4)* = 0.*125*, then, it is possible to compute the efficiency of the codebooks given in Table
- *li has been taken different for* each symbol. Hence:

$L(C) = \sum_{i=1}^{4} l_i * P_i$ = (1 * 1/2) + (2 * 1/4) + (3 * 1/8) + (3 * 1/8) = 1.75 bits/message

Example 2

- Using Equation, the entropy of *Codebook* is calculated as:

$$H(C) = \frac{1}{2}\log\frac{1}{2} + \frac{1}{4}\log\frac{1}{4} + \frac{1}{8}\log\frac{1}{8} + \frac{1}{8}\log\frac{1}{8} = 7/4 = 1.75$$

$$\eta(C) = \frac{H(C)}{L(C)} = \frac{1.75}{1.75} = 100\ \%$$

- Therefore, the encoding efficiency of the codebook, *C2* (see Table) is:

$$\eta(C) = 87.5\ \%$$

- The redundancy in *C2* is:

$$\mathbf{R\ (C) = 1 - \eta} = 1 - 1 = 0 = 0\%$$

- In terms of energy efficiency, this implies that no bits are *transmitted as redundant bits*, because *C* is perfect enough and efficient codebook.

PHYSICAL LAYER – SOURCES ENCODING

Source Coding Techniques

1. Huffman Code
2. Lempel-Ziv Code
3. Shannon-Fano Code
4. Arithmetic Code

Pulse Code Modulation & Delta Modulation

PCM and DM are the two predominantly employed *source encoding* techniques

- *In digital pulse code modulation*
    - the signal is quantized first
    - each sample is represented by a binary word from a finite set of words

- The resolution of a PCM technique and the source encoder bit rate are determined by
    - the size of the individual words
    - the number of words in the set

PHYSICAL LAYER – SOURCES ENCODING

Pulse Code Modulation

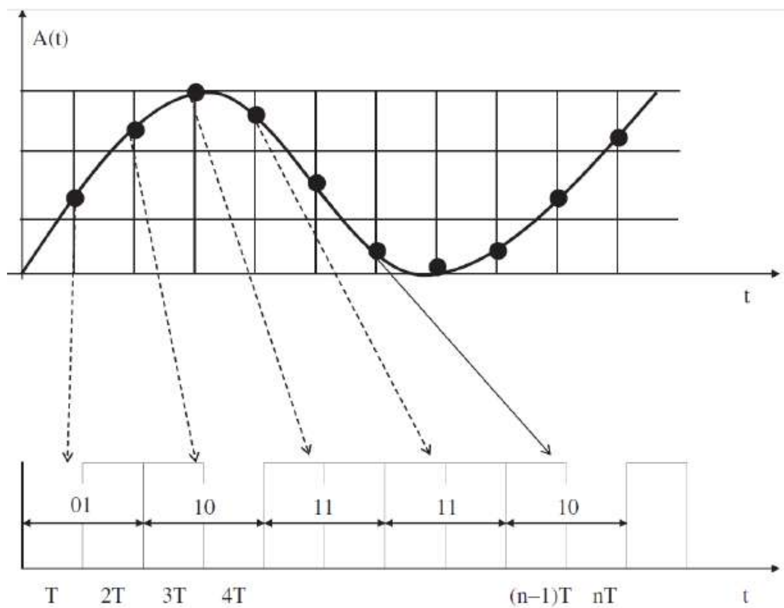
In PCM, information is conveyed in the presence or absence of *pulses*

- greatly enhances the transmission and regeneration of *binary words*

- the associated cost with this form of source encoding is
  - the quantization error, the energy and bandwidth required to transmit the multiple bits for each sampled output!

- Figure illustrates a PCM technique that uses two bits to encode a single sample

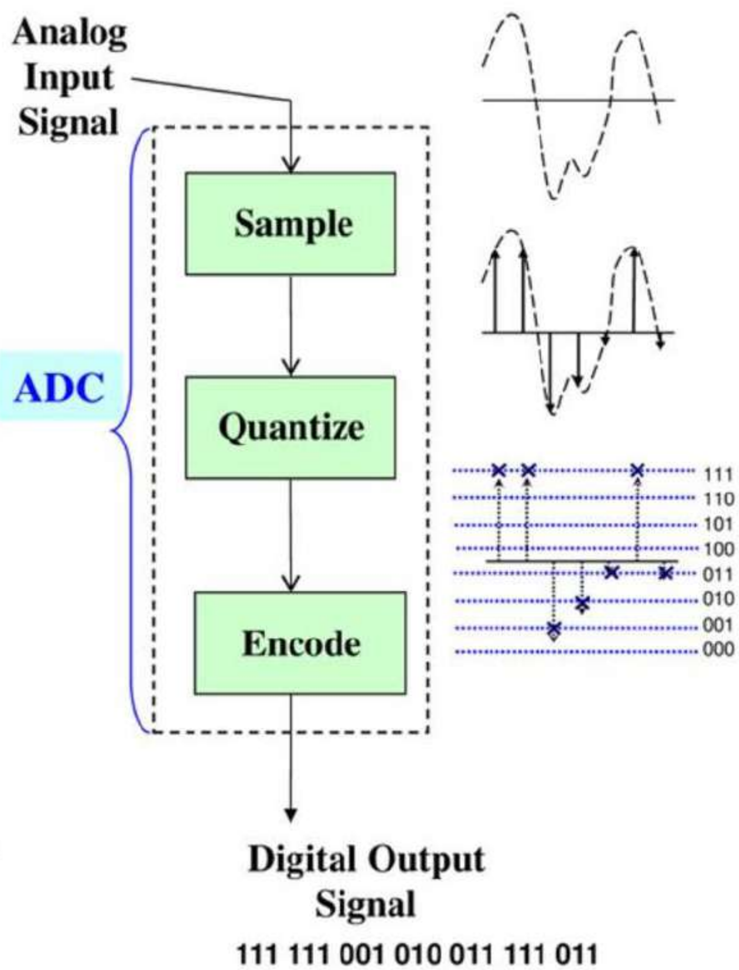
- Four distinct levels are permissible during sampling



A PCM based source encoding.

PHYSICAL LAYER – SOURCES ENCODING

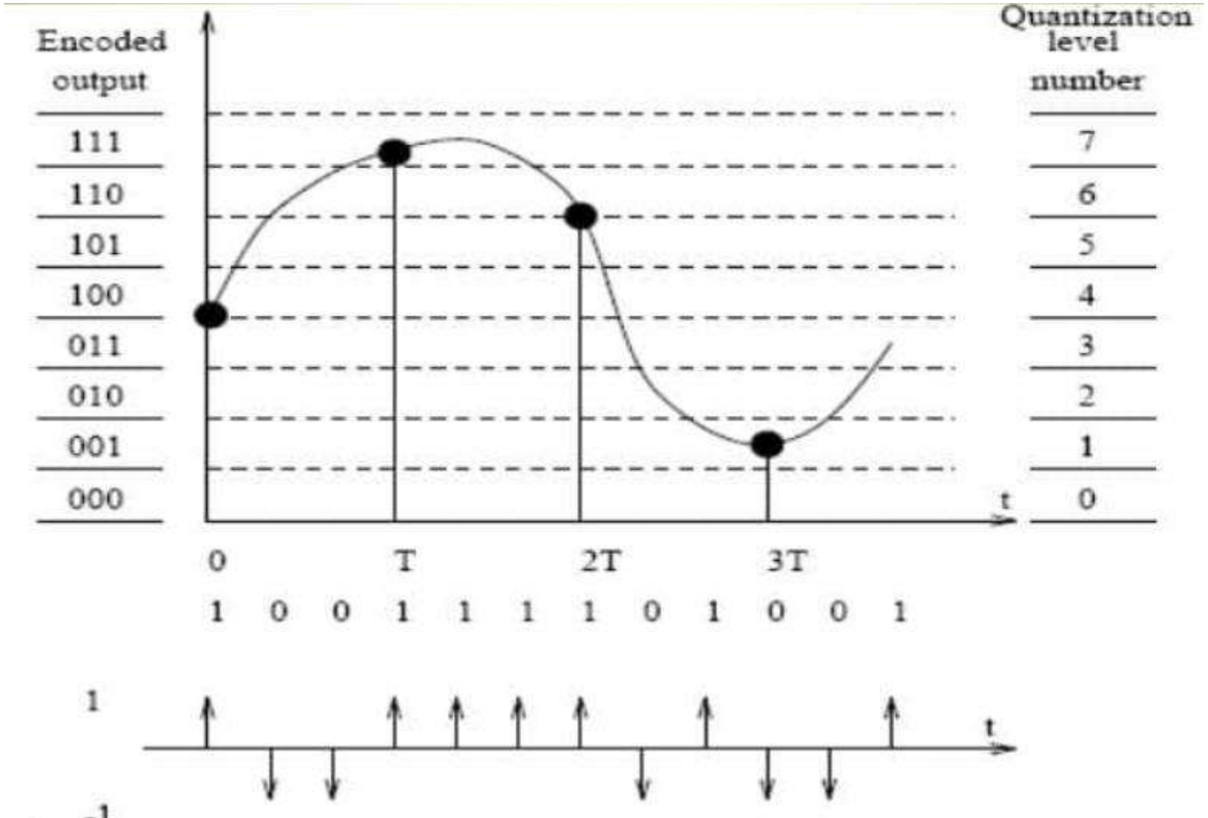
Pulse Code Modulation



- The *Analog-to-digital Converter* (ADC) performs three functions:
  - **Sampling**
    - Makes the signal discrete in time.
    - If the analog input has a bandwidth of *W* Hz, then the ***minimum sample frequency*** such that the signal can be reconstructed without distortion is 2W or more.
  - **Quantization**
    - Makes the signal discrete in amplitude.
    - Round off to one of *q* discrete levels.
  - **Encode**
    - Maps the quantized values to digital words that are ν bits long.
- If the (Nyquist) ***Sampling Theorem*** is satisfied, then only quantization introduces distortion to the system.

PHYSICAL LAYER – SOURCES ENCODING

Pulse Code Modulation



PHYSICAL LAYER – SOURCES ENCODING

Pulse Code Modulation

Level	Code word
0	000
1	001
2	010
3	011
4	100
5	101
6	110
7	111

Levels are encoded using this table

Table: Quantization levels with belonging code words

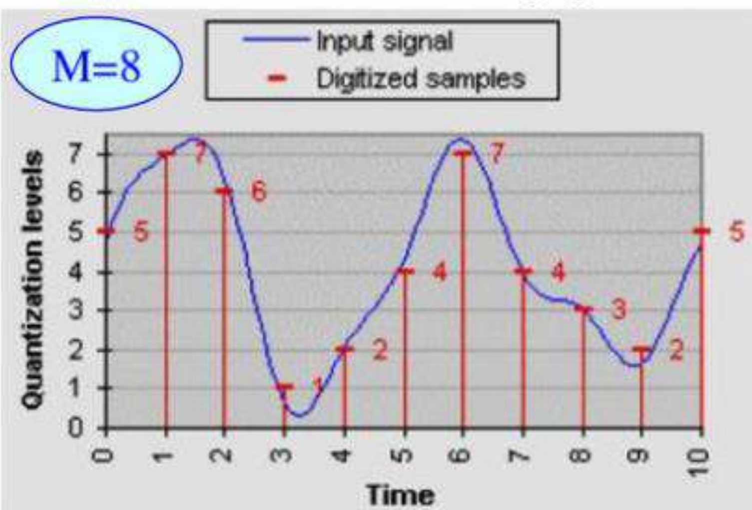


Chart 1. Quantization and digitalization of a signal.

Signal is quantized in 11 time points & 8 quantization segments.

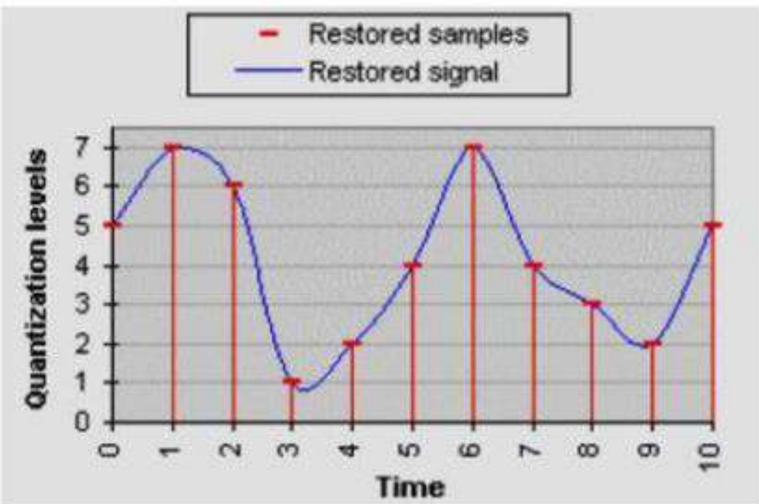


Chart 2. Process of restoring a signal.

PCM encoded signal in binary form:

101 111 110 001 010 100 111 100 011 010 101

Total of **33** bits were used to encode a signal

Delta Modulation

- Linear PCM (LPCM) is a linear quantization PCM.

- Differential PCM (DPCM) encodes PCM values as the difference between the current value and the expected value. The algorithm predicts the next sample based on the previous samples, and the encoder only stores the differentiation between this projection and the actual value. If the projection is reasonable, less bits can be used to represent the same information. For audio, this type of encoding decreases the number of bits required per sample by about 25% compared to PCM.

- Adaptive DPCM (ADPCM) is a variation of DPCM that changes the size of the quantization steps to allow further reduction in the bandwidth required for a given signal-to-noise ratio.

- Delta modulation is a form of DPCM which uses one bit per sample to indicate whether the signal is increasing or decreasing from the previous sample.

Delta Modulation

Delta modulation is a *digital pulse modulation* technique

- it has found widespread acceptance *in low bit rate* digital systems
- it is a *differential* encoder and transmits bits of information
- the information describes *the difference between successive* signal values, as opposed to the actual values of a time-series sequence
- the difference signal, $V_d(t)$, is produced by first estimating the signal's magnitude based on previous samples ($V_i(t_0)$) and comparing this value with the actual input signal, $V_{in}(t_0)$
- The polarity of the difference value indicates the polarity of the pulse transmitted

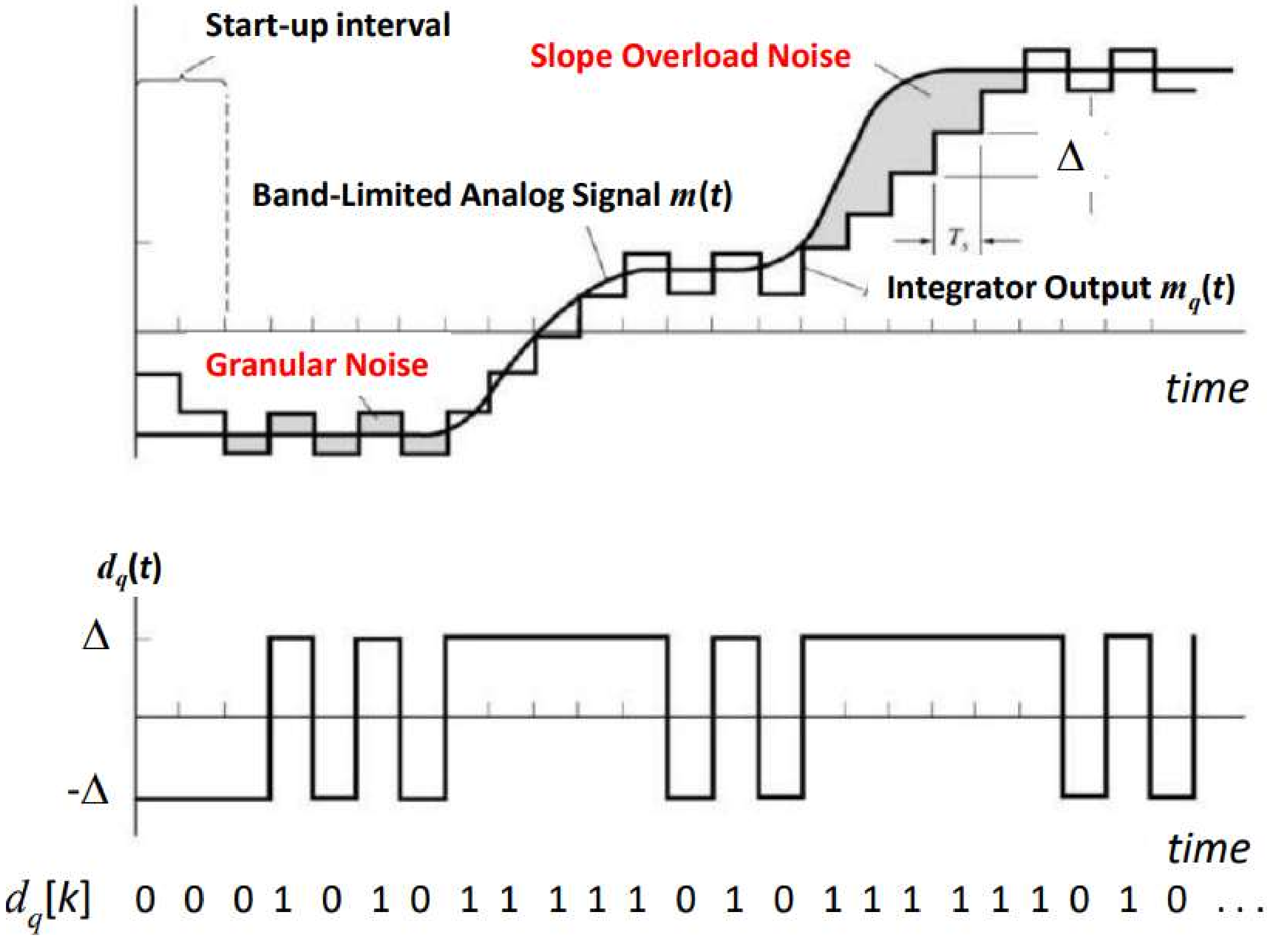
Delta Modulation

Delta modulation is a *digital pulse modulation* technique

- Delta modulation (DM) is the simplest method for analog-to-digital conversion (ADC).
- Delta modulation uses 1-bit per sampling period (TS) – it is a 1-bit ADC.
-
- Delta modulation requires a sampling rate much greater than the Nyquist rate (commonly four or five times the Nyquist rate).
- DM is closely related to DPCM.
- In DM we use a first-order predictor (one time delay TS is the predictor).
- DM uses very simple hardware and is low cost for that reason.
- The transmitted output is a binary stream of $\pm\Delta$ pulses at f_S. It gives a stepwise approximation $m_q(t)$ to $m(t)$.

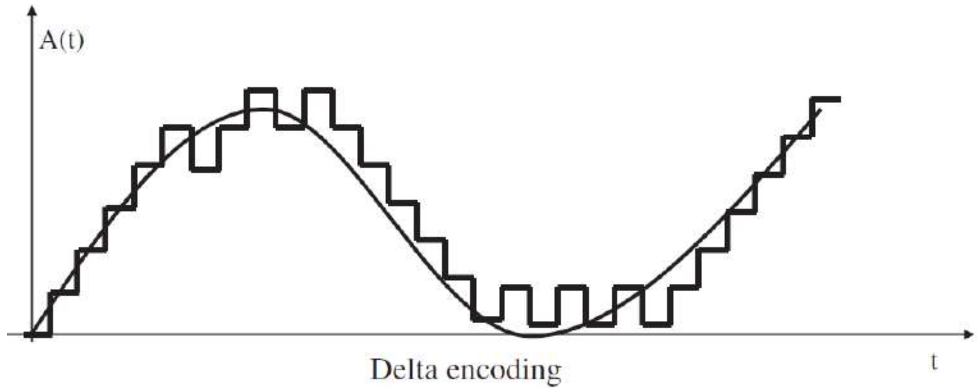
Delta Modulation

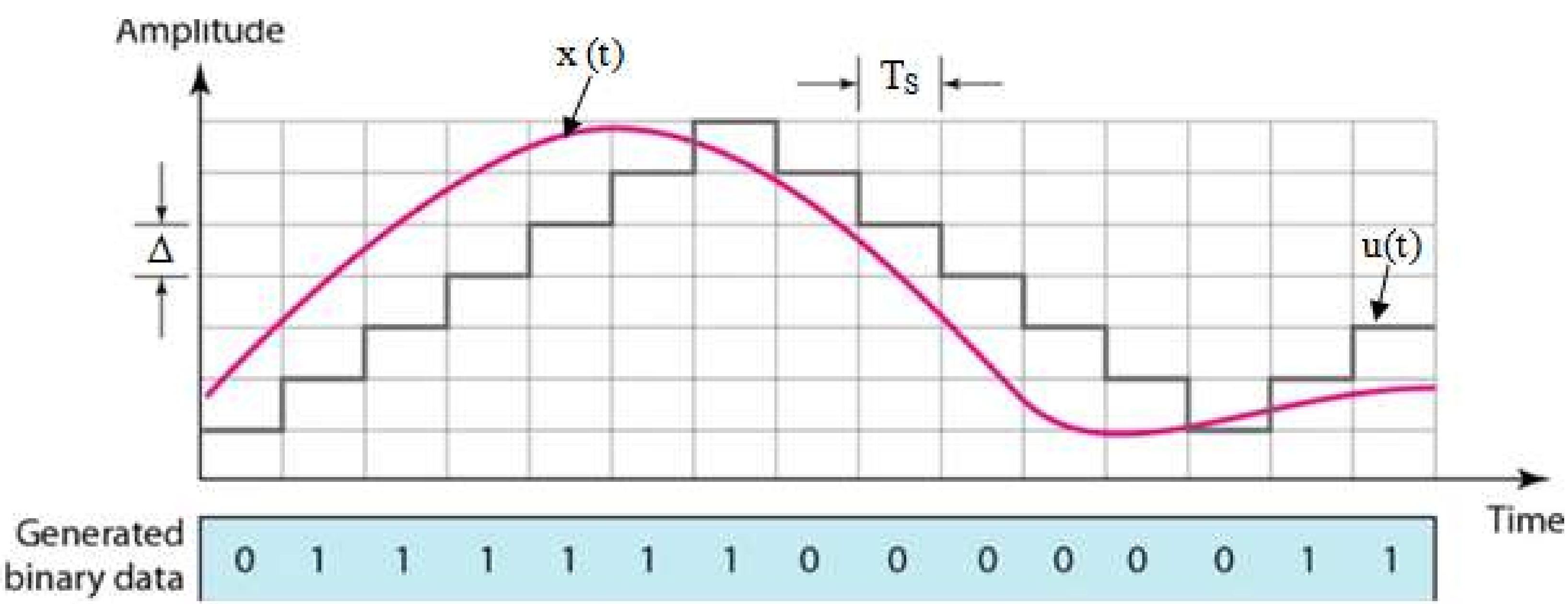
- The difference signal is a *measure* of the *slope of the signal*
  - first, *sampling* the analog signal
  - then, *varying* the amplitude, width, or the position of the digital signal in accordance with the amplitude of the sampled signal



PHYSICAL LAYER – SOURCES ENCODING

Delta Modulation





PHYSICAL LAYER – CHANNEL ENCODING

Channel Encoding

- To produce a sequence of data that is robust to noise.

- To provide error detection and forward error correction mechanisms.

- In simple and cheap transceivers, **forward error correction** is costly and, therefore, the task of channel encoding is limited to the detection of errors in packet transmission.

- The noise present in a channel creates unwanted errors between the input and the output sequences of a digital communication system. The error probability should be very low, **nearly $\leq 10^{-6}$** for a reliable communication.

- The **channel coding** in a communication system, **introduces redundancy** with a control, so as to improve the reliability of the system.

- The **source coding reduces redundancy** to improve the efficiency of the system.

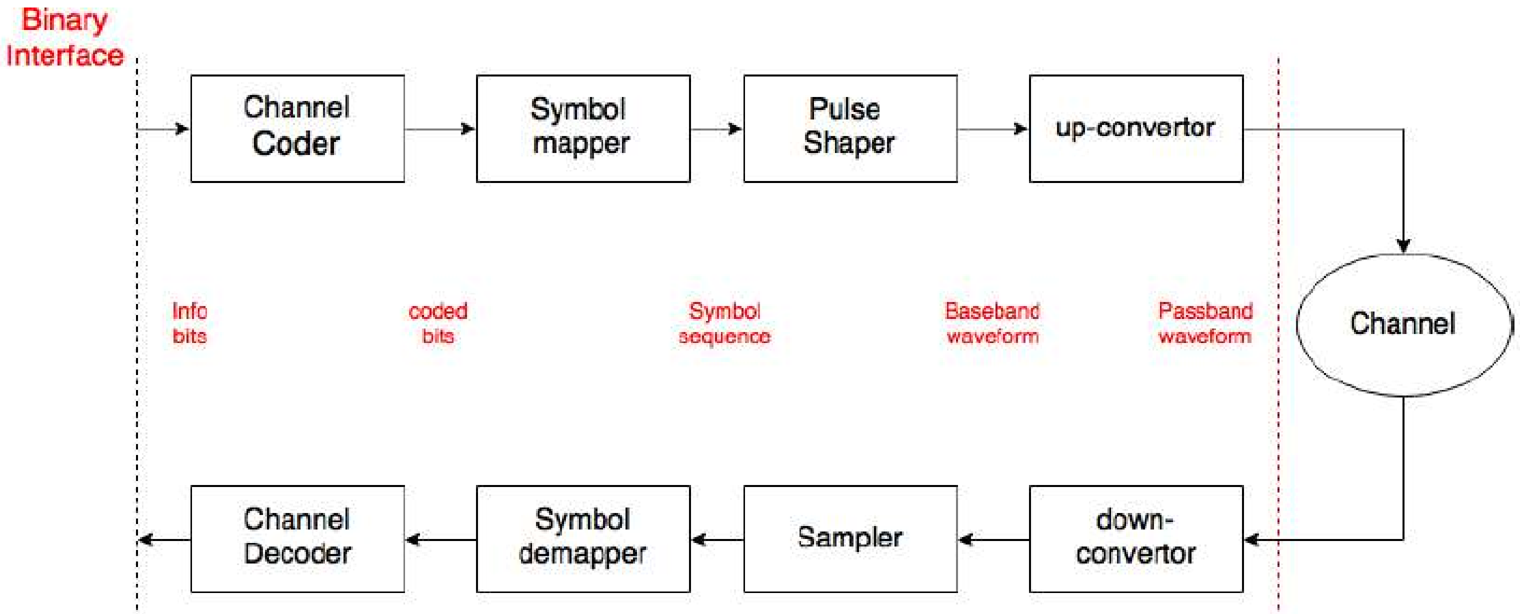
PHYSICAL LAYER – CHANNEL ENCODING

Channel Encoding

- Channel coding consists of two parts of action.
  1. **Mapping** incoming data sequence into a channel input sequence. The mapping is done by the transmitter, with the help of an encoder.

  2. **Inverse Mapping** the channel output sequence into an output data sequence. The inverse mapping is done by the decoder in the receiver.

- The final target is that the overall effect of the **channel noise** should be minimized.



PHYSICAL LAYER – CHANNEL ENCODING

Channel Encoding

- The four major steps are:
  1. **Error correcting codes:** introduce redundancy into the information bits and produce longer coded bits.
  2. **Symbol mapping:** map the coded bits to *constellation points*, each of which is a complex symbol.
  3. **Pulse shaping:** modulate the symbol to suitable *baseband* waveforms.
  4. **Up conversion:** convert the baseband waveform to *passband* waveform, so that the effective frequency band follows the constraints from the physical world.
- Channel decoding does the reverse of encoding.
- These Error Correcting Techniques are also called Error-Control Coding:
  1. Block Code (Parity Check Code, Binary Code Space, Linear Block Code)
  2. Hamming Code
  3. Cyclic Codes
  4. Convolutional Codes

PHYSICAL LAYER – CHANNEL ENCODING

Shannon Channel Coding Theorem

- It is related with the rate of transmission over a communication channel.
- The communication channel may be noisy or limited bandwidth.
- Shannon's Coding Theorem says that

  "It is possible to device a mean whereby a communication system will transmit information with an arbitrarily small possibility of error provided that the information rate R is less than equal to C, Channel Capacity."

- The statements of Shannon's Theorem is:
  - Given a source of M equally likely messages, with M>>1, which is generating information at a rate R. Given a Channel Capacity C. Then, if R<<C, there exists a coding technique such that the output of the source may be transmitted over the channel with a probability of error of receiving the message which may be made arbitrarily small.
  - It indicates that for R << C. error free transmission is possible in the presence of noise.

PHYSICAL LAYER – CHANNEL ENCODING

Shannon Hartley Theorem

- According to the Shannon–Hartley theorem, the capacity of a channel to transmit a message without an error is given as:

$$C = B \cdot \log_2\left(1 + \frac{S}{N}\right)$$

where C is the channel capacity in bits per second;
B is the bandwidth of the channel in hertz;
S is the average signal power over the entire bandwidth, measured in watts; and
N is the average noise power over the entire bandwidth, measured in watts.

- This equation states that
  - For data transmission with free of errors, its transmission rate should be below the channel's capacity.
  - It also indicates how the signal-to-noise (SNR) ratio, can improve the channel's capacity.

PHYSICAL LAYER – CHANNEL ENCODING

Shannon Hartley Theorem

- The equation reveals two independent reasons why errors can be introduced during transmission:
    1. Information will be lost if the message is transmitted at a rate higher than the channel's capacity. This type of error is called ***equivocation*** in information theory. It is characterized as a subtractive error.

    2. Information will be lost because of noise, which adds **irrelevant** information into the signal.

- It is possible to describe the impact of **irrelevance** and **equivocation** as well as the percentage of information that can be transmitted over the channel without an error, which is also called **Transinformation** or **Mutual Information**.

- **Question is How?**
  Answer: Entropy, Joint Entropy & Conditional Entropy

Joint Entropy & Conditional Entropy

- In a single probability scheme, we may study the behavior of either transmitter or receiver by calculating the associated Entropy.

- To study of Communication System, we must simultaneously study the behavior of transmitter and receiver. This will rise the concept of two dimensional probability scheme.
- Let there be two finite discrete sample space S_1 and S_2 and we can assume their product as $S= S_1S_2$, Let

$$[X] = [x_1\ x_2\ x_3 \ldots\ldots x_m]$$
$$[Y] = [y_1\ y_2\ y_3 \ldots\ldots y_n]$$

- Be the sets of events in S_1 and S_2 respectively. Each event x_j of S_1 may occur in conjunction with any event y_k in S_2. Hence, the complete set of events in $S=S_1 S_2$ is

$$\lfloor XY \rfloor = \begin{bmatrix} x_1y_1 & \cdots & x_1y_n \\ \vdots & \ddots & \vdots \\ x_my_1 & \cdots & x_my_n \end{bmatrix}$$

Thus, we have three sets of complete probability schemes

$$P(X) = [P(x_j)]$$
$$P(Y) = [P(y_k)]$$
$$P(XY) = [P\ (x_j,\ y_k)]$$

PHYSICAL LAYER – CHANNEL ENCODING

Joint Entropy & Conditional Entropy

- We have three complete probability schemes and naturally there will be three associative entropies.

$$H(X) = -\sum_{j=1}^{M} p(xj)\log_2 p(xj)$$

$$P(x_j) = \sum_{k=1}^{n} p(xj, yk)$$

$$H(Y) = -\sum_{k=1}^{n} p(y_k)\log_2 p(y_k)$$

$$P(y_k) = \sum_{j=1}^{m} p(xj, yk)$$

$$H(XY) = -\sum_{j=1}^{m}\sum_{k=1}^{n} p(xj, yk)\log_2 p(xj, yk)$$

H(X) and H(Y) are marginal entropies of X & Y respectively, and H(XY) is the joint entropy of X & Y.

The conditional probability p(X/Y) is given by

$$P(X/Y) = \frac{P(X,Y)}{P(Y)}$$

We know that the y_K may occur in conjunction with $x_1, x_2, \ldots\ldots, x_m$.

Joint Entropy & Conditional Entropy

- The conditional entropy may be defined as

$$H(X/Y) = -\sum_{j=1}^{m} \sum_{k=1}^{n} p(xj, yk)\, log_2\, p(xj/yk)$$

- Similarly, it can be shown that

$$H(Y/X) = -\sum_{j=1}^{m} \sum_{k=1}^{n} p(xj, yk)\, log_2\, p(yk/xj)$$

H(X/Y) and H(Y/X) are average conditional entropies or simply conditional entropies.

Joint Entropy & Conditional Entropy

- There are five Entropies associated with a two-dimensional probability scheme. They are: H(X), H(Y), H(X,Y), H(X/Y), and H(Y/X). Where X represents a transmitter and Y a receiver. Therefore, the following interpretations of the different entropies for a communication system can be derived:

- H(X): Average information per character at the transmitter, or entropy of the transmitter. *It is the probabilistic nature of the transmitter.*

$$H(X) = -\sum_{j=1}^{M} p(xj)\log_2 p(xj)$$

- H(Y): Average information per character at the receiver, or entropy of the receiver. *It is the probabilistic nature of the receiver.*

$$H(Y) = -\sum_{k=1}^{n} p(yk)\log_2 p(yk)$$

- H(X, Y): Average information per pair of the transmitted and received characters or, *the average uncertainty of the communication system as a whole.*

$$H(XY) = -\sum_{j=1}^{m}\sum_{k=1}^{n} p(xj, yk)\log_2 p(xj, yk)$$

PHYSICAL LAYER – CHANNEL ENCODING

Joint Entropy & Conditional Entropy

- H(X/Y): A received character y_k may be the result of the transmission of one of the x_j's with a given probability. The entropy associated with this probability scheme, when y_k covers all received symbols. The conditional entropy is a measure of information about the transmitter, where it is known that Y is received.

- The content of information that can be lost because of the channel's inherent constraints can be quantified by observing the input *x* given that the output *y* is known:

$$H(X/Y) = -\sum_{j=1}^{m}\sum_{k=1}^{n} p(xj, yk)\, log_2\, p(xj/yk)$$

A good channel encoding scheme is one that has a high inference probability. This can be achieved by introducing redundancy during channel encoding.

H(X/Y) indicates how well one can recover the transmitted symbols, from the received symbols; i.e. It gives the information lost in the channel. This is also known as equivocation.

Joint Entropy & Conditional Entropy

- H(Y/X): A transmitted character x_j may be the result of the reception of one of the y_k's with a given probability. The entropy associated with this probability scheme, when x_j covers all transmitted symbols. The conditional entropy is a measure of information about the receiver, where it is known that X is transmitted.

  The content of information that can be introduced into the channel due to noise is described as the conditional information content, I*(Y/X)*. It is the information content of *y* that can be observed provided that *x* is known. The conditional entropy is given as:

$$H(Y/X) = -\sum_{j=1}^{m} \sum_{k=1}^{n} p(xj, yk)\, log_2\, p(yk/xj)$$

  A good channel encoder is one that reduces the irrelevance entropy.

  H(Y/X) indicates how well one can recover the received symbols, from the transmitted symbols; i.e. It can be considered as the noise entropy added in the channel. Thus, it is a measure of an error or noise due to channel.

PHYSICAL LAYER – CHANNEL ENCODING

Joint Entropy & Conditional Entropy

- The relationship between the different entropies is found as follows:

$$H(XY) = H(X/Y) + H(Y)$$

Similarly, it can be shown that

$$H(XY) = H(Y/X) + H(X)$$

Mutual Information

- The information content $I(X;Y)$ that overcomes the channel's constraints to reach the destination (the receiver) is called transinformation. The mutual information I(X; Y) indicates a measure of average information per symbol transmitted in the system. Given the input entropy, $H(X)$, and equivocation, $H(X|Y)$, the transinformation is computed as:

$$I(X;Y) = H(X) - H(X/Y) \quad \text{-------} \; 1$$

$$I(X;Y) = H(Y) - H(Y/X) \quad \text{-------} \; 2$$

$$I(X;Y) = H(X) + H(Y) - H(X,Y) \quad \text{------} \; 3$$

Since H(X) represents the uncertainty of the channel input (or source) before the channel output is observed, and

H(X/Y) is the uncertainty of the channel input (or Source) after the channel output is observed.

The mutual information I(X; Y) represents the uncertainty of the channel input (or Source) that is resolved by observing the channel output. *Since, I(X; Y) indicates a measure of the information transferred through the channel. It is also known as transferred information and Transinformation.*

PHYSICAL LAYER – CHANNEL ENCODING

Mutual Information

$$I(X;Y) = H(X) - H(X/Y) \quad \text{-------} \; 1$$

The transinformation is equal to the average source information minus the uncertainty that still remains about the message.

In other words, H(X/Y) is the additional information needed at the receiver after the reception in order to completely specify the message sent.

Thus, H(X/Y) gives the information lost in the channel. This is also known as equivocation.

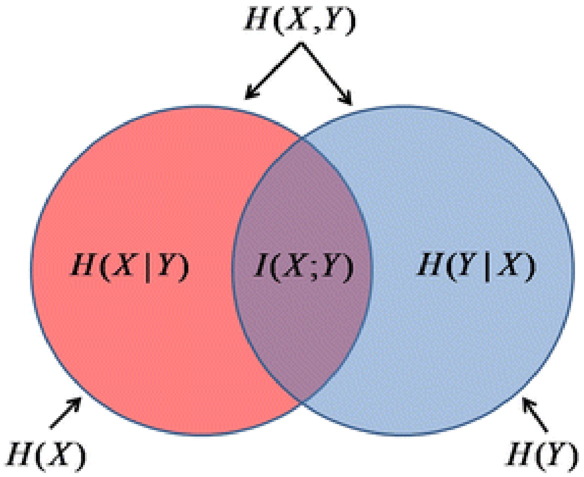
$$I(X;Y) = H(Y) - H(Y/X) \quad \text{-------} \; 2$$

The transinformation is equal to the receiver entropy minus that part of the receiver entropy which is not the information about the source.

Thus, H(Y/X) is considered as the noise entropy added in the channel. Thus, it is a measure of an error or noise due to channel.

PHYSICAL LAYER – CHANNEL ENCODING

Equivocation, Irrelevance & Transinformation



Channel Capacity

The mutual information I(X; Y) indicates a measure of average information per symbol transmitted in the system. A suitable measure for the efficiency of transmission of information may be introduced by comparing the actual rate and the upper bound of the rate of the information transmission of the channel.

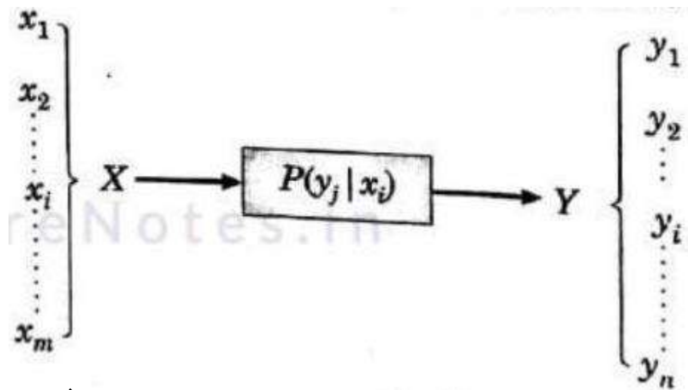
Shannon's has introduced the concept of the Channel Capacity:

Channel Capacity is defined as the maximum of mutual information. Thus the channel capacity C is given by

$$C = \max I(X;Y) = \max [\, H(X) - H(X/Y) \,]$$

Channel Representation

A communication channel may be defined as the path or medium through which the symbols flow to the receiver end. A discrete memory channel (DMC) is a statistical model with an input X and an output Y as shown in figure:



During each unit of time (signaling interval, the channel accepts an input symbol from X, and in response it generates an output symbol from Y. The channel is said to be "**discrete**" when the alphabets of X and Y are both finite.

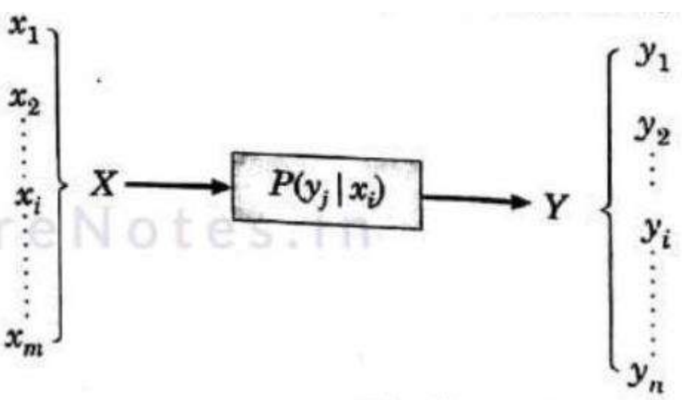
Also, it is said to be "**memoryless**" when the current output depends on only the current input and not on any of the previous inputs.

Channel Representation

A diagram of DMC with m inputs and n outputs has been given in the figure. The input X consists of input symbols x_1, x_2, ..., x_m.

The a priori probabilities of these source symbols $P(x_i)$ are assumed to be known. The output Y consists of output symbols y_1, y_2, ..., y_n.

Each possible input-output path is indicated along with probability $P(y_j/x_i)$, where $P(y_j/x_i)$ is called probability of obtaining output y_j given that input x_i and is called a channel transition probability.



Channel Matrix

A Channel is completely specified by the complete set of transition probabilities. Accordingly, the channel is often specified by the matrix of transition probabilities [P(Y/X)]. This matrix is given by

$$[P(Y|X)] = \begin{bmatrix} P(y_1|x_1) & P(y_2|x_1) & \cdots & P(y_n|x_1) \\ P(y_1|x_2) & P(y_2|x_2) & \cdots & P(y_n|x_2) \\ \cdots & \cdots & \cdots & \cdots \\ P(y_1|x_m) & P(y_2|x_m) & \cdots & P(y_n|x_m) \end{bmatrix}$$

This matrix is called a Channel Matrix. Since each input to the channel results some output, each row of the channel matrix must be sum to unity. This means that

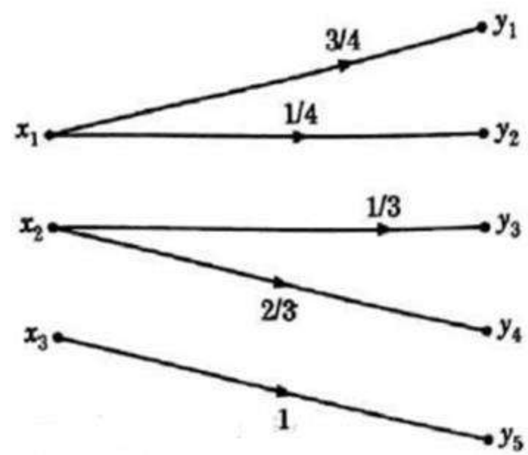
$$\sum_{j=1}^{n} P(y_j|x_i) = 1 \text{ for all } i$$

Lossless Channel

A Channel is described by a channel matrix with only one non-zero element in each column is called a lossless channel. An example of lossless channel has been shown in figure, and the corresponding channel matrix is given in the matrix below:

$$P[Y/X] = \begin{pmatrix} 3/4 & 1/4 & 0 & 0 & 0 \\ 0 & 0 & 1/3 & 2/3 & 0 \\ 0 & 0 & 0 & 0 & 1 \end{pmatrix}$$

It can be shown in the lossless channel, no source information is lost in the transmission.



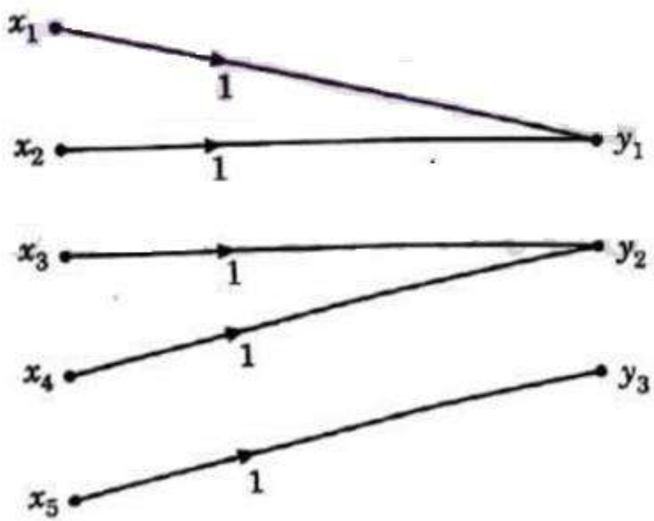
PHYSICAL LAYER – SOURCES ENCODING

Deterministic Channel

A Channel is described by a channel matrix with only one non-zero element in each row is called a deterministic channel. An example of deterministic channel has been shown and the corresponding channel matrix is given below:

It may be noted that since each row has only one non-zero element, therefore, this element must be unity by equation given below. Thus, when a given source symbols is sent in the deterministic channel, it is clear which output symbol will be received.

$$\sum_{j=1}^{n} P(y_j | x_i) = 1 \text{ for all } i$$

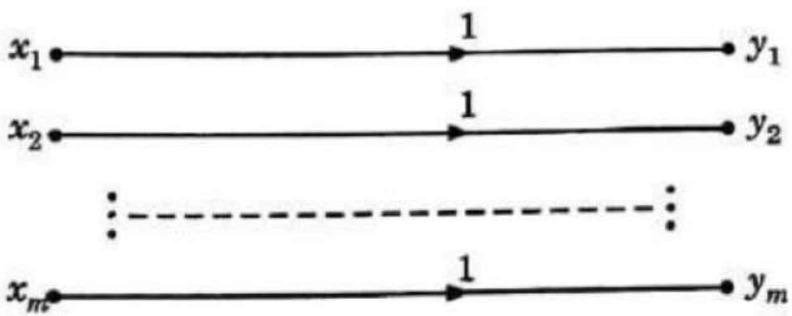


$$[P(Y|X)] = \begin{bmatrix} 1 & 0 & 0 \\ 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{bmatrix}$$

PHYSICAL LAYER – SOURCES ENCODING

Noiseless Channel

A Channel is called noiseless if it is both lossless and deterministic. A noiseless channel has been shown in the figure. The channel matrix has only one element in each row and each column, and this element is unity. Note that the input and output symbols are of the same size, i.e., m=n for the noiseless channel.



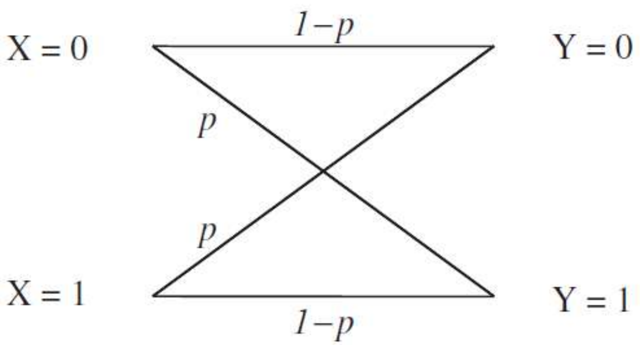
$$\begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

PHYSICAL LAYER – SOURCES ENCODING

Binary Symmetry Channel (BSC)

A Binary Symmetry Channel is defined by the channel diagram and its channel matrix given below:

$$P_{\mathrm{BSC}} = \begin{bmatrix} (1-p) & p \\ p & (1-p) \end{bmatrix}$$



A BSC channel has two inputs (x1=0, x2=1) and two outputs (y1=0, y2=1). This channel is symmetry because the probability of receiving a 1 if a 0 is sent is the same as the probability of receiving a 0 if a 1 is sent,

NODE AND NETWORK MANAGEMENT

LOCALIZATION

OUTLINES

- Ranging Techniques
- Types of Localization Techniques
- Range-based Localization
- Range-free Localization
- Event-driven Localization

OVERVIEW

- Without knowledge of location of a sensor, the information produced by such sensor is of limited use
  - location of sensed events in the physical world
  - location-aware services
  - location often primary sensor information (supply chain management, surveillance)
  - object tracking
  - protocols based on geographic information (routing)
  - coverage area management
- Location often not known a priori, therefore, localization is the task of determining the position (e.g., coordinates) of a sensor or the spatial relationships among objects

OVERVIEW

- Global position
  - ❑ position within general global reference frame
  - ❑ Global Positioning System or GPS (longitudes, latitudes)
  - ❑ Universal Transverse Mercator or UTM (zones and latitude bands)

- Relative position
  - ❑ based on arbitrary coordinate systems and reference frames
  - ❑ distances between sensors (no relationship to global coordinates)

- Accuracy versus precision
  - ❑ GPS: true within 10m for 90% of all measurements
  - ❑ accuracy: 10m (“how close is the reading to the ground truth?”)
  - ❑ precision: 90% (“how consistent are the readings?”)

RANGING TECHNIQUES

- Time of Arrival (ToA, time of flight)
  - distance between sender and receiver of a signal can be determined using the measured signal propagation time and known signal velocity
  - sound waves: 343m/s, i.e., approx. 30ms to travel 10m
  - radio signals: 300km/s, i.e., approx. 30ns to travel 10m

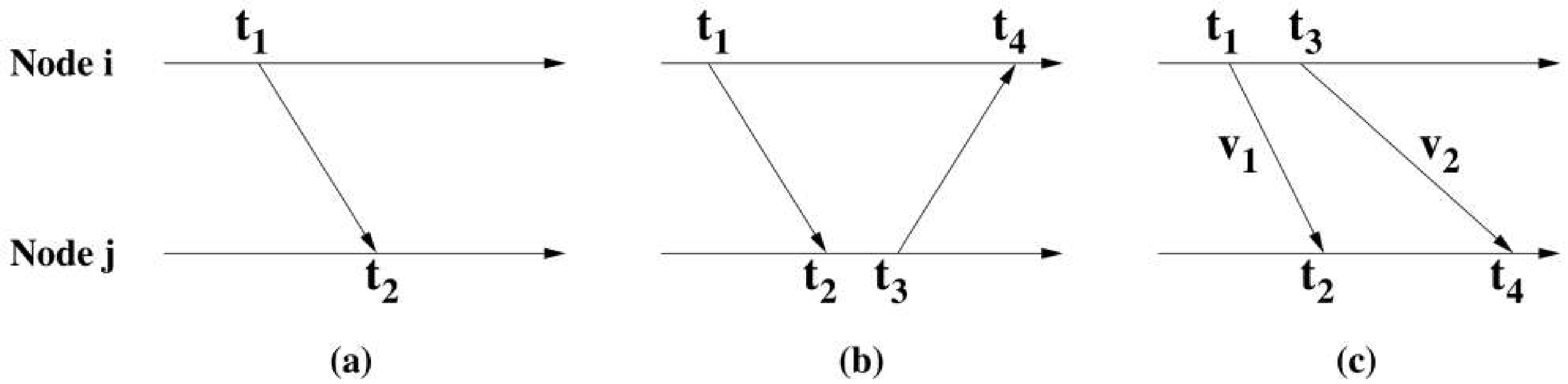
- One-way ToA
  - one-way propagation of signal
  - requires highly accurate synchronization of sender and receiver clocks

$$dist_{ij} = (t_2 - t_1) * v$$

- Two-way ToA
  - round-trip time of signal is measured at sender device
  - third message if receiver wants to know the distance

$$dist_{ij} = \frac{(t_4 - t_1) - (t_3 - t_2)}{2} * v$$

RANGING TECHNIQUES



- Time Difference of Arrival (TDoA)
  - two signals with different velocities
  - example: radio signal (sent at t1 and received at t2), followed by acoustic signal (sent at t3=t1+twait and received at t4)

$$dist = (v_1 - v_2) * (t_4 - t_2 - t_{wait})$$

  - no clock synchronization required
  - distance measurements can be very accurate need for additional hardware

RANGING TECHNIQUES

- Angle of Arrival (AoA)
  - direction of signal propagation
  - typically achieved using an array of antennas or microphones
  - angle between signal and some reference is orientation
  - spatial separation of antennas or microphones leads to differences in arrival times, amplitudes, and phases
  - accuracy can be high (within a few degrees)
  - adds significant hardware cost

RANGING TECHNIQUES

- Received Signal Strength (RSS)
  - ❑ signal decays with distance
  - ❑ many devices measure signal strength with received signal strength indicator (RSSI)
    - ➢ vendor-specific interpretation and representation
    - ➢ typical RSSI values are in range of 0..RSSI_Max
    - ➢ common values for RSSI_Max: 100, 128, 256
  - ❑ in free space, RSS degrades with square of distance
  - ❑ expressed by Friis transmission equation

$$\frac{P_r}{P_t} = G_t G_r \frac{\lambda^2}{(4\pi)^2 R^2}$$

  - ❑ In practice, the actual attenuation depends on multipath propagation effects, reflections, noise, etc. □ realistic models replace R2 with Rn (n=3..5)

TYPES OF LOCALIZATION TECHNIQUES

- Localization schemes are classified as anchor based or anchor free, centralized or distributed, GPS based or GPS free, fine grained or coarse grained, stationary or mobile sensor nodes, and range based or range free. We will briefly discuss all of these methods.

- Anchor Based and Anchor Free

  In anchor-based mechanisms, the positions of few nodes are known. Unlocalized nodes are localized by these known nodes positions. Accuracy is highly depending on the number of anchor nodes.

  Anchor-free algorithms estimate relative positions of nodes instead of computing absolute node positions.

TYPES OF LOCALIZATION TECHNIQUES

- Centralized and Distributed

  In centralized schemes, all information is passed to one central point or node which is usually called “sink node or base station”. Sink node computes position of nodes and forwards information to respected nodes. Computation cost of centralized based algorithm is decreased, and it takes less energy as compared with computation at individual node.

  In distributed schemes, sensors calculate and estimate their positions individually and directly communicate with anchor nodes. There is no clustering in distributed schemes, and every node estimates its own position

TYPES OF LOCALIZATION TECHNIQUES

- GPS Based and GPS Free

  GPS-based schemes are very costly because GPS receiver has to be put on every node. Localization accuracy is very high as well.

  GPS-free algorithms do not use GPS, and they calculate the distance between the nodes relative to local network and are less costly as compared with GPS-based schemes. Some nodes need to be localized through GPS which are called anchor or beacon nodes that initiate the localization process

- Coarse Grained and Fine Grained

  Fine-grained localization schemes result when localization methods use features of received signal strength, while coarse-grained localization schemes result without using received signal strength.

TYPES OF LOCALIZATION TECHNIQUES

- Stationary and Mobile Sensor Nodes

  Localization algorithms are also designed according to field of sensor nodes in which they are deployed. Some nodes are static in nature and are fixed at one place, and the majority applications use static nodes. That is why many localization algorithms are designed for static nodes. Few applications use mobile sensor nodes, for which few mechanisms are designed

- Range-Free and Range-Based Localization

  Range-free methods are distance vector (DV) hop, hop terrain, centroid system, APIT, and gradient algorithm. Range-free methods use radio connectivity to communicate between nodes to infer their location. In range-free schemes, distance measurement, angle of arrival, and special hardware are not used

  Range-based schemes are distance-estimation- and angle-estimation-based techniques. Important techniques used in range-based localization are received signal strength indication (RSSI), angle of arrival (AOA), time difference of arrival (TDOA), and time of arrival (TOA)

TRIANGULATION

- Triangulation is the process of determining the location of a point by measuring only angles to it from known points at either end of a fixed baseline by using trigonometry, rather than measuring distances to the point directly as in trilateration.

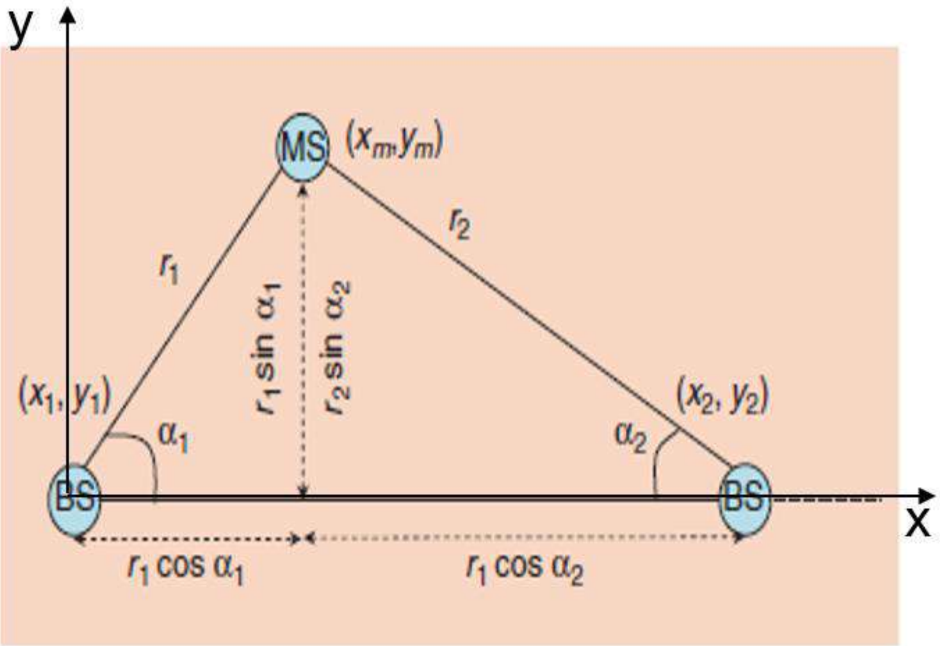
- The point can then be fixed as the third point of a triangle with one known side and two known angles.

- In this approach, information about angles (using AoA) is used instead of distances. Position computation can be done remotely or by the node itself (auto-localization); the latter is more common in WSN.

- In this last case, depicted in (b) at least three reference nodes are required. The unknown node estimates its angle to each of the three reference nodes and, based on these angles and the positions of the reference nodes (which form a triangle), computes its own position using simple trigonometrically relationships.

TRIANGULATION

- Example of range-based localization
- Uses the geometric properties of triangles to estimate location
- Relies on angle (bearing) measurements
- Minimum of two bearing lines (and the locations of anchor nodes or the distance between them) are needed for two-dimensional space
- AoA to compute the angles.
- The number of BSs needed for the location process is less.
- Compute the linear least-square solution.
- Assuming $(x_1,y_1)=(0,0)$ and the x axis defined by the two beacon nodes we have:



$$\begin{bmatrix} x_m \\ y_m \end{bmatrix} = \begin{bmatrix} r_1 \cos \alpha_1 \\ r_1 \sin \alpha_1 \end{bmatrix}$$

and

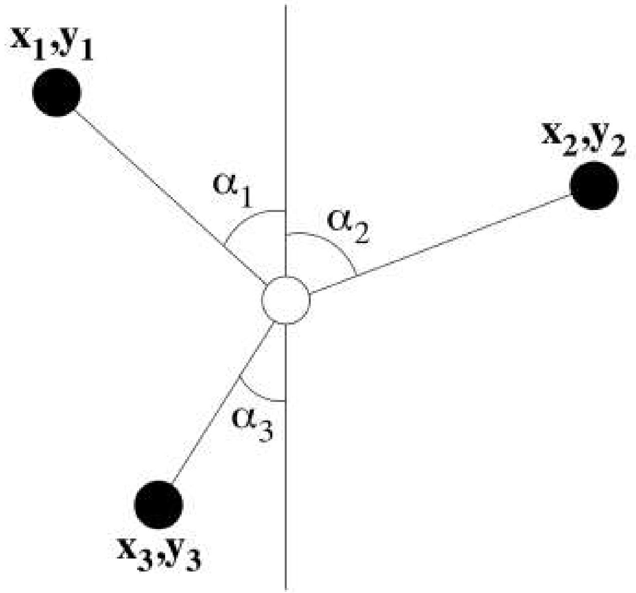
$$\begin{bmatrix} x_m \\ y_m \end{bmatrix} = \begin{bmatrix} x_2 \\ 0 \end{bmatrix} + \begin{bmatrix} r_2 \cos \alpha_2 \\ r_2 \sin \alpha_2 \end{bmatrix}$$

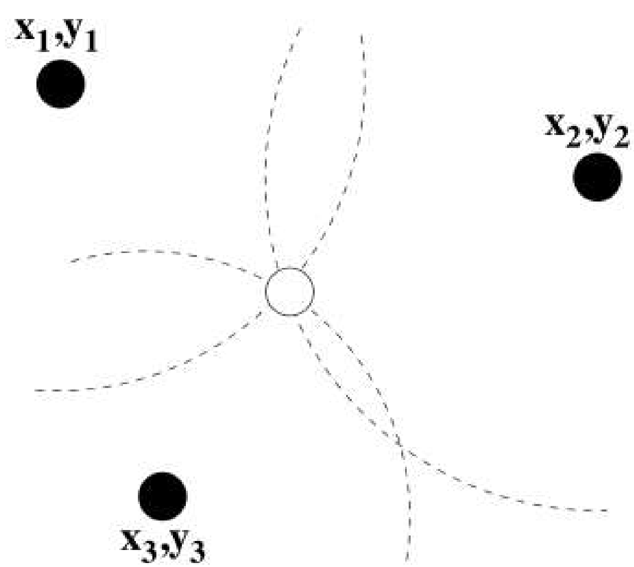
TRIALTERATION

- Localization based on measured distances between a node and a number of anchor points with known locations.

- Basic concept: given the distance to an anchor, it is known that the node must be along the circumference of a circle centered at anchor and a radius equal to the node-anchor distance.

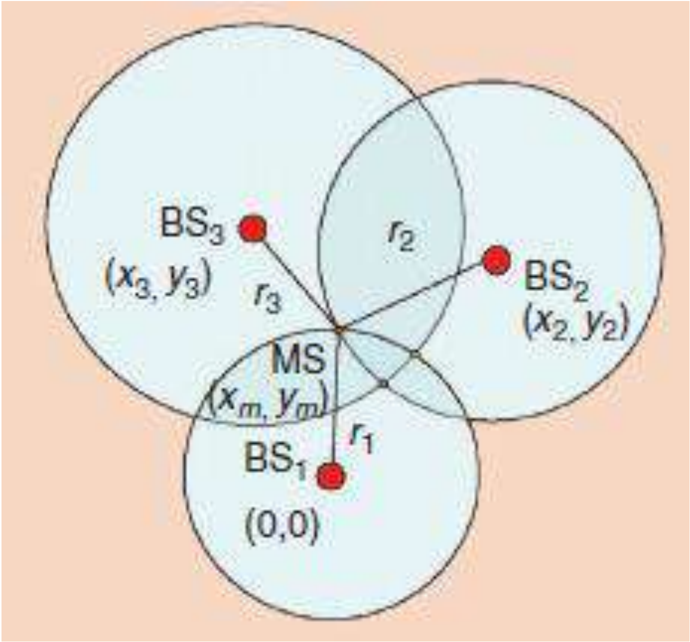
- In two-dimensional space, at least three non-collinear anchors are needed and in three-dimensional space, at least four non-coplanar anchors are needed





TRIALTERATION

- Distance measured: RSSI, ToA, TDoA
- Requires at least 3 BNs in 2D, and 4 BNs in 3D..
- Compute the linear least-squares solution
- Multilateration if more than three beacons are used to estimate the sensor's position



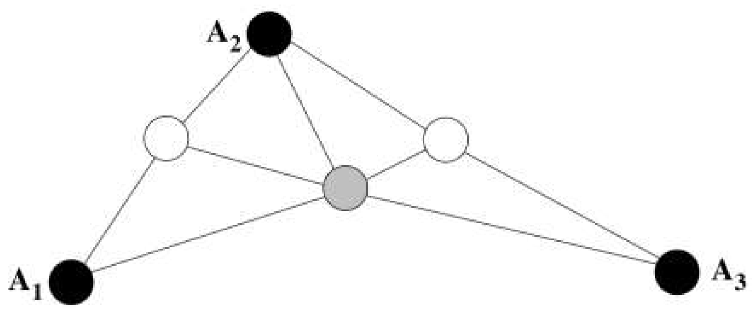
$$r_1^2 = x_m^2 + y_m^2$$
$$r_2^2 = (x_2 - x_m)^2 + (y_2 - y_m)^2$$
$$r_3^2 = (x_3 - x_m)^2 + (y_3 - y_m)^2 .$$

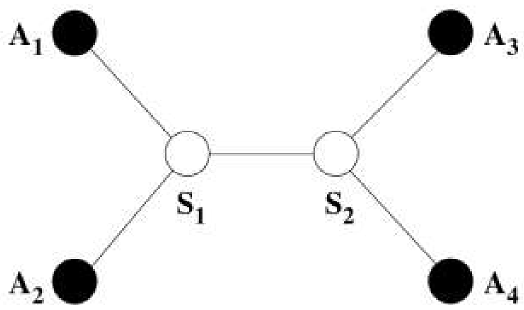
ITERATIVE/COLLABORATIVE MULTILATERATION

- Problem: what if node does not have at least three neighboring anchors?

- Solution: once a node has determined its position, it becomes an anchor Iterative Multilateration:
  - ❑ repeats until all nodes have been localized
  - ❑ error accumulates with each iteration

- Collaborative Multilateration:
  - ❑ goal: construct a graph of participating nodes, i.e., nodes that are anchors or have at least three participating neighbors
  - ❑ node then tries to estimate its position by solving the corresponding system of over constrained quadratic equations relating the distances among the node and its neighbors





GPS BASED LOCALIZATION

- Global Positioning System
  - most widely publicized location-sensing system
  - provides lateration framework for determining geographic positions
  - originally established as NAVSTAR (Navigation Satellite Timing and Ranging)
  - only fully operational global navigation satellite system (GNSS)
  - consists of at least 24 satellites orbiting at approx. 11,000 miles
  - started in 1973, fully operational in 1995

- Two levels of service:
  - Standard Positioning Service (SPS)
    - available to all users, no restrictions or direct charge
    - high-quality receivers have accuracies of 3m and better horizontally

  - Precise Positioning Service (PPS)
    - used by US and Allied military users
    - uses two signals to reduce transmission errors

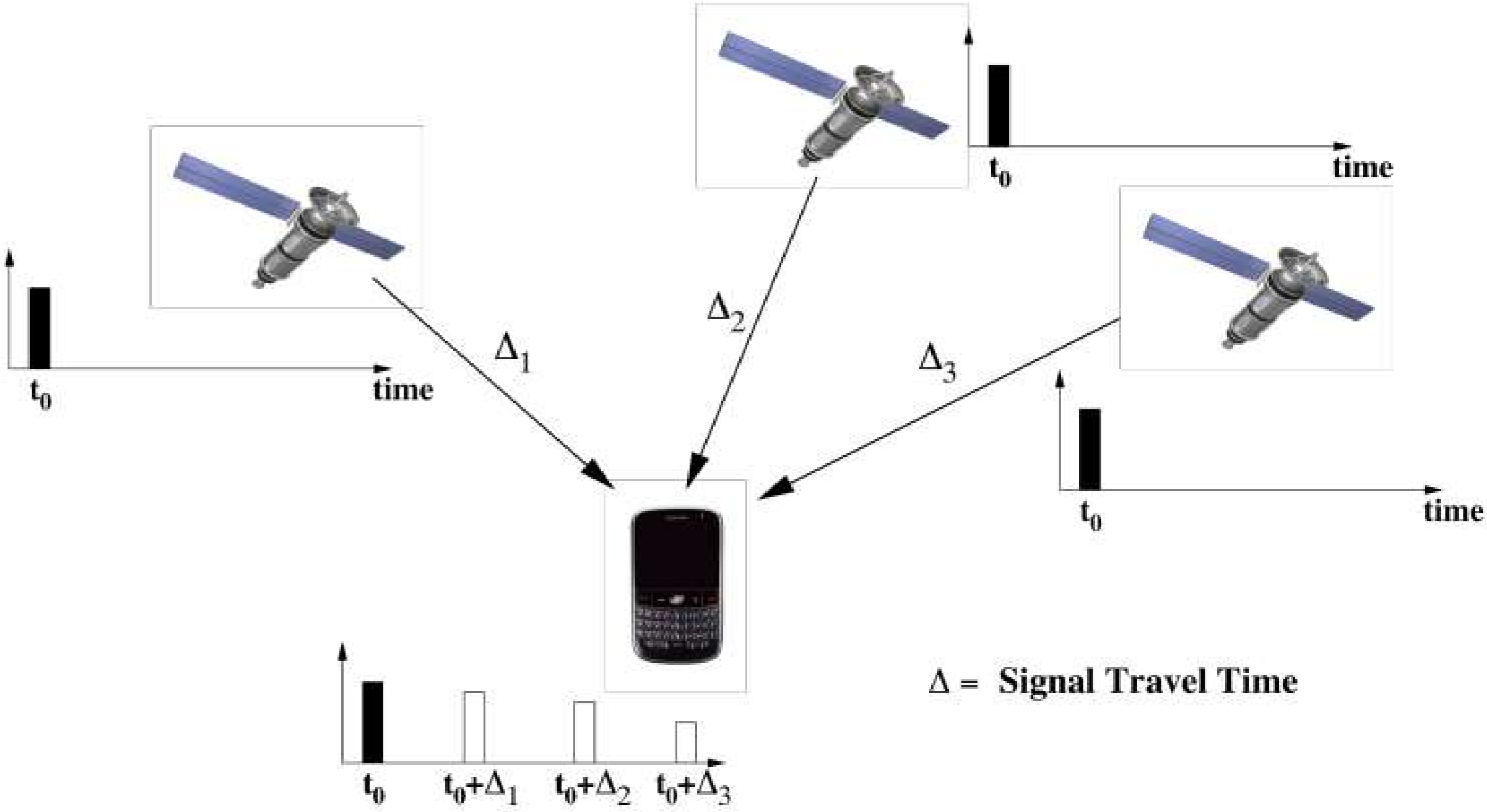
GPS BASED LOCALIZATION

- Satellites are uniformly distributed in six orbits (4 satellites per orbit)
- Satellites circle earth twice a day at approx. 7000 miles/hour
- At least 8 satellites can be seen simultaneously from almost anywhere
- Each satellite broadcasts coded radio waves (pseudorandom code), containing
  - ❑ identity of satellite
  - ❑ location of satellite
  - ❑ the satellite's status
  - ❑ data and time when signal was sent
- Six monitor stations constantly receive satellite data and forward data to a master control station (MCS)
- MCS is located near Colorado Springs, Colorado
- MCS uses the data from monitor stations to compute corrections to the satellites' orbital and clock information which are sent back to the satellites

GPS BASED LOCALIZATION

- Satellites and receivers use accurate and synchronized clocks

- Receiver compares generated code with received code to determine
  - ❑ the actual code generation time of the satellite
  - ❑ time difference Δ between code generation time and current time
  - ❑ Δ expresses the travel time of the code from satellite to receiver



GPS BASED LOCALIZATION

- Radio waves travel at the speed of light (approx. 186,000 miles/second)
- With known Δ, the distance can be determined
- Receiver knows that it is located somewhere on a sphere centered on the satellite with a radius equal to this distance
- With three satellites, the location can be narrowed down to two points
  - typically one of these two points can be eliminated easily

GPS BASED LOCALIZATION

- With four satellites, accurate localization is possible
  - ❑ accurate positioning relies on accurate timing
  - ❑ receiver clocks are much less accurate than atomic GPS clocks
  - ❑ small timing errors lead to large position errors
    - ➢ example: clock error of 1ms translates to a position error of 300km
  - ❑ fourth sphere would ideally intersect with all three other spheres in one exact location
  - ❑ spheres too large: reduce them by adjusting the clock (moving it forward)
  - ❑ spheres too small: increase them by adjusting the clock (moving it backward

GPS BASED LOCALIZATION

- Most GPS receivers today can achieve good accuracy (e.g., 10m or less)

- Additional advanced techniques can be used to further improve accuracy:
  - example: Differential GPS (DGPS)
    - relies on land-based receivers with exactly known locations
    - they receive signals, compute correction factors, and broadcast them to GPS receivers
    - GPS receivers correct their own measurements

- GPS in wireless sensor networks
  - prohibitive factors: power consumption, cost, size, need for LOS
  - deployment can be limited to a few (more powerful) nodes
    - used as anchor nodes and reference points for range-free localization techniques

AD HOC POSITIONING SYSTEM (APS)

- Example of a range-free localization approach
  - based on connectivity information instead of distance/angle measurements
  - no additional hardware required (cost-effective)

- APS is a distributed connectivity-based localization algorithm
  - estimates node locations with the support of at least three anchor nodes
    - localization errors can be reduced by increasing the number of anchors
  - uses concept of DV (distance vector), where nodes exchange routing tables with their one-hop neighbors

AD HOC POSITIONING SYSTEM (APS)

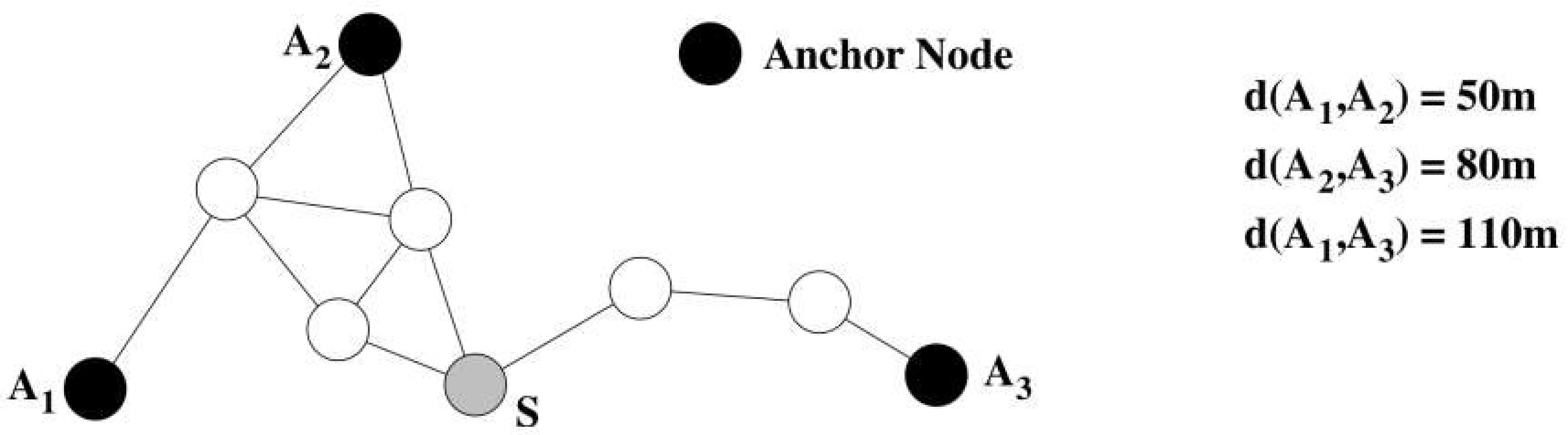
- Most basic scheme of APS: DV-hop
  - each node maintains a table {Xi , Yi , hi } (location of node i and distance in hops between this node and node i)
  - when an anchor obtains distances to other anchors, it determines the average hop length ("correction factor" ci), which is then propagated throughout the network

$$c_i = \frac{\sum \sqrt{(X_i - X_j)^2 + (Y_i - Y_j)^2}}{\sum h_i}$$

  - given the correction factor and the anchor locations, a node can perform trilateration

AD HOC POSITIONING SYSTEM (APS)

- Example with three anchors
  - A1 knows its distance to A2 (50m) and A3 (110m)
  - A1 knows its hop distance to A2 (2) and A3 (6)
  - correction factor: (50+110)/(2+6) = 20 (estimated distance of a hop)
  - corrections are propagated using controlled flooding, i.e., a node only uses one correction factor and ignores subsequently received ones



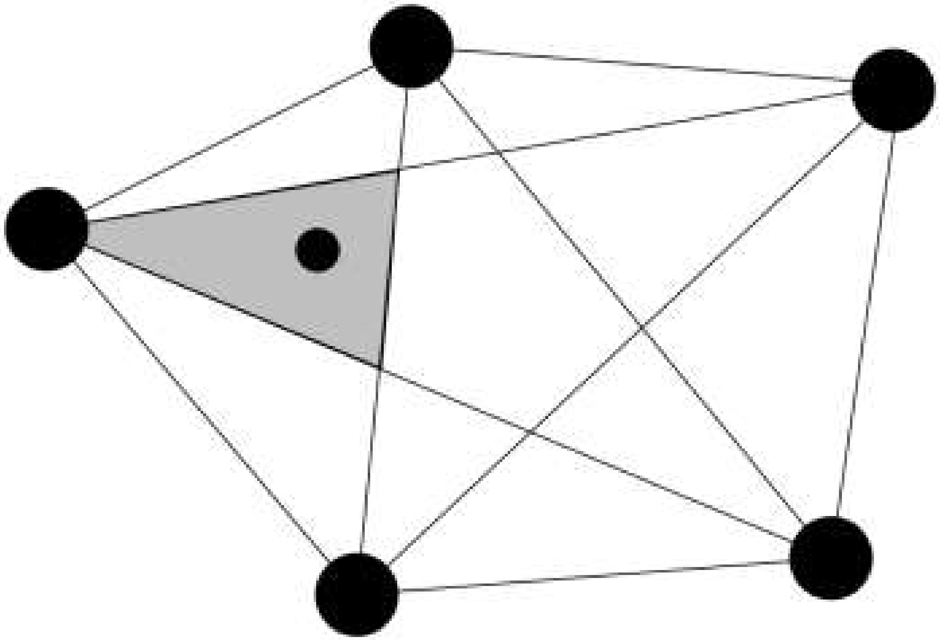
AD HOC POSITIONING SYSTEM (APS)

- Variation of this approach: DV-distance method
  - distances are determined using radio signal strength measurements
  - distances are propagated to other nodes
  - provides finer granularity (not all hops are estimated to be the same size)
  - more sensitive to measurement errors

- Another variation: Euclidean method
  - true Euclidian distances to anchors are used
  - node must have at least two neighbors that have distance measurements to anchors and the distance between the two neighbors is known
  - simple trigonometric relationships are used to determine the distance between node and anchor

APPROXIMATE POINT IN TRIANGULATION (APIT)

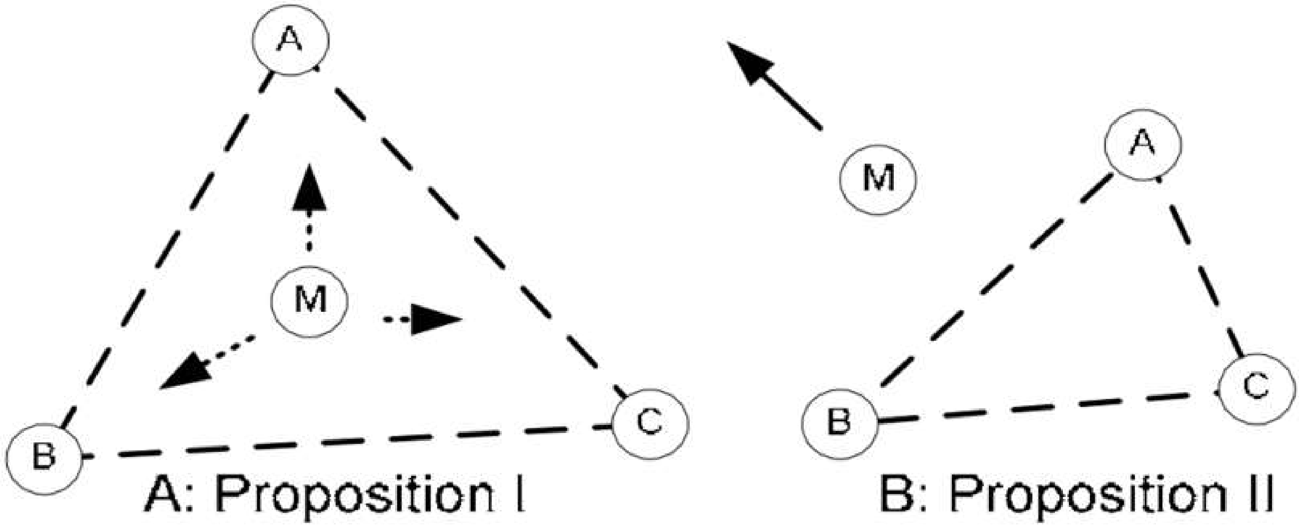
- Example of an area-based range-free localization scheme
- APIT relies on anchor nodes
  - any combination of three anchors forms a triangle
  - a node determines its presence inside or outside a triangle using the Point in Triangulation (PIT) test



APPROXIMATE POINT IN TRIANGULATION (APIT)

Perfect PIT Test

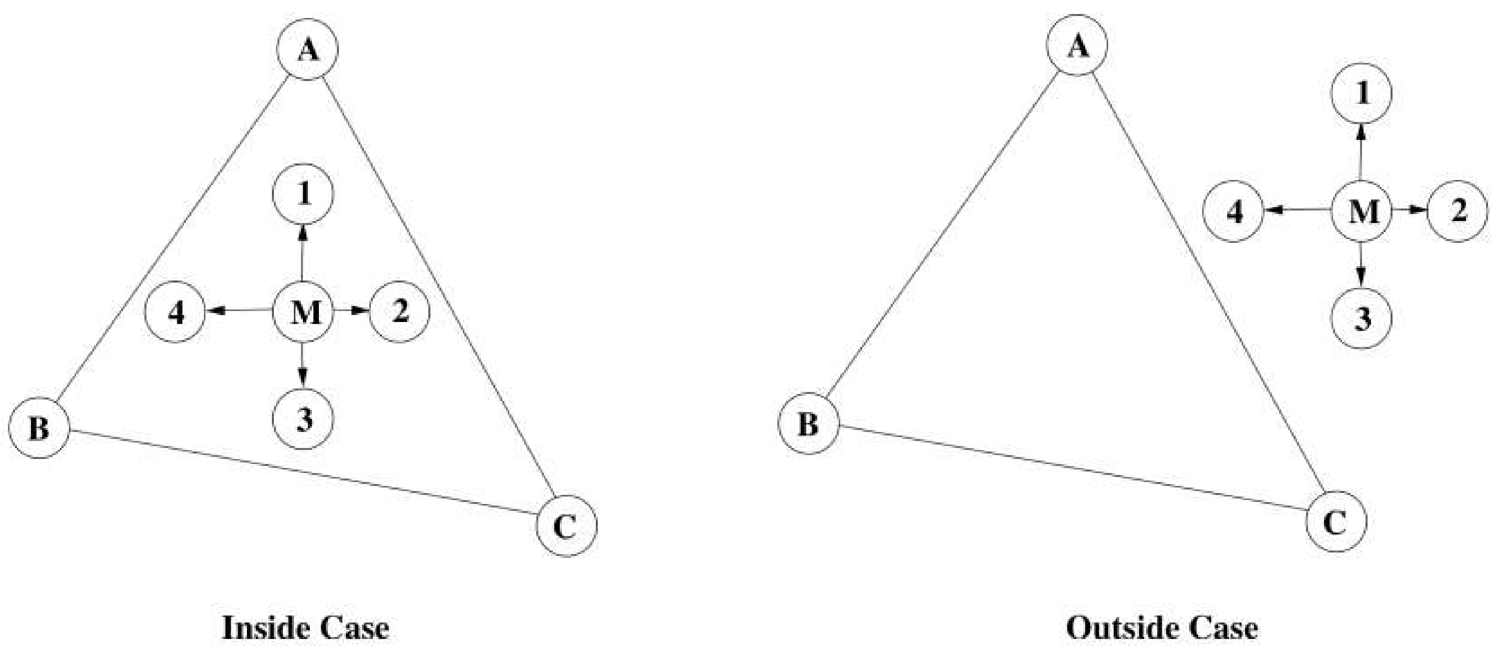
- Proposition I: Node M in triangle if: M shifted in any direction New position is nearer/ further from at least one anchor
- Proposition II: Node M outside if: M can be shifted in a direction New position is nearer/ further to all three anchors



- This perfect PIT test is infeasible in practice since it would require that a node can be moved in any direction

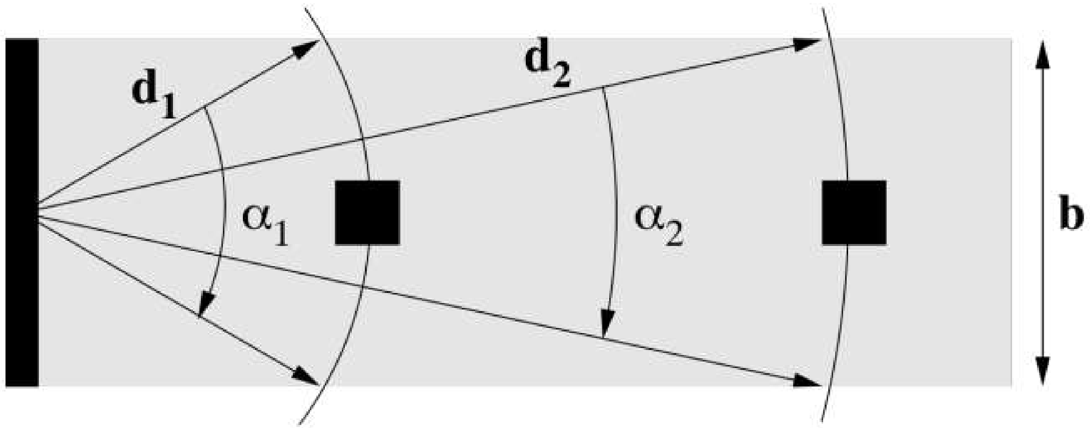
APPROXIMATE POINT IN TRIANGULATION (APIT)

- In dense networks, node movement can be emulated using neighbor information (exchanged via beacons)
  - signal strength measurements can be used to determine if a node is closer to an anchor or further away
  - if no neighbor of node M is further from or closer to three anchors A, B, C simultaneously, M can assume that it is inside the triangle



LIGHTHOUSE APPROACH

- Example of an event-driven localization approach
- Requirement: base station with light emitter
- Idealistic light source: emitted beam of light is parallel (constant width b)
- Light source rotates s. t. sensor sees beam of light for t_{beam}



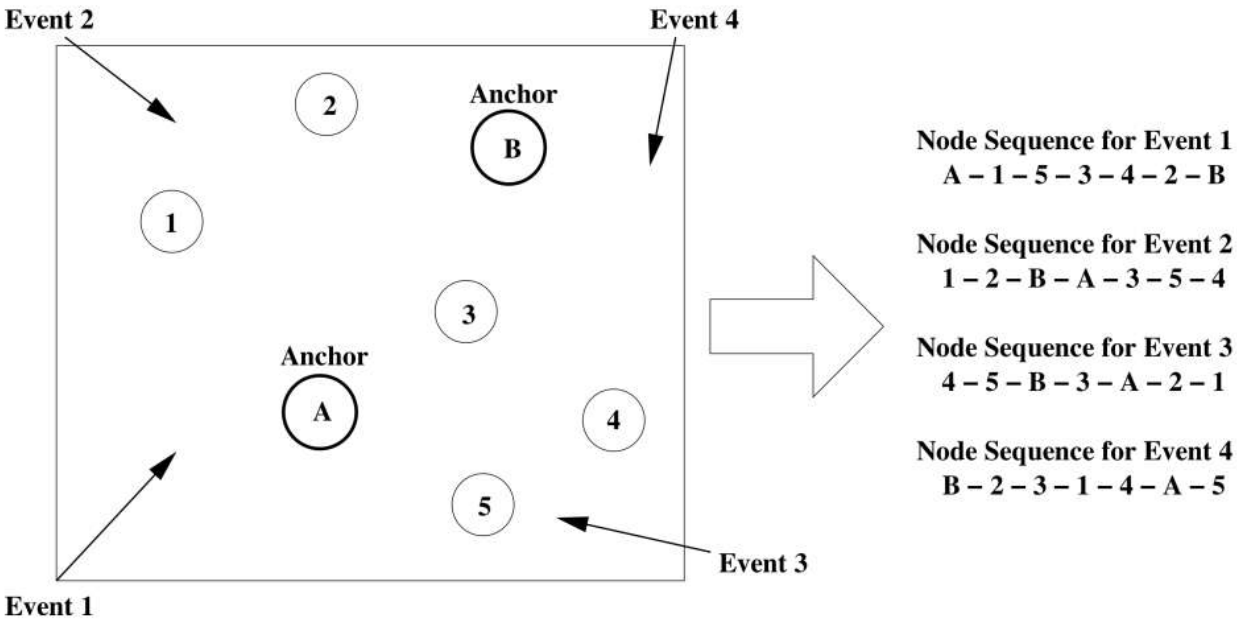
$$d = \frac{b}{2\sin(\alpha/2)} \qquad \alpha = 2\pi \frac{t_{beam}}{t_{turn}}$$

LIGHTHOUSE APPROACH

- Perfectly parallel light beams are hard to realize in practice
- Small beam spreads can result in large localization errors
  - if b=10cm and spread=1°, b'=18.7cm at 5m distance
- Beam width should be large to keep inaccuracies small
- Solution: two laser beams that outline a "virtual" parallel beam
  - only edges of the virtual beam are of interest

MULTISEQUENCE POSITIONING

- MSP works by extracting relative location information from multiple simple one dimensional orderings of sensor nodes
  - event generators at different locations trigger events (e.g., ultrasound signals or laser scans)
  - nodes observe events at different times, leading to node sequence
  - multisequence processing algorithm narrows potential locations for each node
  - distribution-based estimation method can estimate exact locations



POWER MANAGEMENT

OUTLINES

- Local Power Management Aspects
  - Processor Subsystem
  - Communication Subsystem
  - Bus Frequency and RAM Timing
  - Active Memory
  - Power Subsystem
  - Battery
  - DC – DC Converter

- Dynamic Power Management
  - Dynamic Operation Modes
  - Transition Costs
  - Dynamic Scaling
  - Task Scheduling

- Conceptual Architecture
  - Architectural Overview

POWER MANAGEMENT

The power consumption of a wireless sensor network (WSN) is of crucial concern because of the scarcity of energy. Whereas energy is a scarce resource in every wireless device, the problem in WSNs is amplified for the following reasons:

1. The ***nodes are very small*** *in size* to accommodate high-capacity power supplies compared to the complexity of the task they carry out.
2. It is ***impossible to manually*** *change*, replace, or recharge batteries - WSNs consist of a large number of nodes.
3. *The* ***size of nodes*** is still a ***constraining*** factor for renewable energy and ***self-recharging mechanisms***.
4. *The* ***failure of a few nodes*** may cause the entire network to fragment prematurely.

POWER MANAGEMENT

- The problem of power consumption can be approached from *two angles*:
  - Develop energy-efficient communication protocols
    - self-organization, medium access, and routing protocols.
  - Identify activities in the networks that are both wasteful and unnecessary then mitigate their impact.

- Most inefficient activities are results of *non-optimal configurations* in hardware and software components:
  - e.g., a considerable amount of energy is wasted by an idle processing or a communication subsystem.
  - A radio that aimlessly senses the media or overhears while neighboring nodes communicate with each other consumes a significant amount of power.

POWER MANAGEMENT

Wasteful and unnecessary activities can be described as *local* or *global*":

- *e.g.,* some nodes exhausted their batteries prematurely because of *unexpected overhearing* of traffic that caused the communication subsystem to become operational for a longer time than originally intended.

- some nodes exhausted their batteries prematurely because they aimlessly *attempted to establish links with a network* that had become no longer accessible to them

POWER MANAGEMENT

A *dynamic power management (DPM)* strategy ensures that power is consumed economically

- The strategy can have a *local* or *global* scope, or *both*
- *A local DPM strategy* aims to
  - Minimize the power consumption of *individual nodes*
    - by providing each subsystem with the amount of power that is sufficient to carry out a task at hand
    - when there is no task to be processed, the DPM strategy forces some of the subsystems to operate at the *most economical power mode* or puts them into a *sleeping mode*“

- *A global DPM strategy* attempts to
  - Minimize the power consumption of *the overall network* by defining a *network-wide sleeping state*

POWER MANAGEMENT

- *Synchronous sleeping schedule*
  - Let individual nodes ***define*** their own sleeping schedules
  - ***Share*** these schedules with their neighbors to enable a coordinated sensing and an efficient inter-node communication
  - The ***problem*** is that neighbors need to synchronize time as well as schedules and the process is energy intensive

- *Asynchronous sleeping schedule*
  - Let individual **nodes *keep*** their sleeping schedules to themselves
  - A node that initiates a communication should ***send a preamble*** until it receives an acknowledgment from its receiving partner
  - Avoids the needs to synchronize schedules
  - It can have *a* ***latency side-effect*** on data transmission

POWER MANAGEMENT

- In both approaches, individual nodes ***wake up periodically***:
  - to determine whether there is a node that wishes to communicate with them
  - to process tasks waiting in a queue

- Fundamental premises about Embedded systems:
  - predominantly event-driven
  - experience non-uniform workload during operation time
- DPM refers to ***selectively shutting-off*** and/or ***slowing-down*** system components that are idle or underutilised
- A policy determines the type and timing of power transitions based on ***system history***, ***workload*** and ***performance constraints***

LOCAL POWER MANAGEMENT ASPECTS

The first step is the *understanding of how power is consumed* by the different subsystems of a wireless sensor node:

- Wasteful activities to be avoided and to frugally budget power
- One to estimate the overall power dissipation rate in a node and how this rate affects the lifetime of the entire network
- In the following subsections, a mode detail observation into the different subsystems of a node is made:
  - Processor Subsystem
  - Communication Subsystem
  - Bus Frequency and RAM Timing
  - Active Memory
  - Power Subsystem
    - Battery
    - DC – DC Converter

POWER SUBSYSTEM

- Most existing processing subsystems employ microcontrollers, notably
  - Intel's StrongARM and Atmel's AVR
- These microcontrollers can be configured to operate at ***various power modes*"*.**
  - *e.g.,* the ATmega128L microcontroller has ***six different power modes***:
    - Idle,
    - ADC noise reduction,
    - Power save,
    - Power down,
    - Standby, and
    - Extended standby.

POWER SUBSYSTEM

Sleep Mode	Active clock domains					Oscillators		Wake up sources					
	clk_{CPU}	clk_{FLASH}	clk_{IO}	clk_{ADC}	clk_{ASY}	Main Clock Source Enabled	Timer Osc Enabled	INT7	TWI Addr. Match	Timer	EEPROM Ready	ADC	Other I/O
Idle			X	X	X	X	X	X	X	X	X	X	X
ADC noise red.				X	X	X	X	X	X	X	X	X	
power down								X	X				
Power save					x		x	x	x	x			
standby						x		x	x				
Ext. standby						x	x	x	x	x			

- The *idle* mode stops the CPU
    - While allowing the SRAM, Timer/Counters, SPI port and interrupt system to continue functioning.
- The *power down* mode saves the registers' content
    - While freezing the oscillator and disabling all other chip functions until the next interrupt or Hardware Reset.

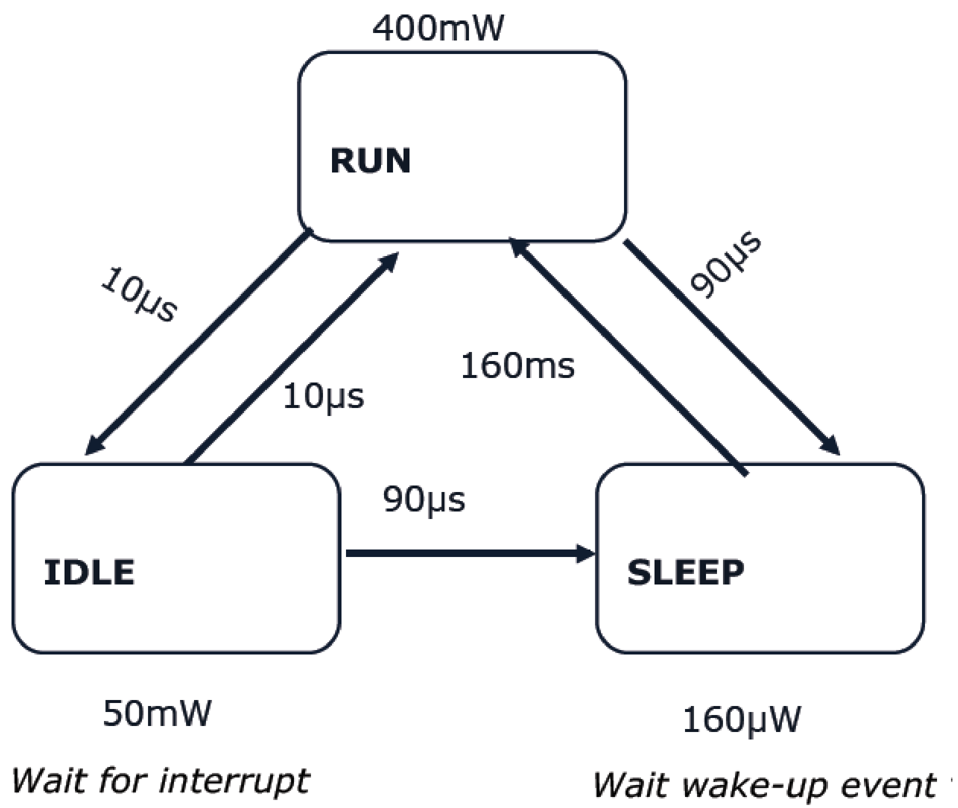
POWER SUBSYSTEM

- In the *power-save* mode, the asynchronous timer continues to run
  - Allowing the user to maintain a timer base while the remaining components of the device enter into a sleep mode
- The *ADC noise reduction* mode stops the CPU and all I/O modules
  - *Except* the asynchronous *timer* and the *ADC*
  - The aim is to minimize switching noise during ADC conversions
- In *standby* mode, a crystal/resonator oscillator runs while the remaining hardware components enter into a sleep mode
  - This allows very fast start-up combined with low power consumption
- In *extended standby* mode, both the main oscillator and the asynchronous timer continue to operate

POWER SUBSYSTEM

- Additional to the above configurations, the processing subsystem can *operate* with *different supply voltages* and *clock frequencies.*
- *Transiting* from one power mode to another also has its own power and latency cost.

Power state machine for the StrongARM-1100 processor



COMMUNICATION SUBSYSTEM

- The *power consumption* of the communication subsystem can *be influenced by several aspects*:
  - the modulation type and index
  - the transmitter's power amplifier and antenna efficiency
  - the transmission range and rate
  - the sensitivity of the receiver
- These aspects can be *dynamically reconfigured*

COMMUNICATION SUBSYSTEM

Determining the most efficient active state operational mode is *not a simple decision*

- *e.g.,* the power consumption of a transmitter may *not* necessarily be reduced by simply reducing the transmission rate or the transmission power!
  - the reason is that there is a *tradeoff* between *the useful power required for data transmission* and the *power dissipated in the form of heat* at the power amplifier
- usually, the dissipation power (heat energy) *increases* as the transmission power *decreases*
- in fact most commercially available transmitters operate efficiently at one or two transmission power levels
  - below a certain level, the efficiency of the power amplifier *falls drastically*

COMMUNICATION SUBSYSTEM

- In some cheap transceivers, even when at the maximum transmission power mode, more than ***60%*** of the supply DC power is dissipated in the form of useless ***heat***.
- For example, the ***Chipcon*** *CC2420* transceiver has eight programmable output power levels ranging from −24 dBm to 0 dBm.

Chipcon CC2420: Output power settings and typical current consumption at 2.45 GHz

PA level	Output power dBm	Output power mW	Current consumption mA	Power consumption* mW
31	0	1	17.4	31.32
27	−1	0.794328235	16.5	29.7
23	−3	0.501187234	15.2	27.36
19	−5	0.316227766	13.9	25.02
15	−7	0.199526231	12.5	22.5
11	−10	0.1	11.2	20.16
7	−15	0.031622777	9.9	17.82
3	−25	0.003162278	8.5	15.3

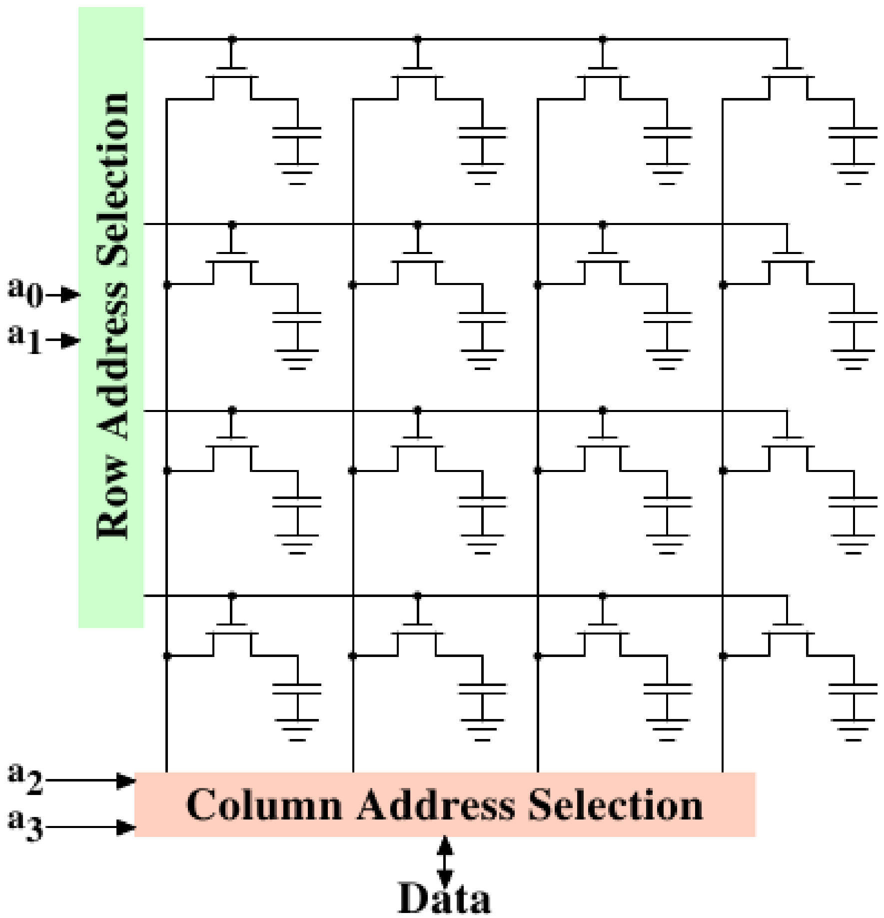
*$V_{dd} = 1.8$ V

BUS FREQUENCY AND RAM TIMING

- The processor subsystem consumes power when it interacts with the other subsystems via *the internal high speed buses.*
- The specific amount depends on the *frequency* and *bandwidth* of the communication.
- These two parameters can be optimally configured depending on the *interaction type*, but *bus protocol timings* are usually optimized for particular bus frequencies.
- Moreover, bus controller drivers require to be notified *when bus frequencies change* to ensure optimal performance.

ACTIVE MEMORY

- It is made up of *capacitor-transistor pairs (DRAM)*
  - arranged in rows and columns, each row being a single memory bank
  - have to be recharged periodically in order to store data



ACTIVE MEMORY

- The *refresh interval:*
  - a measure of the number of rows that must be refreshed
  - a *low refresh interval* corresponds to a *high clock frequency*
  - a *higher refresh interval* corresponds to a *low clock frequency*
- Consider two typical values: 2K and 4K
  - 2K: *refreshes more cells* at a low interval and completes the process faster, thus it *consumes more power*
  - 4K: *refreshes less cells* at a slower frequency, but it *consumes less power*
- A DRAM memory unit can be configured to operate in one of the following *power modes*:
  - temperature-compensated self-refresh mode
  - partial array self-refresh mode
  - power down mode

ACTIVE MEMORY

- *Temperature-compensated self-refresh mode*
  - the standard refresh rate of a memory unit can be adjusted according to its ambient *temperature.*
- *Partial array self-refresh mode*
  - the self-refresh rate can be *increased* if the entire memory array is not needed to store data
  - the refresh operation can be *limited* to the portion of the memory array in which data will be stored
- *Power down mode*
  - if no actual data storage is required, the supply voltage of most or the entire on-board memory array can be *switched off*

ACTIVE MEMORY

The *RAM timing* is another parameter that affects the power consumption of the memory unit

- it refers to the *latency* associated with accessing the memory unit
- before a processor subsystem accesses a particular cell in a memory, it should first *determine* the particular *row or bank*
- then *activate* the row with a *row access strobe (RAS)* signal
- the activated row can be accessed until the data is exhausted
- the time required to activate a row in a memory is t_{RAS}, which is relatively small but could *impact the system's stability* if set incorrectly

ACTIVE MEMORY

The *delays* between the activation of a row (a cell) and the writing of data into or reading of data from the cell is given as t_{RCD}.

- This time can be short or long, depending on how the memory cell is accessed.
- If it is accessed sequentially, it is insignificant.
- If it is accessed in a random fashion, the current active row must first be *deactivated* before a new row is activated.
- In this case, t_{RCD} can cause significant *latency.*

ACTIVE MEMORY

A memory cell is activated through a *column access strobe (CAS)*

- The delay between the CAS signal and the availability of valid data on the data pins is called *CAS latency.*
- *Low CAS latency* means *high performance* but also *high power consumption.*
- The time required to terminate one row access and begin the next row access is t_{RP}.
- The time required to switch rows and select the next cell for reading, writing, or refreshing is expressed as $t_{RP} + t_{RCD}$.
- The duration of time required between the active and precharge commands is called t_{RAS}.
- It is a measure of how long the processor must wait before the next memory access can begin.

ACTIVE MEMORY

Parameters of RAM timing

Parameter	Description
RAS	Row Address Strobe or Row Address Select
CAS	Column Address Strobe or Column Address Select
t_{RAS}	A time delay between the precharge and activation of a row
t_{RCD}	The time required between RAS and CAS access
t_{CL}	CAS latency
t_{RP}	The time required to switch from one row to the next row
t_{CLK}	The duration of a clock cycle
Command rate	The delay between Chip Select (CS)
Latency	The total time required before data can be written to or read from memory

When a RAM is accessed by *clocked logic*, the times are generally rounded up to the *nearest clock cycle*

- for example, when accessed by a 100-MHz processor (with 10 ns clock duration), a 50-ns SDRAM can perform the first read in 5 clock cycles and additional reads within the same page every 2 clock cycles.
- this is generally described as “5 – 2 – 2 – 2” timing.

POWER SUBSYSTEMS

- The power subsystem supplies power to all the other subsystems
- It consists of:
  - ***The battery***
  - ***The DC – DC converter***
    - It provides the right amount of supply voltage to each individual hardware component
    - By transforming the main DC supply voltage into a suitable level
    - The transformation can be a *step-down (buck)*, a *step-up (boost)*, or an *inversion (flyback)* process, depending on the requirements of the individual subsystem

POWER SUBSYSTEMS

The battery

- A wireless sensor node is powered by *exhaustible batteries*
  - the main factor affect the quality of these batteries is *cost.*
- Batteries are specified by a rated current capacity, *C*, expressed in *ampere-hour*
  - This quantity describes the rate at which a battery discharges without significantly affecting the prescribed supply voltage
  - As the discharge rate *increases*, the rated capacity *decreases*
  - Most portable batteries are rated at 1*C*
    - This means a 1000 mAh battery provides 1000mA for 1 hour, if it is discharged at a rate of 1*C.*
      - *e.g.,* at a rate of 0.5*C*, providing 500mA for 2 hours.
      - at a rate of 2*C*, 2000mA for 30 minutes.

The battery

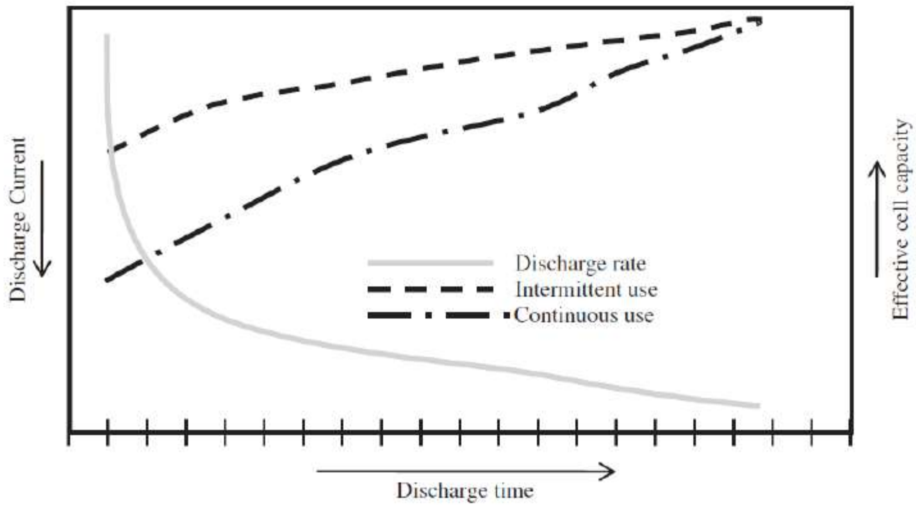
In reality, batteries perform at *less than the prescribed rate*. Often, the *Peukert Equation* is applied to quantifying the capacity offset (i.e., how long a battery lasts in reality):

$$t = \frac{C}{I^n}$$

- Where *C* is the theoretical capacity of the battery expressed in ampere-hours
- *I* is the current drawn in Ampere (A)
- *t* is the time of discharge in seconds
- *n* is the Peukert number, a constant that directly relates to the internal resistance of the battery (A value close to 1 indicates that the battery performs well; the higher the number, the more capacity is lost when the battery is discharged at high current.)

Drawing current at a rate greater than the discharge rate results in a current consumption rate higher than the rate of diffusion of the active elements in the electrolyte. If this process continues for a long time, the electrodes run out of active material even though the electrolyte has not yet exhausted its active materials. This situation can be overcome by intermittently drawing current from the battery.

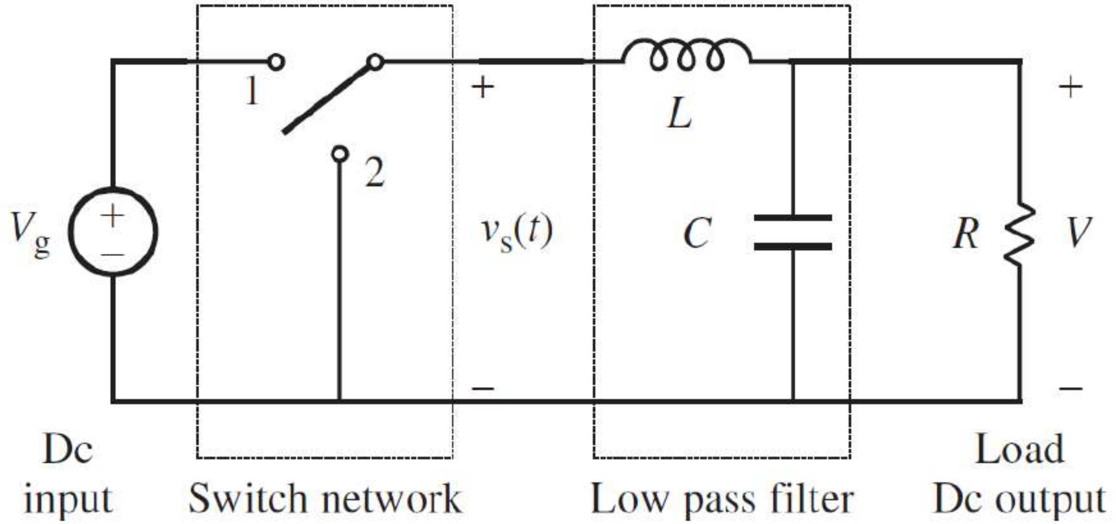
by intermittently using the battery, it is possible during quiescent periods to increase the diffusion and transport rates of active ingredients and to match up the depletion created by excessive discharge



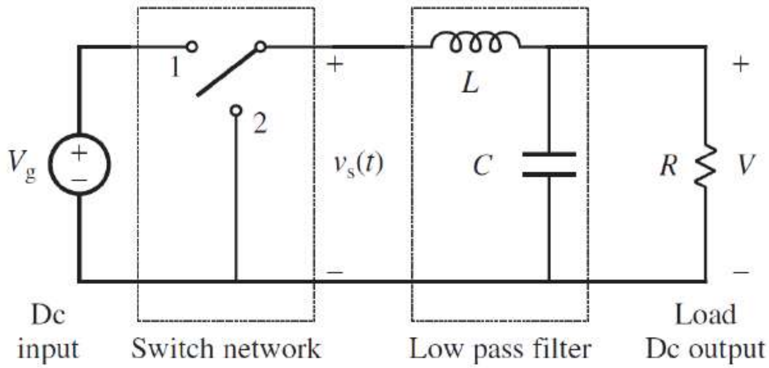
because of this potential for recovery, the capacity reduction can be undermined and the operating efficiency can be enhanced

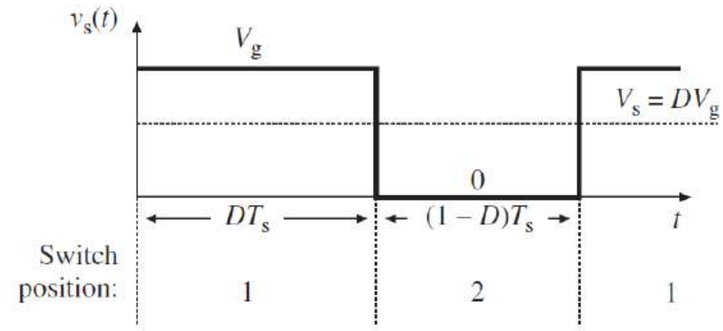
POWER SUBSYSTEMS

- The DC – DC converter transforms one voltage level into another
  - *The main problem* is its *conversion efficiency*
- A typical DC – DC convertor consists of
  - A power supply
  - A switching circuit
  - A filter circuit
  - A load resistor



A DC–DC converter consisting of a supply voltage, a switching circuit, a filter circuit, and a load resistance.





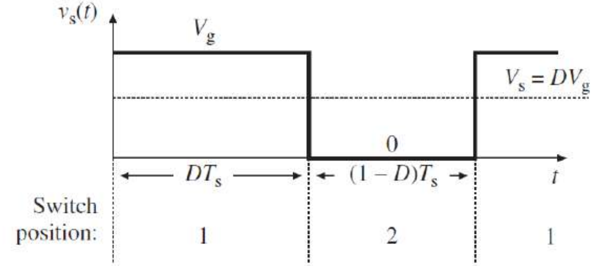
The output voltage of a switching circuit of a DC–DC converter

In the figure, the circuit consists of *a single-pole, double-throw (SPDT) switch*

- SPDT is connected to a DC supply voltage, *Vg.*
- Considering the inductor, *L*, as a short circuit.
- The capacitor, *C*, as an open circuit for the DC supply voltage.
- The switch's output voltage, *Vs (t)* = *Vg* when the switch is in position 1.
- *s (t)* = 0 When it is in position 2.
- Varying the position of the switch at a frequency, *fs* yields a periodically varying square wave, *vs (t),* that has a period *Ts = 1/fs.*
- *vs (t)* can be expressed by a duty cycle *D.*
- *D* describes the fraction of time that the switch in position 1, (0 ≤ D ≤ 1).

POWER SUBSYSTEMS

- A DC – DC converter is realized
  - by employing active switching components
  - such as diodes and power MOSFETs



The output voltage of a switching circuit of a DC–DC converter

- Using the *inverse Fourier transformation*
  - The DC component of *vs (t)* (*Vs*) is described as:

$$V_s = \frac{1}{T_s}\int_0^{T_s} v_s(t)\mathrm{d}t = DV_g$$

which is the average value of *vs (t).*

- In other words, the integral value represents the area under the waveform of Figure for a single period, or the height of *Vg* multiplied by the time *Ts.*
- It can be seen that the switching circuit reduces the DC component of the supply voltage by a factor that equals to the duty cycle, *D* Since 0 ≤ *D* ≤ 1 holds, the expression: *Vs* ≤ *Vg* is true.

POWER SUBSYSTEMS

The switching circuit consumes power

- Due to the existence of a resistive component in the switching circuit, there is power dissipation
- The efficiency of a typical switching circuit is between 70 and 90%
- In addition to the desired DC voltage, $v_s(t)$ also contains undesired harmonics of the switching frequency, f_s
- These harmonics must be removed so that the converter's output voltage $v(t)$ is essentially equal to the DC component $V = V_s$
- For this purpose, a DC – DC converter employs a *low pass filter*

POWER SUBSYSTEMS

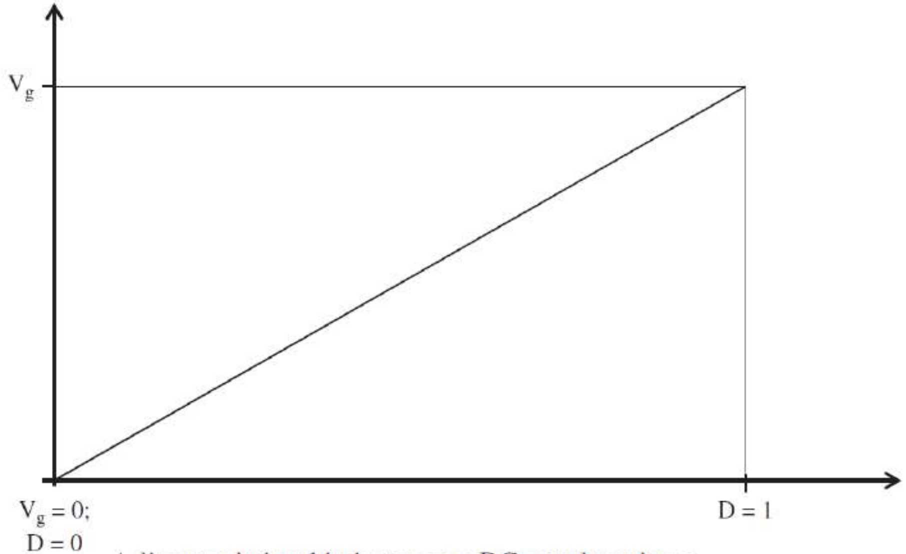
- In Figure, a first-order *LC* low pass filter is connected to the switching circuit
  - The filter's cutoff frequency is given by: $f_c = \dfrac{1}{2\pi\sqrt{LC}}$
  - The cutoff frequency, *fc*, should be sufficiently *less* than the switching frequency, *fs*
  - So that the low pass filter allows only the DC component of *vs (t)* to pass
- In an ideal filter, there is no power dissipation
  - Because the passive components (inductors and capacitors) are energy storage components
- Subsequently, the DC–DC converter produces a DC output voltage
  - Its magnitude is controlled by the duty cycle, *D*, using circuit elements that (ideally) do not dissipate power

POWER SUBSYSTEMS

- The conversion ratio, *M(D)*, is defined as the ratio of the DC output voltage, *V* , to the DC input voltage, *Vg*, under a steady-state condition:

$$M(D) = \frac{V}{V_g}$$

- For the buck converter shown in Figure, *M(D) = D*
- Figure illustrates the linear relationship between the input DC voltage, *Vg* and the switching circuit's duty cycle, *D.*

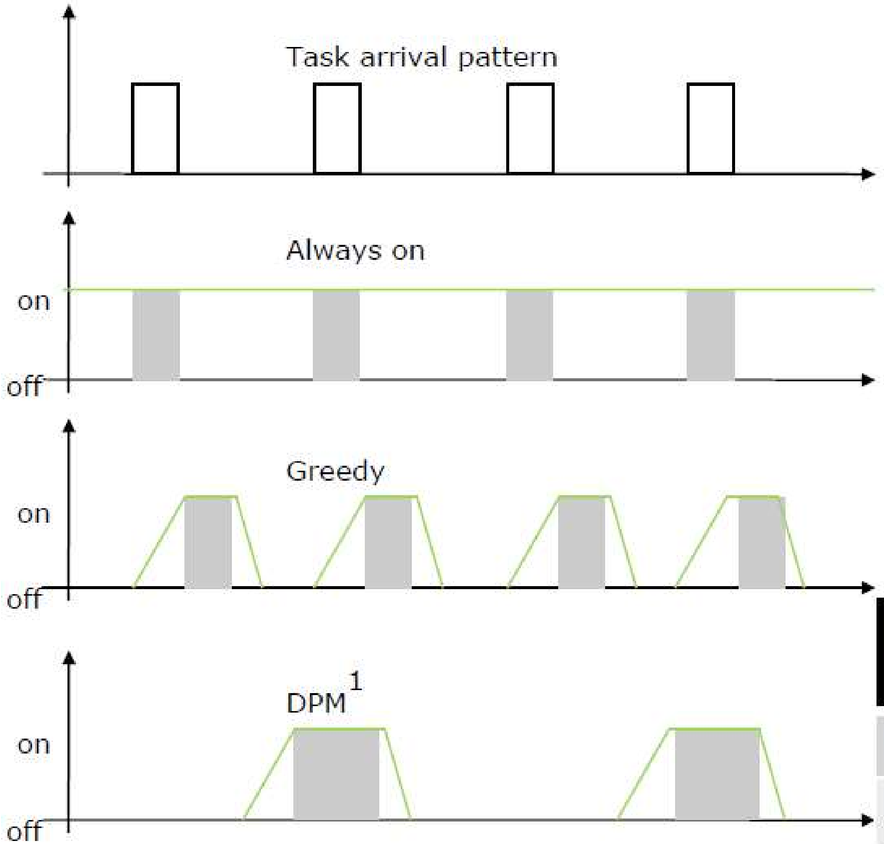


A linear relationship between a DC supply voltage and the duty cycle of a switching circuit.

DYNAMIC OPERATION MODES

- In general, a subcomponent or a part it can have *n* different power modes
  - If there are *x* hardware components that can have *n* distinct power consumption levels, a DPM strategy can define $x \times n$ différent power mode configurations, *Pn.*
- The *task* of the DPM strategy is:
  - select the optimal configuration that matches the activity of a wireless sensor node.
- Two associated *challenges*:
  - Transition between the different power configurations costs extra power.
  - A transition has an associated delay and the potential of missing the occurrence of an interesting event.

DYNAMIC OPERATION MODES

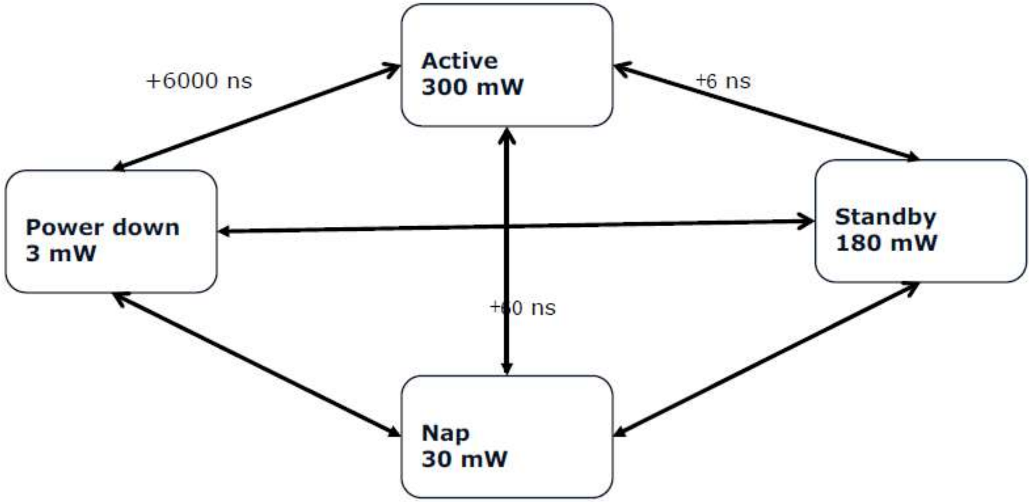


Parameter	Value
P_{on}	10 W
P_{off}	0 W
$P_{on \rightarrow off}$	10 W
$P_{off \rightarrow on}$	40 W
$t_{on \rightarrow off}$	1 s
$t_{off \rightarrow on}$	2 s
t_R	25 s

Policy	Energy	Avg. Latency
Always on	250 J	1 s
Reactive greedy	240 J	3 s
Power-aware	140 J	2.5 s

DYNAMIC OPERATION MODES

- ## Memory Access

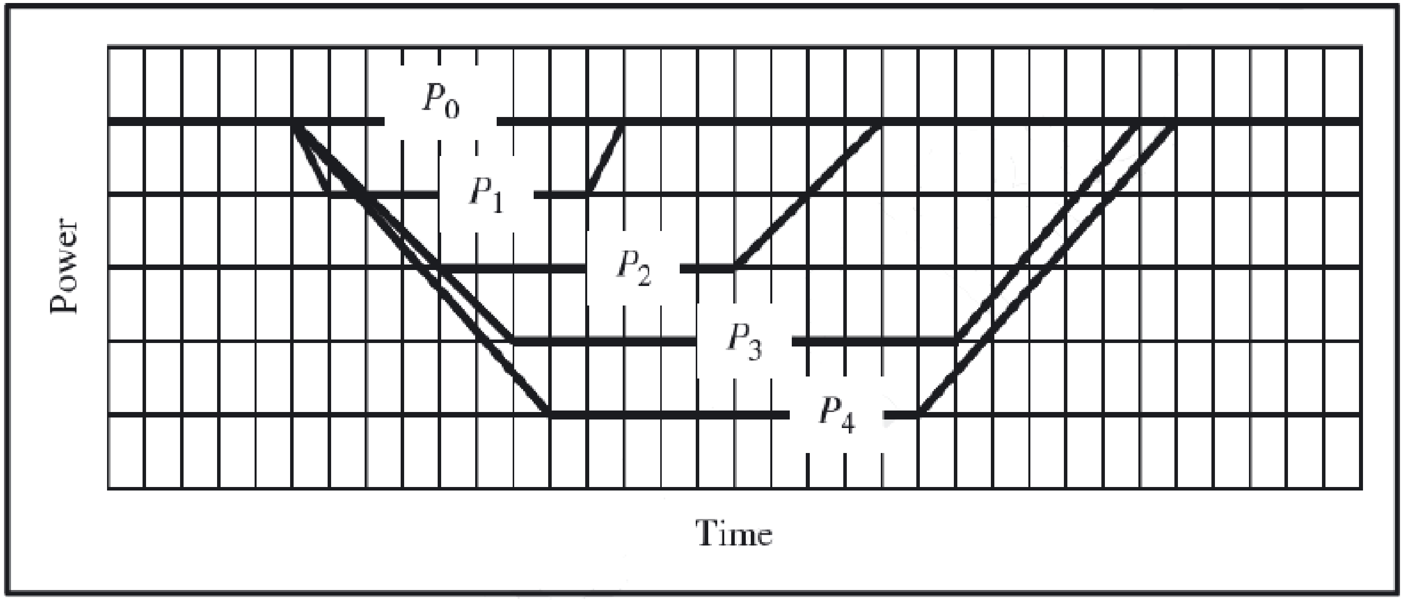


Power Mode	StrongARM	Memory	MEMS & ADC	RF
P_0	Sleep	Sleep	Off	Off
P_1	Sleep	Sleep	On	Off
P_2	Sleep	Sleep	On	RX
P_3	Idle	Sleep	On	RX
P_4	Active	Active	On	TX, RX

DYNAMIC OPERATION MODES

Power saving configurations

Configuration	Processor	Memory	Sensing subsystem	Communication subsystem
P_0	Active	Active	On	Transmitting/receiving
P_1	Active	On	On	On (transmitting)
P_2	Idle	On	On	Receiving
P_3	Sleep	On	On	Receiving
P_4	Sleep	Off	On	Off
P_5	Sleep	Off	Off	Off



Transition between different power modes and the associated transition costs.

DYNAMIC OPERATION MODES

- The decision for a particular power mode depends on
  - The anticipated task in the queues of the different hardware components
- Failure to realistically estimate future tasks can cause a node to miss interesting events or to delay in response
- In a WSN, the events outside of the network cannot be modeled as deterministic phenomena
  - e.g., a leak in a pipeline; a pestilence in a farm
  - No need for setting up a monitoring system
- An accurate event arrival model enables a DPM strategy to decide for the right configuration that has a *long duration* and *minimal power consumption*

DYNAMIC OPERATION MODES

Transition Cost

Suppose:

- Each subsystem of a wireless sensor node operates in just two different power modes only, it can be either *on* or *off.*
- Moreover, assume that the transition from *on* to *off* does not have an associated power cost
- But the reverse transition (from *off* to *on*) has a cost in terms both of power and a time delay.
- These costs are justified if the power it saves in the *off* state is large enough.
- In other words, the amount of the *off* state power is considerably large and the duration of the *off* state is long
- It is useful to quantify these costs and to set up a transition threshold

DYNAMIC OPERATION MODES

Transition Cost

Suppose:

- The minimum time that a subsystem stays in an *off* state is t_{off}
- The power consumed during this time is P_{off}
- The transition time is $t_{off,on}$
- The power consumed during the transition is $p_{off,on}$
- The power consumed in an *on* state is P_{on}. Hence:

$$P_{\text{off}} \cdot t_{\text{off}} + P_{\text{off,on}} \cdot t_{\text{off,on}} \geq P_{\text{on}} \cdot \left(t_{\text{off}} + t_{\text{off,on}}\right) \quad \text{....... Eq. 1}$$

- Therefore, t_{off} is justified if:

$$t_{\text{off}} \geq \max\left(0, \frac{\left(P_{\text{on}} - P_{\text{off,on}}\right) \cdot t_{\text{off,on}}}{P_{\text{on}} - P_{\text{off}}}\right) \quad \text{....... Eq. 2}$$

DYNAMIC OPERATION MODES

Transition Cost

Equations .5) and (8.6) can describe a subsystem with *n* distinct operational power modes

- In this case a transition from any state *i* into *j* is described as *ti,j*
- Hence, the transition is justified if Equation (8.7) is satisfied

$$t_j \geq \max\left(0, \frac{\left(P_i - P_{j,k}\right) t_{i,j}}{P_i - P_j}\right) \quad \text{....... Eq. 3}$$

where *tj* is the duration of the subsystem in state *j*

NODE AND NETWORK MANAGEMENT

TIME SYNCHRONIZATION

OUTLINES

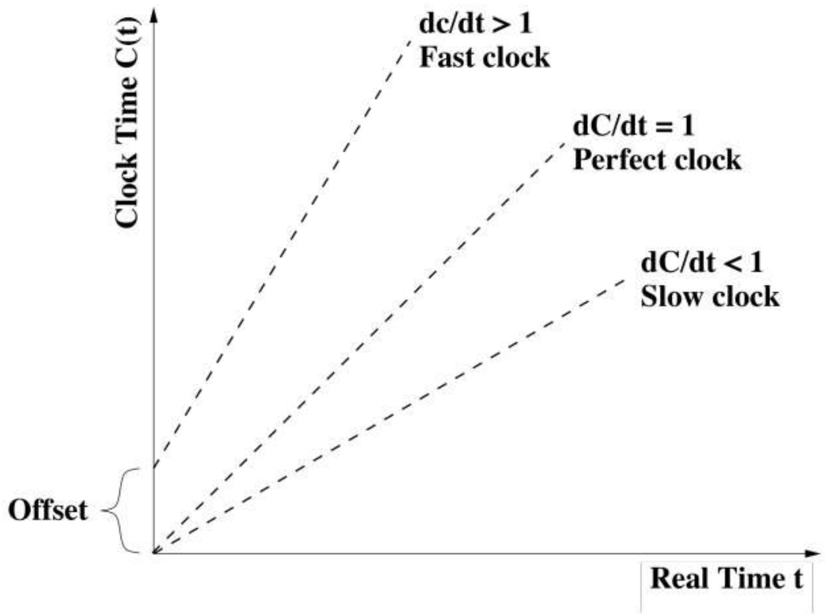
- The time synchronization problem
- Time synchronization in wireless sensor networks
- Basic techniques for time synchronization
- Time synchronization protocols

CLOCKS AND THE SYNCHRONIZATION PROBLEM

- Common time scale among sensor nodes is important for a variety of reasons
  - identify causal relationships between events in the physical world
  - support the elimination of redundant data
  - facilitate sensor network operation and protocols
- Typical clocks consist of quartz-stabilized oscillator and a counter that is decremented with every oscillation of the quartz crystal
- When counter reaches 0, it is reset to original value and interrupt is generated
- Each interrupt (clock tick) increments software clock (another counter)
- Software clock can be read by applications using API
- Software clock provides local time with C(t) being the clock reading at real time t
- Time resolution is the distance between two increments (ticks) of software clock

CLOCK PARAMETERS

- Clock offset: difference between the local times of two nodes
- Synchronization is required to adjust clock readings such that they match
- Clock rate: frequency at which a clock progresses
- Clock skew: difference in frequencies of two clocks
- Clock rate dC/dt depends on temperature, humidity, supply voltage, age of quartz, etc., resulting in drift rate (dC/dt-1)



CLOCK PARAMETERS

- Maximum drift rate ρ given by manufacturer (typical 1ppm to 100ppm)

- Guarantees that: Drift rate causes clocks to differ even after synchronization

$$1 - \rho \le \frac{dC}{dt} \le 1 + \rho$$

- Two synchronized identical clocks can drift from each other at rate of at most

  $2\rho_{max}$

- To limit relative offset to δ seconds, the resynchronization interval τ_{sync} must meet the requirement:

$$\tau_{sync} \le \frac{\delta}{2\rho_{\max}}$$

CLOCK PARAMETERS

- C(t) must be piecewise continuous (strictly monotone function of time)

  - Clock adjustments should occur gradually, e.g., using a linear compensation function that changes the slope of the local time

  - Simply jumping forward/backward in time can have unintended consequences
    - time-triggered events may be repeated or skipped

TIME SYNCHRONIZATION

- External synchronization
  - clocks are synchronized with external source of time (reference clock)
  - reference clock is accurate real-time standard (e.g., UTC)

- Internal synchronization
  - clocks are synchronized with each other (no support of reference clock)
  - goal is to obtain consistent view of time across all nodes in network
  - network-wide time may differ from external real-time standards

- External synchronization also provides internal synchronization

- Accuracy: maximum offset of a clock with respect to reference clock

- Precision: maximum offset between any two clocks

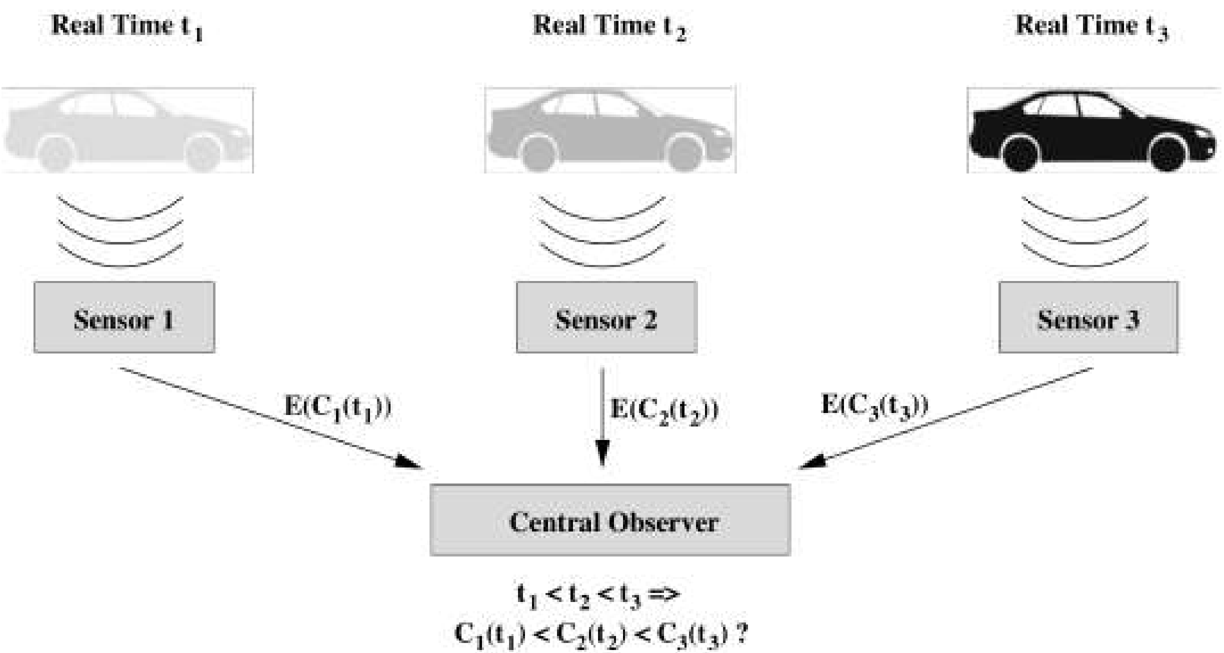
- If two nodes synchronized externally with accuracy of Δ, also synchronized internally with precision 2Δ

WHY TIME SYNCHRONIZATION IN WSNs?

- Sensors in WSNs monitor objects and events in the physical world

- Accurate temporal correlation is crucial to answer questions such as
  - ❑ how many moving objects have been detected?
  - ❑ what is the direction of the moving object?
  - ❑ what is the speed of the moving object?

- If real-time ordering of events is t1<t2<t3, then sensor times should reflect this ordering: C1(t1)<C2(t2)<C3(t3)



WHY TIME SYNCHRONIZATION IN WSNs?

- Time difference between sensor time stamps should correspond to real-time differences: $\Delta = C2(t2) - C1(t1) = t2 - t1$
  - important for data fusion (aggregation of data from multiple sensors)

- Synchronization needed by variety of applications and algorithms
  - communication protocols (at-most-once message delivery)
  - security (limit use of keys, detect replay attacks)
  - data consistency (caches, replicated data)
  - concurrency control (atomicity and mutual exclusion)
  - medium access control (accurate timing of channel access)
  - duty cycling (know when to sleep or wake up)
  - localization (based on techniques such as time-of-flight measurements)

CHALLENGES FOR TIME SYNCHRONIZATION IN WSNs?

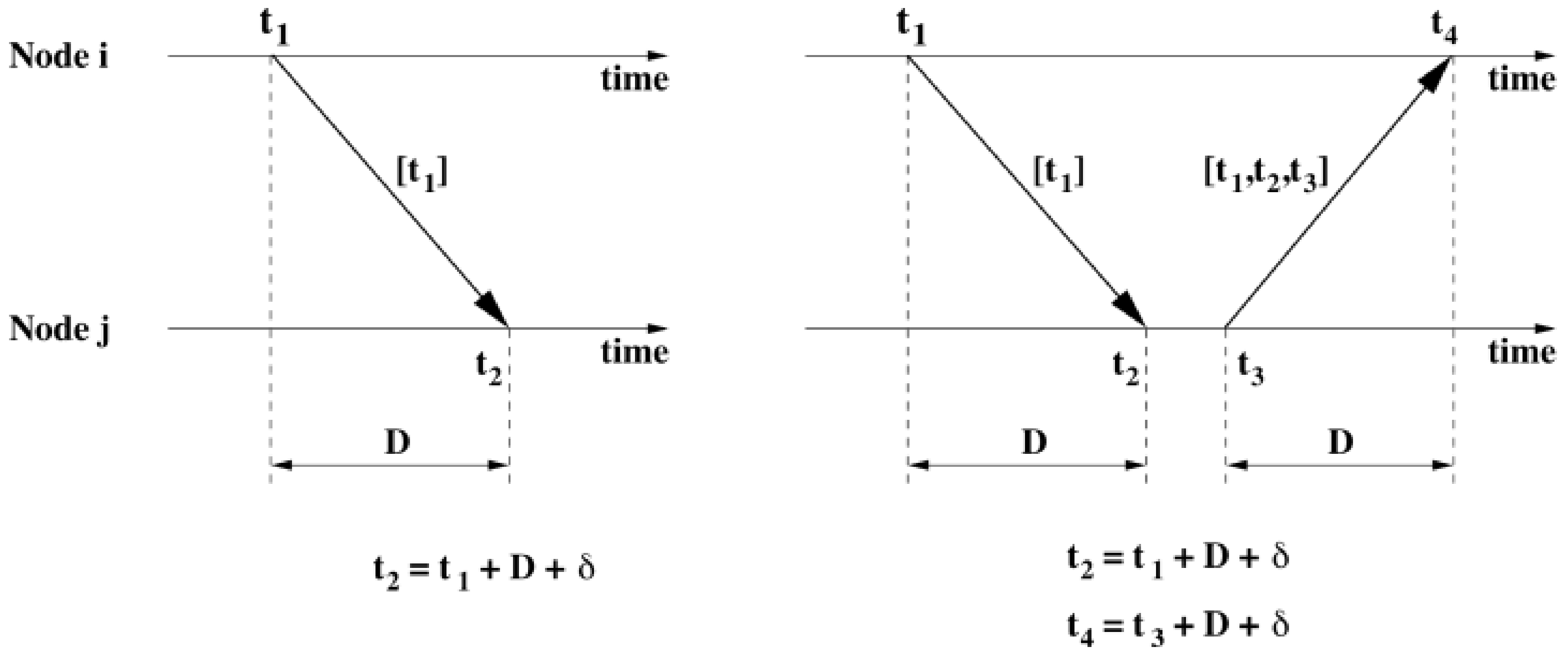
- Traditional protocols (e.g., NTP) are designed for wired networks
- WSNs pose a variety of additional challenges
- Environmental effects
  - ❑ sensors often placed in harsh environments
  - ❑ fluctuations in temperature, pressure, humidity
- Energy constraints
  - ❑ finite power sources (batteries)
  - ❑ time synchronization solutions should be energy-efficient
- Wireless medium and mobility
  - ❑ throughput variations, error rates, radio interferences, asymmetric links
  - ❑ topology changes, density changes
  - ❑ node failure (battery depletion)
- Other challenges
  - ❑ limited processor speeds or memory (lightweight algorithms)
  - ❑ cost and size of synchronization hardware (GPS)

SYNCHRONIZATION MESSAGES

- Pairwise synchronization: two nodes synchronize using at least one message

- Network-wide synchronization: repeat pairwise synchronization throughout network

- One-way message exchange:
  - ❑ single message containing a time stamp
  - ❑ difference can be obtained from (t2-t1)=D+δ (D=propagation delay)

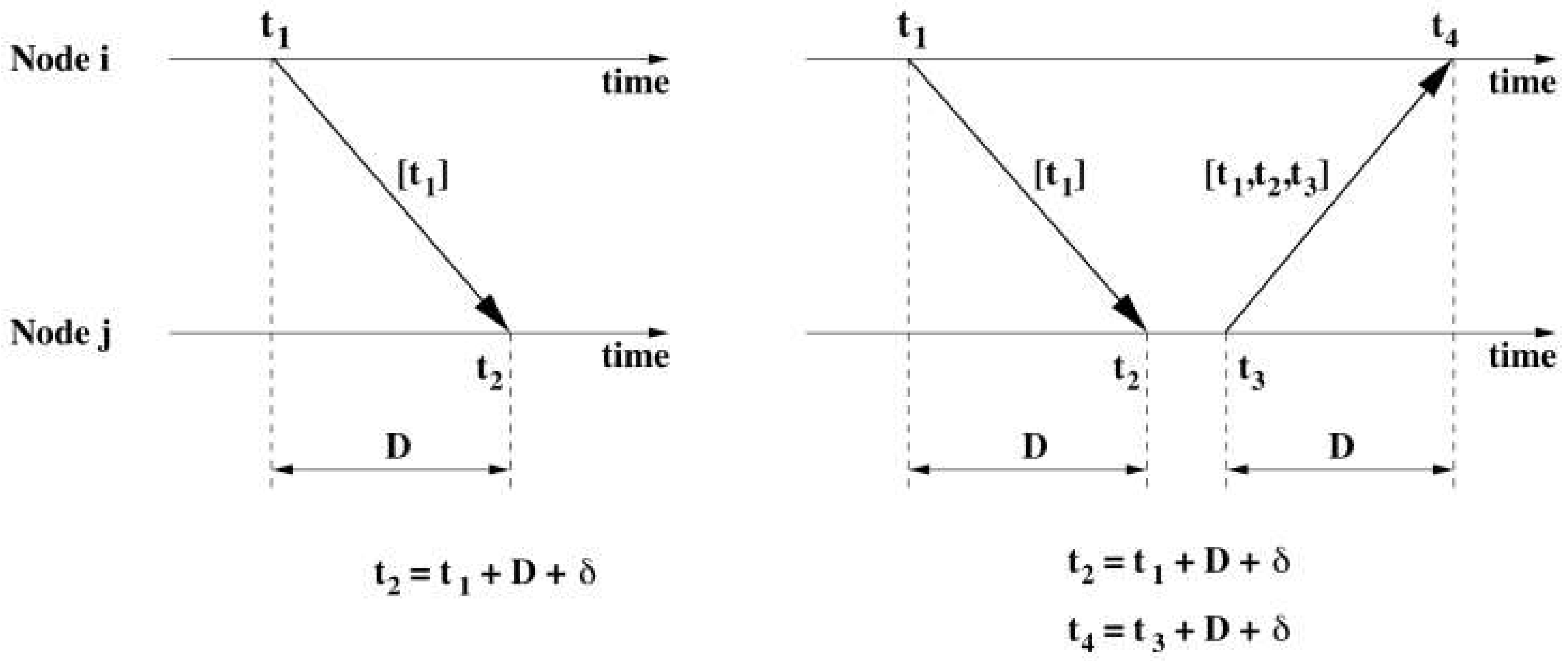


SYNCHRONIZATION MESSAGES

- Two-way message exchange:
  - receiver node responds with message containing three time stamps
  - assumption: propagation delay is identical in both directions and clock drift does not change between measurements

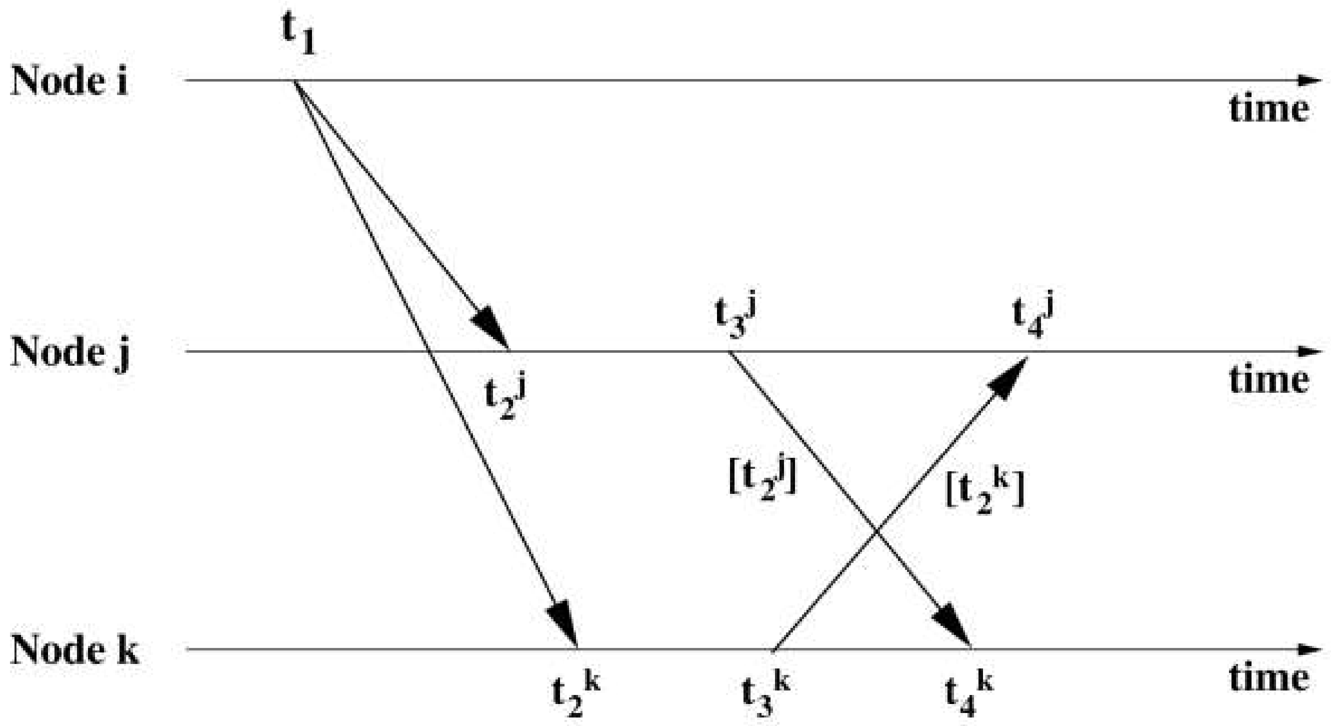
$$D = \frac{(t_2 - t_1) + (t_4 - t_3)}{2}$$

$$offset = \frac{(t_2 - t_1) - (t_4 - t_3)}{2}$$



RECEIVER-RECEIVER SYNCHRONIZATION

- So far: sender-receiver approaches
- Receiver-receiver: multiple receivers of broadcast messages exchange their message arrival times to compute offsets among them
- Example: 2 receivers; 3 messages (1 broadcast, 2 exchange messages)
- No time stamp in broadcast message required



NONDETERMINISM OF COMMUNICATION LATENCY

- Several components contribute to total communication latency

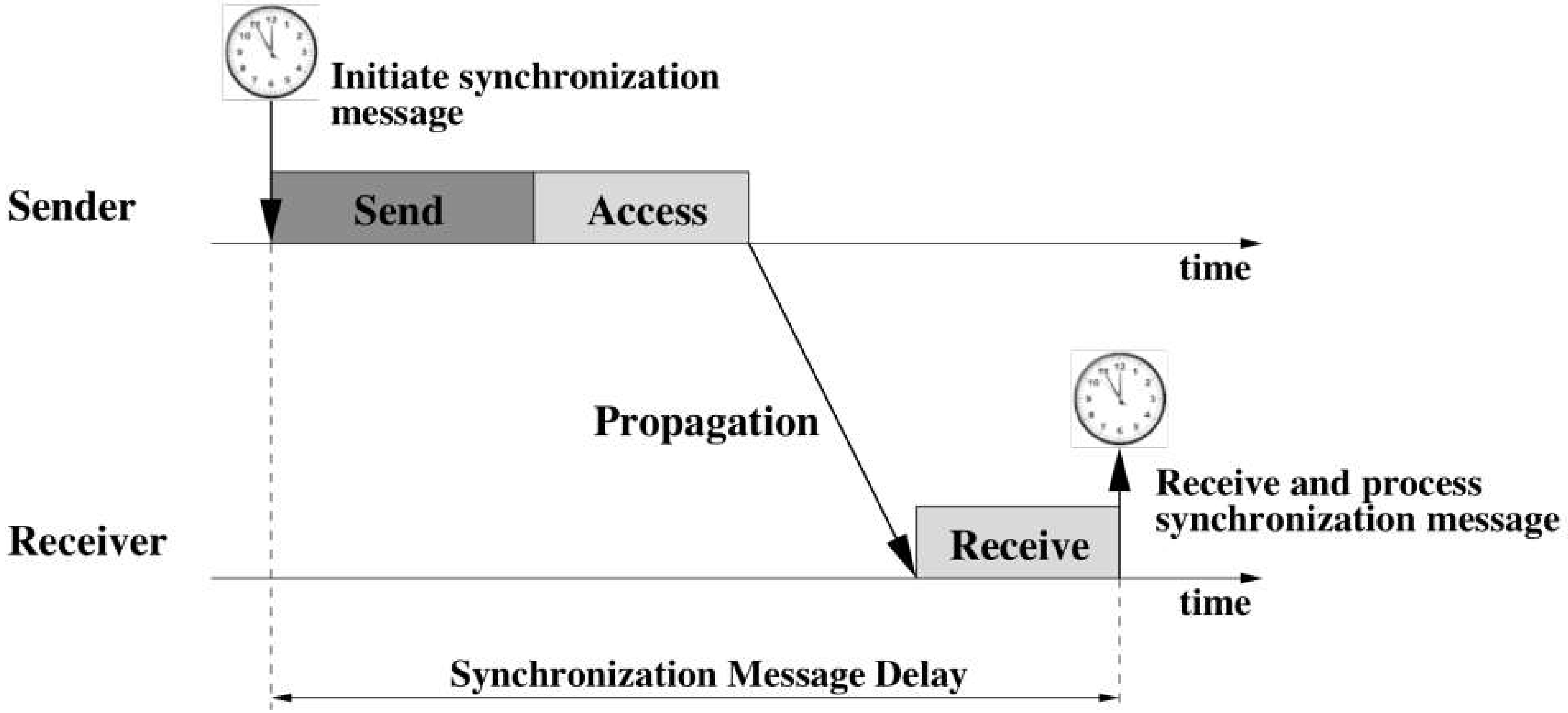
- Send delay:
  - generation of synchronization message
  - passing message to network interface
  - includes delays caused by OS, network protocol stack, device driver

- Access delay:
  - accessing the physical channel
  - mostly determined by medium access control (MAC) protocol

- Propagation delay:
  - actual time for message to travel to sender (typically small)

- Receive delay:
  - receiving and processing the message
  - notifying the host (e.g., interrupt)

NONDETERMINISM OF COMMUNICATION LATENCY



REFERENCE BROADCASTS

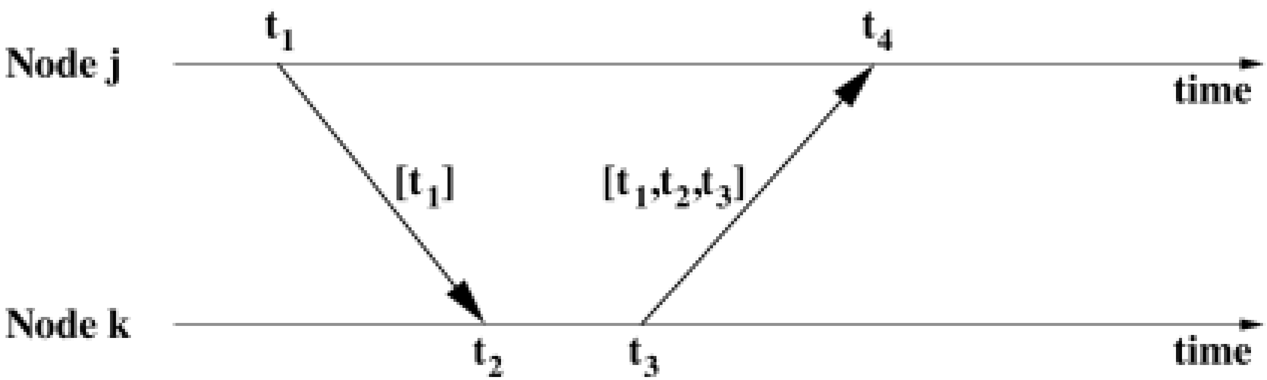
- Global Positioning System (GPS) is a well-known global source of time
  - ❑ time measured from epoch started at 0h January 6, 1980 UTC
  - ❑ unlike UTC, GPS not perturbed by leap seconds
  - ❑ GPS is ahead by 15 seconds (and increasing)

- Terrestrial radio stations
  - ❑ WWV/WWVH & WWVB (National Institute of Standards & Technology)
  - ❑ continuously broadcast time based on atomic clocks

- Problems with these techniques:
  - ❑ not universally available (underwater, indoors, outer space)
  - ❑ need for high-power receivers
  - ❑ size
  - ❑ cost

LIGHTWEIGHT TREE BASED SYNCHRONIZATION

- Goal of LTS is to provide specified precision with little overhead
- Based on pairwise synchronization:
  - message from j to k, containing time stamp t1 (j's clock)
  - message from k to j, containing t1 (j's clock) and t2, t3 (k's clock)



- assuming message delay D

$$offset = \frac{t_2 - t_4 - t_1 + t_3}{2}$$

LIGHTWEIGHT TREE BASED SYNCHRONIZATION

- Centralized multi-hop version of LTS
  - reference node is root of spanning tree containing all nodes
  - breadth first search used to construct tree
  - once tree established, reference nodes synchronizes with children
  - errors from pairwise synchronization are additive
    - keep depth of tree small
  - overhead of pairwise synchronization: 3 messages
  - overhead of network-wide synchronization: 3n-3 messages (n edges)

LIGHTWEIGHT TREE BASED SYNCHRONIZATION

- Distributed multi-hop version of LTS
  - One or more reference nodes contacted by sensors whenever synchronization is required
  - Nodes determine resynchronization period based on desired clock accuracy, distance to reference node, clock drift ρ, time of last synchronization
  - Node can query neighbors for pending synchronization requests, i.e., node synchronizes with neighbor instead of reference node

TIMING-SYNC PROTOCOL FOR SENSOR NETWORKS

- TPSN is another sender-receiver technique
  - Uses a tree to organize network
  - Uses two phases for synchronization
  - Discovery phase
  - Synchronization phase

TIMING-SYNC PROTOCOL FOR SENSOR NETWORKS

- Level discovery phase
  - establish hierarchical topology
    - root resides at level 0
  - root initiates phase by broadcasting level_discovery message (contains level and identity of sender)
  - receiver can determine own level (level of sender plus one)
  - receiver re-broadcasts message with its own identity and level
  - receiver discards multiple received broadcasts
  - if node does not know its level (corrupted messages, etc.), it can issue level_request message to neighbors (which reply with their levels)
    - node's level is then one greater than the smallest level received
    - node failures can be handled similarly (i.e., if all neighbors at level i-1 disappear, node issues level_request message
    - if root node dies, nodes in level 1 execute leader election algorithm

TIMING-SYNC PROTOCOL FOR SENSOR NETWORKS

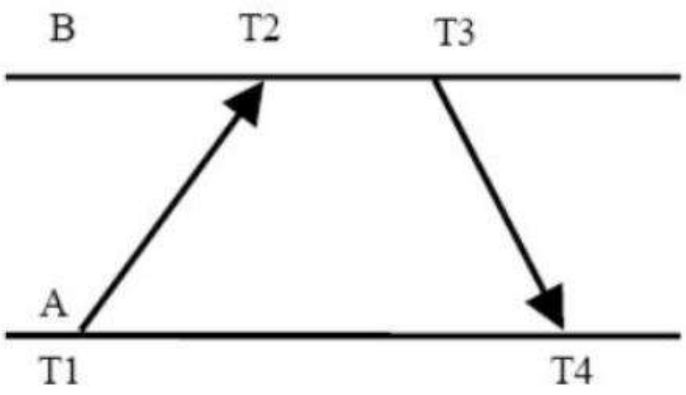
- Synchronization phase
  - pairwise synchronization along the edges of hierarchical structure

  - each node on level i synchronizes with nodes on level i-1
    - approach similar to LTS:
      - node A issues synchronization pulse at t1 (containing level and time stamp)
      - node B receives message at t2 and responds with an ACK at t3 (containing t1, t2, t3, and level)
      - node A receives ACK at t4

  - node A calculates drift and propagation delay

$$D = \frac{(t_2 - t_1) + (t_4 - t_3)}{2}$$

$$offset = \frac{(t_2 - t_1) - (t_4 - t_3)}{2}$$



TIMING-SYNC PROTOCOL FOR SENSOR NETWORKS

- Synchronization phase (contd.)
  - phase initiate by root node issuing time_sync packet
  - after waiting for random interval (to reduce contention), nodes in level 1 initiate two-way message exchange with root node
  - nodes on level 2 will overhear synchronization pulse and initiate two-way message exchange with level 1 nodes after random delay
  - process continues throughout network

- Synchronization error of TPSN
  - depth of hierarchical structure
  - end-to-end latencies

FLOODING TIME SYNCHRONIZATION PROTOCOL

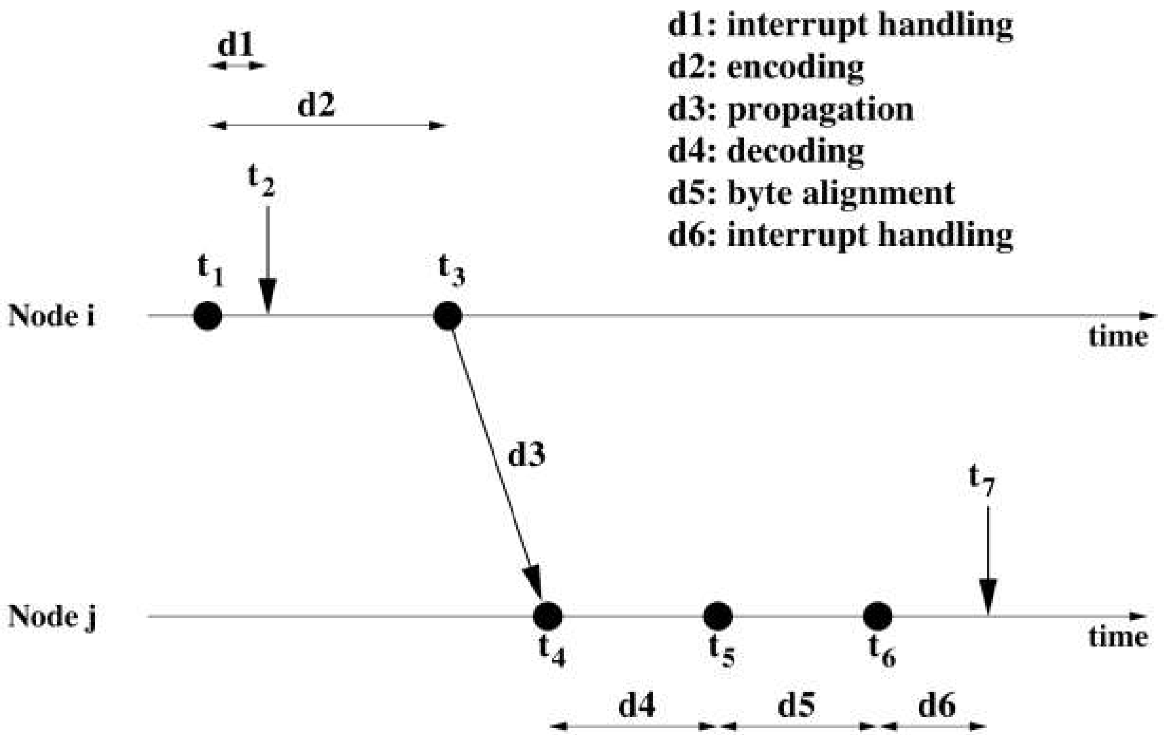
- Goals of FTSP include:
  - network-wide synchronization with errors in microsecond range
  - scalability up to hundreds of nodes
  - robustness to topology changes

- FTSP uses single broadcast message to establish synchronization points

- Decomposes end-to-end delay into different components

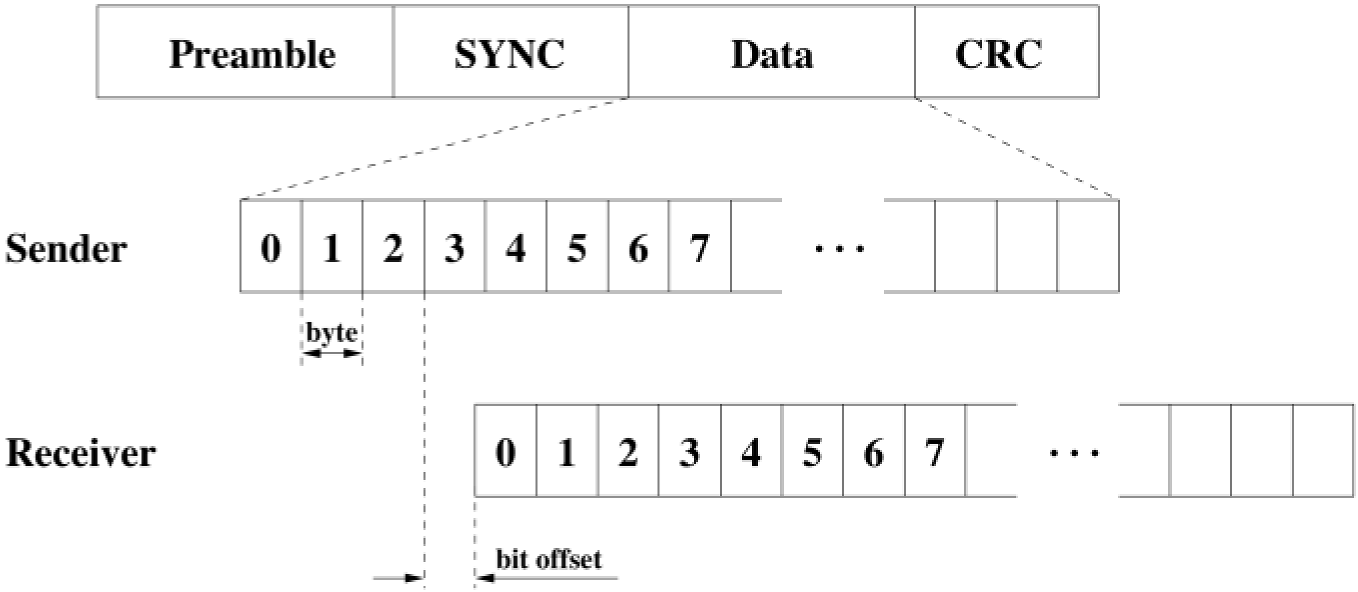
FLOODING TIME SYNCHRONIZATION PROTOCOL

- t1: wireless radio informs CPU that it is ready for next message
- d1: interrupt handling time (few microseconds)
- t2: CPU generates time stamp
- d2: encoding time (transform message into electromagnetic waves; deterministic, low hundreds of microseconds)
- d3: propagation delay (from t3 on node i to t4 on node j; typically very small and deterministic)
- d4: decoding time (deterministic, low hundreds of microseconds)
- d5: byte alignment time (delay caused by different byte alignments (bit offsets), i.e., receiving radio has to determine the offset from a known synchronization byte and then shift incoming message accordingly); can reach several hundreds of microseconds
- t7: interrupt, CPU obtains time stamp



FLOODING TIME SYNCHRONIZATION PROTOCOL

- Time-stamping in FTSP
  - sender sends single broadcast containing time stamp (estimated global time)
  - receiver extracts time stamp from message and time-stamps arrival (leads to global-local time pair, providing a synchronization point)
  - synchronization message begins with preamble followed by SYNC bytes, data field, and CRC
  - preamble bytes are used to synchronize receiver radio to carrier frequency
  - SYNC bytes are used to calculate bit offset



FLOODING TIME SYNCHRONIZATION PROTOCOL

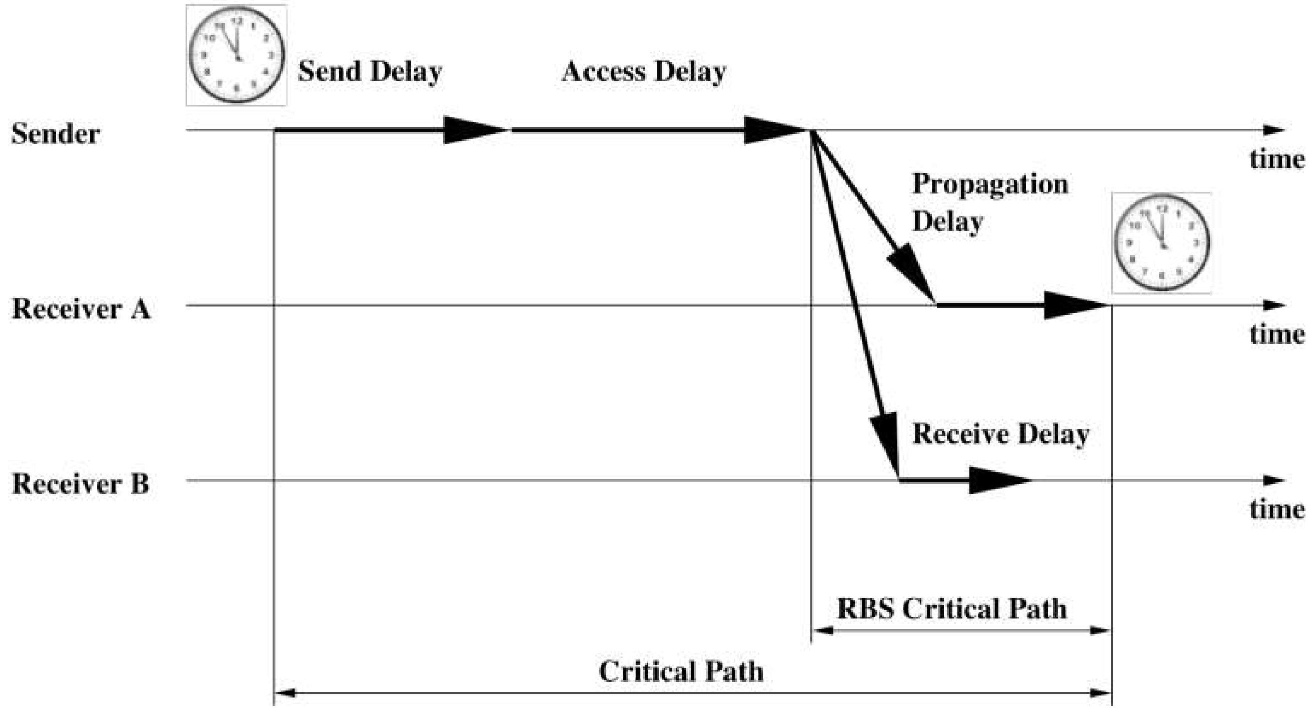
- Time-stamping in FTSP (contd.)
  - ❑ multiple time stamps are used at both sender and receiver to reduce jitter of interrupt handling and encoding/decoding times
  - ❑ time stamps are recorded at each byte boundary after the SYNC bytes as they are transmitted or received
  - ❑ time stamps are normalized by subtracting appropriate integer multiple of nominal byte transmission time (e.g., approx. 417µs on Mica2)
  - ❑ jitter in interrupt handling can be reduced by taking the minimum of normalized time stamps
  - ❑ jitter in encoding/decoding can be reduced by averaging these corrected normalized time stamps
  - ❑ final (error-corrected) time stamp is added into data part of message
  - ❑ at receiver side, time stamp must further be corrected by the byte alignment time (can be determined from transmission speed and bit offset)

FLOODING TIME SYNCHRONIZATION PROTOCOL

- Multi-hop synchronization
  - ❑ root node is elected based on unique node IDs
  - ❑ root node maintains global time and all other nodes synchronize to root
  - ❑ synchronization is triggered by broadcast message by the root node
    - ✓ whenever node does not receive synchronization message for certain amount of time, it declares itself to be the new root
    - ✓ whenever root receives a message from node with lower node ID, it gives up root status
  - ❑ all receiver nodes within range establish synchronization points
  - ❑ other nodes establish synchronization points from broadcasts of synchronized nodes that are closer to the root
  - ❑ a new node joining the network with lowest node ID will first collect synchronization messages to adjust its own clock before claiming root status

REFERENCE-BROADCAST SYNCHRONIZATION

- Key idea of RBS: in the wireless medium, broadcast messages will arrive at receivers at approximately the same time
  - ❑ set of receivers synchronize with each other using a broadcast message
  - ❑ variability in message delay dominated by propagation delay and time needed to receive and process incoming message (send delay and access delay are identical)
  - ❑ RBS critical path is short than critical path of traditional technique



REFERENCE-BROADCAST SYNCHRONIZATION

- Example with 2 receivers:
  - receivers record arrival of synchronization message
  - receivers exchange recorded information
  - receivers calculate offset (difference of arrival times)

- More than 2 receivers:
  - maximum phase error between all receiver pairs is expressed as group dispersion
  - likelihood that a receiver is poorly synchronized increases with the number of receivers (larger group dispersion)
  - increasing the number of broadcasts can reduce group dispersion

- Offsets between two nodes can be computed as the average phase offsets for all m packets received by receivers i and j:

$$offset[i,j] = \frac{1}{m}\sum_{k=1}^{m}(T_{j,k} - T_{i,k})$$

REFERENCE-BROADCAST SYNCHRONIZATION

- Multi-hop scenarios possible by establishing multiple reference beacons, each with its own broadcast domain

- Domains can overlap and nodes within overlapping regions serve as bridges to allow synchronization across domains

- RBS uses large amount of message exchanges

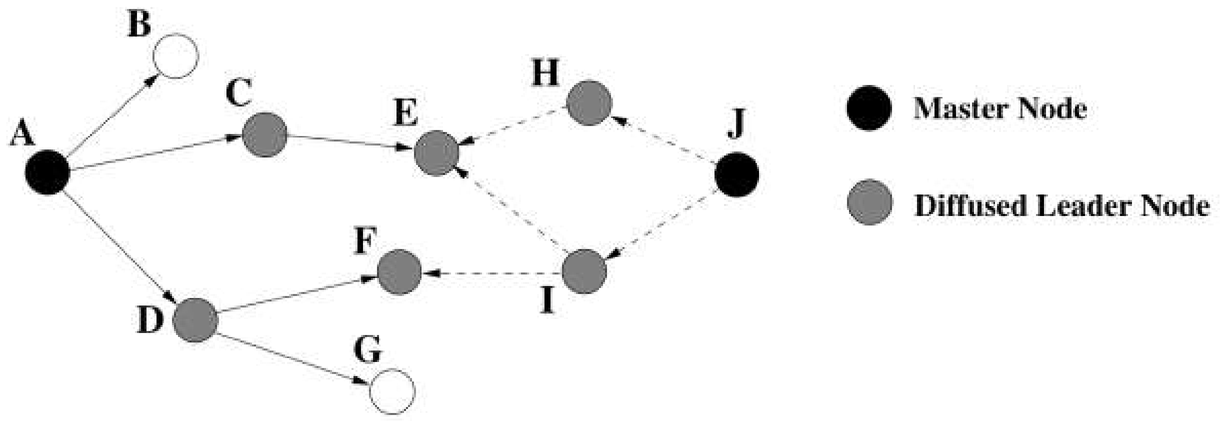
- However, RBS is a good candidate for post-facto synchronization
  - nodes synchronize after event of interest has occurred to reconcile their clocks

TIME-DIFFUSION SYNCHRONIZATION PROTOCOL

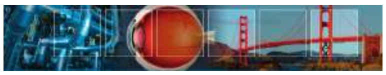
- In TDP, nodes agree on network-wide equilibrium time and maintain clocks within a small bounded deviation from this time
- Nodes structure themselves into tree-like configuration with two roles:
  - ❑ master nodes
  - ❑ diffused leader nodes
- TDP's Time Diffusion Procedure (TP) diffuses time information from master nodes to neighbors, some of which become diffused leader nodes responsible for propagating the master node's messages
- During the active phase of TDP, master nodes are elected every τ seconds using an Election/ Reelection Procedure (ERP)
  - ❑ balances workload in the network
  - ❑ τ further divided into intervals of δ seconds, each beginning with the election of diffused leader nodes
  - ❑ ERP eliminates leaf nodes and nodes with clocks that deviate from neighboring clocks by more than a certain threshold (achieved through message exchanges to compare clocks)
  - ❑ ERP also considers energy status in election process
- During the inactive phase of TDP, no time synchronization takes place

TIME-DIFFUSION SYNCHRONIZATION PROTOCOL

- Elected master node broadcasts timing information to neighbors
- Diffused leader nodes respond with ACK message
- Master nodes determine round-trip delay Δj for each neighbor j, an estimate of one-way delay for all neighbors ($\Delta avg/2$), and standard deviation of the round-trip delays
- Standard deviation is sent in another time-stamped message to each neighboring diffused leader node
- Diffused leader nodes adjust their clocks using the time-stamp, the one-way delay estimation, and the standard deviation
- Diffused leader nodes repeat process with their neighbors (n times, where n is the distance from the master node in hops)
- Nodes receiving timing information messages from multiple masters use the standard deviations as weighted ratio of their time contribution to the adjusted time



Chapter 11: Security



Chapter 11: Roadmap

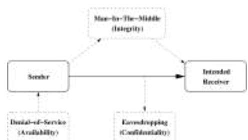
- Security fundamentals
- Security challenges
- Security attacks
- Security protocols and mechanisms
- IEEE 802.15.4 and ZigBee security

Fundamentals

- Security and privacy are big challenges for any type of computing and networking environment
- Well-known CIA security model:
  - Confidentiality
    - ensure that only the intended receiver can read/interpret a message
    - unauthorized access is prevented
  - Integrity
    - ensure that a message cannot be modified
    - unauthorized individuals should not be able to destroy/alter message
  - Availability
    - ensure that system/network is able to perform its tasks without interruption
    - often measured in terms of percentages of up/down time

Examples

- Confidentiality:
  - eavesdropping: unauthorized message reception
- Integrity:
  - man-in-the-middle attack: unauthorized individual/system positions itself between sender and receiver to intercept, modify, and retransmit messages
- Availability:
  - denial-of-service attack: attempt to disrupt transmission or service



More Terminology

- Authentication
  - process of establishing or confirming the identity of user/device
  - ensures that message came from who it claims to have come from
- Nonrepudiation
  - process of proving that person/device has performed a certain transaction/transmission
- Digital signatures
  - often used to support authentication, nonrepudiation, and integrity

Cryptography

- Process of protecting information using encoding/decoding techniques
- Symmetric key cryptography
  - single key shared between communicating parties
  - simple example: shift cipher (key = fixed shift in alphabet)
  - challenge: secure distribution of shared key
  - examples: DES, AES, IDEA
- Public key cryptography
  - secret key: will never be shared with anyone else
  - public key: can be shared freely
  - message encrypted with secret key can only be decrypted with corresponding public key (e.g., for authenticating the sender)
  - message encrypted with public key can only be decrypted with corresponding secret key (e.g., for providing confidentiality)
  - examples: RSA, Diffie-Hellman agreement protocol

Challenges of Security in WSNs

- Resource constraints
  - limited computational, networking, and storage capabilities of sensors
  - energy constraints of sensors
- Lack of central control
  - large WSNs often don't have centralized control
  - requires distributed/decentralized security solutions
- Remote location
  - sensors often left unattended
  - difficult to prevent unauthorized physical access and tampering
- Error-prone communication
  - difficult to distinguish wireless communication errors from attacks

Security in WSNs

- WSN characteristics that facilitate security:
  - self-managing and self-repairing nature
  - redundancy
- Data freshness problem
  - WSN security must ensure that sensor data are recent (and not replays of old data)
  - particularly important for key distribution schemes
- WSNs provide more opportunities for attacks than other networks
  - many sensor protocols require location information
  - many sensor nodes require accurate time synchronization
  - both can be affected by modifying, injecting, dropping messages (e.g., beacons) carrying such information

Denial-of-Service (DoS)

- Attempt to stop network/system from functioning or providing a service
- Physical Layer DoS
  - jamming attack
    - interfere with the radio frequencies of a WSN
    - even small numbers of attacking nodes can be effective if well positioned (e.g., close to an important node such as a BS) or if their signals are strong
    - countermeasure: spread-spectrum communication (e.g., FHSS)
  - tampering attack
    - attacker obtains physical access to sensor node
    - used to modify/destroy node, obtain sensitive information or use as entry points for further attacks into the network
    - countermeasures: tamper-proof materials and enclosures, disable device when attack detected

Denial-of-Service (DoS)

- Link Layer DoS
  - collision attack
    - attempt to interfere with packet transmissions
    - causes costly exponential backoff procedures and retransmissions
    - often tries to cause collisions near the end of a frame, requiring retransmission of entire frame
  - exhaustion attack
    - attacks (such as collision attack) with the goal of premature depletion of a sensor's energy sources
    - example: issue RTS message to prompt CTS response from another node (exploiting handshake techniques)

Attacks on Routing

- Blackhole attack
  - malicious node on a route simply drops all packets
- Selective forwarding attack
  - similar to blackhole attack, but not all traffic is dropped
  - more difficult to detect (hard to distinguish attack from poor connectivity)
- Rushing attack
  - exploits route discovery techniques of on-demand protocols
  - route request packets are rushed towards destination, increasing the malicious node's probability to be on the selected route
- Sinkhole attack
  - node attempts to position itself on as many network flows as possible
- Sybil attack
  - attacker claims to have multiple identities or locations
- Wormhole attack
  - out-of-band (bandwidth-rich) connection between attackers used to face short path to the gateway, attracting many flows to these nodes

Attacks on Transport Layer

- Flooding attack
  - exploits fact that many transport protocols maintain state information and are therefore vulnerable to memory exhaustion
  - example: attacker makes many (incomplete) connection requests, forcing a node to allocate more and more resources
- Desynchronization attack
  - attempt to disrupt communication between nodes by repeatedly forging messages to these nodes
  - example: fake packets carry old sequence numbers to make a node believe that its previous transmissions were not correctly received

Attacks on Data Aggregation

- Aggregation (and fusion) operations are often easily affected by an attacker
  - average function $f(x_1,\dots,x_n)=(x_1+\dots+x_n)/n$
    - replacing a single measurement x_1 with a fake reading x_1^*, the average will change from $y=f(x_1,\dots,x_n)$ to $f(x_1^*,x_2,\dots,x_n) = y+(x_1^*-x_1/n)$
    - attacker can choose x_1^* and thereby determine outcome of aggregation
  - sum function $f(x_1,\dots,x_n)=x_1+\dots+x_n$
    - replacing a single measurement x_1 with a fake reading x_1^*
  - minimum function $f(x_1,\dots,x_n)=\min(x_1,\dots,x_n)$
    - replacing a single reading does not always lead to incorrect aggregation
    - replacing x_1 with x_1^* raises minimum if x_1 is unique smallest reading of all x_i
    - replacing any xi with very small value can lower the minimum
    - similarly true for maximum function
  - count function: each sensor contributes 0 or 1 to the result
    - changing k readings changes result by at most k
    - may be negligible if k is small compared to the number of measurements

Privacy Attacks

- Attempts to obtain sensitive information collected and communicated in WSNs
- Eavesdropping
  - made easy by broadcast nature of wireless networks
- Traffic analysis
  - used to identify sensor nodes of interest (data of interest), sensor nodes that are vulnerable, and sensor nodes that are critical to the correct operation of the entire network (e.g., gateway devices, cluster heads)

Symmetric versus Public Key

- Public key cryptography
  - used to provide confidentiality, integrity, and authentication
  - computationally expensive
  - some implementations for resource-constrained devices exist (ECC, elliptic curve cryptography)
- Symmetric key cryptography
  - more resource-efficient
  - problem of key distribution

Key Management

- Reliable and secure establishment of shared cryptographic keys
- Peer Intermediaries for Key Establishment (PIKE)
  - uses sensor nodes as trusted intermediaries for key distribution
  - every sensor shares different pairwise key with each of O(√n) nodes
  - for any pair of nodes A and B, there is at least one node, C, that shares a pairwise key with both A and B
  - each sensor has an ID *(x,y)* and the network is represented as a matrix with √n rows and columns, where a node's position in the matrix it its node ID
  - then, each node *(x,y)* shares a pairwise key with each node in the following two sets:

$$(i, y) \forall i \in \{0,1,2,...,\sqrt{n}-1\}$$
$$(x, j) \forall j \in \{0,1,2,...,\sqrt{n}-1\}$$

Key Management

- PIKE (contd.)
  - example: node *(x,y)* will share key $K_{(x,y),(1,y)}$ with node *(1,y)* and another key $K_{(x,y)(2,y)}$ with node *(2,y)*
  - a node will maintain 2(√n-1) keys

00	01	02	03	04	05	...	09
10	11	12	13	14	15	...	19
20	21	22	23	24	25	...	29
30	31	31	33	34	35	...	39
.	.	.	.	.	.		.
.	.	.	.	.	.		.
.	.	.	.	.	.		.
90	91	92	93	94	95	...	99

Defenses Against DoS Attacks

- Jamming attacks
  - isolate affected region by re-routing traffic
  - use spread-spectrum techniques
- Collision and exhaustion attacks
  - error-correcting codes
  - rate-limiting schemes
- Spoofing and alteration
  - message authentication codes (MAC)
- Path-based DoS
  - attacker overwhelms nodes by flooding a multi-hop end-to-end communication path with replayed or injected packets
  - one-way hash chains can be used to validate received packets

Defenses Against Aggregation Attacks

- Delayed aggregation and delayed authentication
  - base station generates a one-way key chain using a public one-way function F, where $K_i=F(K_{i+1})$
  - each device stores key K_0 before deployment ($K_0=F^n(K)$, i.e., F applied to a secret key n times)
  - first base station transmissions are encrypted using $K_1=F^{n-1}(K)$
  - once all messages transmitted using K_1 have been received:
    - the base station reveals K_1
    - all nodes compute $F(K_1)=F(F^{n-1}(K))$ and verify that it matches $K_0=F^n(K)$
    - sensor nodes decrypt the messages

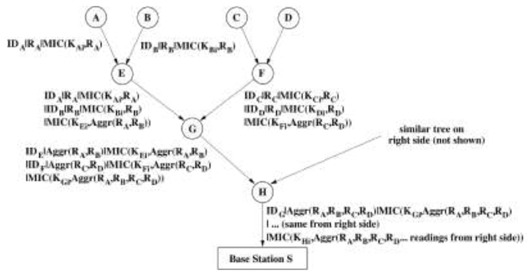
Defenses Against Aggregation Attacks

- Example:
  - nodes A-D send messages to the base station, each node's message contains the sender's ID, the sensor data, and a MAC calculated over the data using a temporary key
  - parent node cannot yet verify the MAC since it doesn't have the child's key
  - parent node stores this message and retransmits it to its own parent after certain timeout value
  - E's message to parent G contains messages received from its children (nodes A and B) and a MAC computed over the aggregate of A's and B's data using E's key
  - this process continues, i.e., every parent combines data from its children and adds its own MAC over the aggregate using its own key
  - once the base station receives messages from its children, it can compute the final aggregate value

Defenses Against Aggregation Attacks

- Example (contd.):
  - base station has shared temporary key with each sensor, therefore it can verify whether a received message was transmitted by H by calculating the MAC of the aggregation using K_{Hi} and comparing it to the MAC in the message
  - this validates that H sent the final message, but it does not validate that the message correctly reflects the readings from the other nodes
  - base station reveals the temporary keys to the network by sending each key (along with a MAC) to all sensor nodes using its own current key K_i
  - base station sends out its current key K_i such that nodes can check the MAC values and to advance to the next key in the chain

Defenses Against Aggregation Attacks



Defenses Against Routing Attacks

- Attacks from "outside" versus "inside" the network
  - link-layer encryption and authentication can prevent adversary from joining a network, preventing many possible routing attacks
  - these techniques may be ineffective if network is attacked from the inside (e.g., using a compromised node)
- Sybil attacks
  - sensor nodes can share unique symmetric key with a trusted base station to verify each other's identity
  - base station can also limit the number of neighbors a node is allowed to have (i.e., a compromised node can communicate with only a few other nodes)
- Sinkhole attacks
  - difficult to defend against where protocols are used that establish routes based on information that it difficult to verify (e.g., energy)
  - easier for routes based on minimum hop counts, but hop counts can be misrepresented through a wormhole
  - with geographic routing, it is difficult to redirect traffic elsewhere to create a sinkhole

Defenses Against Routing Attacks

- Rushing attacks
  - secure neighbor detection approach can be used to allow sender and receiver of a route request to verify that the other party is in fact within normal transmission range
  - example of a three-round mutual authentication protocol:
    - sender sends a neighbor solicitation packet
    - receiver responds with neighbor reply packet
    - sender sends a neighbor verification message (which includes broadcast authentication of a timestamp and the link from the source to the destination)

Security Protocols for Sensor Networks

- SPINS provides:
  - Secure Network Encryption Protocol (SNEP) for confidentiality, two-party data authentication, and data freshness
  - a "micro" version of the Timed, Efficient, Streaming, Loss-tolerant Authentication protocol (μTESLA) for authentication for data broadcast
  - assumption is that every node has a secret key shared with the base station

Security Protocols for Sensor Networks

- Secure Network Encryption Protocol (SNEP)
  - symmetric security (same message is encrypted differently each time)
  - replay protection
  - low communication overhead
  - uses MAC for two-party authentication and integrity

  - nodes A and B share a secret master key
  - master key used to derive four independent keys using pseudorandom function
    - two keys used for encryption of messages in each direction (K_{AB} and K_{BA})
    - two keys are used as message integrity codes (K'_{AB} and K'_{BA})

Security Protocols for Sensor Networks

- Secure Network Encryption Protocol (SNEP) (contd.)
  - complete encrypted message:
    $A \rightarrow B: \{D\}_{(KAB,CA)}, MAC(K'_{AB}C_A \| \{D\}_{(KAB,CA)})$
  - D = data, K = key, C = counter, MAC computed as $MAC(K',C \| E)$
  - provides authentication (using MAC)
  - provides replay protection (using counter value in MAC)
  - freshness (counter value enforces message ordering); considered weak since sending ordering is enforced within node B, but no absolute assurance to node A that message was created by B in response to an event in A (nonce can be added to obtain strong freshness)
  - semantic security (counter is encrypted with each message, i.e., same message will be encrypted differently)
  - low communication overhead (counter state is kept at each end point and is not sent in message)

Security Protocols for Sensor Networks

- μTESLA
  - extension of TESLA protocol (by considering resource limitations)
  - focuses on need for authenticated broadcast in WSNs
  - relies on symmetric mechanisms provided by SNEP to authenticate first packet in broadcast message
  - TESLA uses digital signatures to authenticate initial packet and has an overhead of 24 bytes per packet
  - μTESLA emulates asymmetric cryptographic mechanism through a delayed disclosure of symmetric keys
  - μTESLA assumes that base station (BS) and sensor nodes are loosely time synchronized and each sensor knows upper bound on maximum synchronization error
  - when BS sends a message, it authenticates it by computing a MAC on the packet with secret key
  - when a node receives the packet, node knows that MAC key is only known to BS
  - node stores packet until the BS broadcasts the verification key to all receivers

TinySec

- Lightweight and generic link-layer security package
- Can easily be integrated into sensor network applications
- Supports two different security options:
  - authenticated encryption (TinySec-AE)
    - data payload is encrypted
    - MAC is used to authenticate packet
  - authentication only (TinySec-Auth)
    - entire packet is authenticated with MAC
    - payload is left unencrypted
- Relies on cipher block chaining (CBC) with specially formatted 8-byte initialization vector (IV) for encryption
- Relies on efficient and fast cipher block chaining construction (CBC-MAC) for computing and verifying MACs
  - using block cipher, number of cryptographic primitives that must be implemented is minimized
  - length of MAC is 4 bytes (attacker must try at most 2^{32} blind forgeries)

Localized Encryption and Authentication Protocol

- LEAP is a key management protocol for sensor networks, designed to support in-network processing
- Key observation is that different types of packets (control versus data) have different security requirements
- LEAP provides four keying mechanisms:
  - individual keys
    - every node has unique key shared with BS
    - key used for confidentiality and MAC
  - group keys
    - globally shared key used by BS to communicate with entire network
  - cluster keys
    - shared key between sensor and its neighbors
    - used for securing local broadcast messages
  - pairwise shared keys
    - shared key between sensor and one of its immediate neighbors

Localized Encryption and Authentication Protocol

- LEAP also provides a technique for local broadcast authentication
  - every node generates a one-way key chain of certain length
  - every node transmits the first key in the chain to each neighbor (encrypted with the pairwise shared key)
  - whenever a node sends a message, it takes the next key from the chain (each key is called an AUTH key) and attaches it to message
  - keys are disclosed in reverse order of their generation and a receiver can verify the message based on the first received key or a recently disclosed AUTH key

IEEE 802.15.4

- Four basic security models:
  - access control
  - message integrity
  - message confidentiality
  - replay protection
- Security is handled by the MAC layer
- Application can choose specific security requirements by setting appropriate parameters in the radio stack (default: no security)

IEEE 802.15.4: Security Suites

Name	Description
Null	No security
AES-CTR	Encryption only, CTR mode
AES-CBC-MAC-128	128-bit MAC
AES-CBC-MAC-64	64-bit MAC
AES-CBC-MAC-32	32-bit MAC
AES-CCM-128	Encryption and 128-bit MAC
AES-CCM-64	Encryption and 64-bit MAC
AES-CCM-32	Encryption and 32-bit MAC

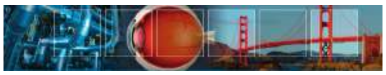
ZigBee Security

- Introduces the concept of trust center (responsibility assumed by the ZigBee coordinator)
  - responsible for authentication of devices wishing to join network (trust manager)
  - responsible for maintaining and distributing keys (network manager)
  - responsible for enabling end-to-end security (configuration manager)
- Residential mode
  - trust center allows nodes to join network, but does not establish keys with the network devices
- Commercial mode
  - trust center generates and maintains keys and freshness counters with every device in the network
  - large memory cost

ZigBee Security

- ZigBee uses the CCM* mode for security, which is a combination of CTR mode and CBC-MAC mode
- Compared to CCM, CCM* offers encryption-only and integrity-only capabilities
- ZigBee has several levels of security, including:
  - no security
  - encryption only
  - authentication only
  - encryption and authentication
- ZigBee's MAC can vary from 4 to 16 bytes

Chapter 12: Sensor Network Programming



Chapter 12: Roadmap

- Challenges
- Node-centric programming
- Macroprogramming
- Dynamic reprogramming
- Sensor network simulators

Challenges in WSN Programming

- Reliability
  - resilience to changes and failures is important in WSNs
  - should be supposed by a programming environment
- Resource constraints
  - resource limitations of WSNs affect maximum code size, performance, memory/storage capacities
  - programming environment should allow programmer to exploit energy-saving techniques
- Scalability
  - sensor networks can be very large, i.e., programming models must scale
  - manual configuration, maintenance, repair may be infeasible
- Data-centric networks
  - focus is on the data, not the devices

Node-Centric Programming

- Programming abstractions, languages, and tools focus on development of SW on a per-node basis
- Overall network-wide sensing task is then a collection of pairwise interactions of sensor nodes

nesC Programming Language

- TinyOS and nesC have become the de facto standard in WSN programming
- nesC is an extension to C programming language
- Provides set of language construct to implement SW for sensors

- nesC applications consist of a collection of components and each component has "provides" and "uses" interfaces
  - interface: describes the use of some kind of service
  - provides: set of method calls that are exposed to higher layers
  - uses: set of method calls that hide details of lower-layer components

Fundamentals of Wireless Sensor Networks: Theory and Practice
Waltenegus Dargie and Christian Poellabauer © 2010 John Wiley & Sons Ltd.

nesC Programming Language

```
module TimerModule {
    provides {
          interface StdControl;
          interface Timer;
    }
    uses interface Clock as Clk;
}

interface StdControl {
    command result_t init ();
}

interface Timer {
    command result_t start (char type, uint32_t interval);
    command result_t stop ();
    event result_t fired ();
}
```

Fundamentals of Wireless Sensor Networks: Theory and Practice
Waltenegus Dargie and Christian Poellabauer © 2010 John Wiley & Sons Ltd.

nesC Programming Language

```
interface Clock {
    command result_t setRate (char interval, char scale);
    event result_t fire ();
}

interface Send {
    command result_t send (TOS_Msg *msg, uint16_t length);
    event result_t sendDone (TOS_Msg *msg, result_t success);
}

interface ADC {
    command result_t getData ();
    event result_t dataReady (uint16_t data);
}
```


nesC Programming Language

- Example: timer interface:
  - two types of commands (functions): start, stop
    - commands are implemented by providers of an interface
  - event (function)
    - events are implemented by the users
- Components have an implementation
  - modules are components implemented by application code
  - configurations are components implemented by connecting interfaces of existing components
  - every nesC application has a top-level configuration
    - describes how components are "wired together"
  - functions are described as *f.i* (*f* is a function in an interface *i*)
  - functions are invoked using the call operation (for commands) and the signal operation (for events)

nesC Programming Language

```
module PeriodicSampling {
    provides interface StdControl;
    uses interface ADC;
    uses interface Timer;
    uses interface Send;
}
```

nesC Programming Language

```
implementation P
    uint16_t sensorReading;

    command result_t StdControl.init () {
            return call Timer.start (TIMER_REPEAT, 1000);
    }
    event result_ Timer.fired () {
            call ADC.getData ();
            return SUCCESS;
    }
    event result_t ADC.dataReady (uint16_t data) {
            sensorReading = data;
            …
            return SUCCESS;
    }
    …
}
```


nesC Programming Language

- StdControl.init is called at boot time
  - creates a timer that expires every 1000ms
  - upon time expiration, a new sensor sample is obained
  - ADC.getData triggers actual sensor data acquisition (ADC.dataReady)

nesC Programming Language

- Example of wiring subcomponents: timer service in TinyOS (TimerC)

```
configuration TimerC {
        provides {
                interface StdControl;
                interface Timer;
                }
}
implementation {
        components TimerModule, HWClock;

        StdControl = TimerModule.StdControl;
        Timer = TimerModule.Timer;

        TimerModule.Clk -> HWClock.Clock;
}
```

nesC Programming Language

- In TinyOS, code executes either asynchronously (in response to interrupt) or synchronously (as a scheduled task)
- Asynchronous code (AC): nesC code that is reachable from at least one interrupt handler
- Synchronous code (SC): nesC code that is reachable only from tasks
  - always atomic to other synchronous codes (tasks are always executed sequentially and without preemption)
- Race conditions occur when concurrent updates to shared state are performed
  - shared state is modified from AC or
  - shared state is modified from SC that is also modified from AC

nesC Programming Language

- nesC provides two options to ensure atomicity
  - convert all sharing code to tasks (SC only)
  - use atomic sections to modify shared state

```
…
event result_t Timer.fired () {
   bool localBusy;
   atomic {
           localBusy = busy;
           busy = TRUE;
   }
   …
}
…
```

- Nonpreemption can be obtained by disabling interrupts during atomic section (no call/signal allowed to ensure that atomic sections are brief)

TinyGALS

- Globally Asynchronous and Locally Synchronous (GALS)
- TinyGALS consists of modules composed of components
- A component C has
  - set of internal variables V_C
  - set of external variables X_C
  - set of methods I_C (that operate on these variables)
- Methods are further divided:
  - calls in the $ACCEPTS_C$ set (can be called by other components)
  - calls in the $USES_C$ set (needed by C and may belong to other components)

TinyGALS

- TinyGALS defines components using an interface definition and implementation (similar to nesC)

```
COMPONENT DownSample
ACCEPTS {
    void init (void);
    void fire (int in);
};
USES {
    void fireOut (int out);
};
```


TinyGALS

- Implementation for the *DownSample* component (*_active* is an internal boolean variable that ensures that for every other *fire()* method, the component will call *fireOut()* with the same integer argument)

```
void init () {
    _active = true;
}
void fire (int in) {
    if (_active) {
            CALL_COMMAND (fireOut) (in);
            _active = false;
    } else {
            _active = true;
    }
}
```


TinyGALS

- TinyGALS modules consist of components
- Module M is a 6-tuple

$M=(COMPONENTS_M, INIT_M, INPORTS_M, OUTPORTS_M, PARAMETERS_M, LINKS_M)$

$COMPONENTS_M$ … set of components of M

$INIT_M$ … list of methods of M's components

$INPORTS_M$ … inputs of the module

$OUTPORTS_M$ … outputs of the module

$PARAMETERS_M$ … set of variables external to the components

$LINKS_M$ … relationship between the method call interfaces and the inputs and outputs of the module

TinyGALS

- Modules are connected to each other to form complete TinyGALS system, where a system is a 5-tuple

  $S=(MODULES_S, GLOBALS_S, VAR\_MAPS_S, CONNECTIONS_S, START_S)$

  - $MODULES_S$ … set of modules
  - $GLOBALS_S$ … global variables
  - VAR_MAPS_S … set of mappings (map global variable to a parameter of a module)
  - $CONNECTIONS_S$ … list of connections between module output and input ports
  - $START_S$ … name of an input port of exactly one module (starting point for execution)

TinyGALS

- Highly structured architecture of TinyGALS can be exploited to automate the generation of scheduling and event handling code
  - frees developers from writing error-prone concurrency control code
- Code generation tools can automatically produce:
  - all necessary code for component links and module connections
  - code system initialization
  - code for start of execution
  - code for intermodule communication
  - code for global variables reads and writes
- Modules use message passing and are therefore decoupled from each other (easier independent development)
  - each message triggers scheduler and activate receiving module
  - TinyGUYS (Guarded Yet Synchronous) variables:
    - modules read global variables without delay (synchronously)
    - modules write global variables using buffer (asynchronously)
      - buffer size is 1 (i.e., last writing module wins)

Sensor Network Application Construction Kit

- SNACK consists of a configuration language, component and service library, and compiler
- Goals are:
  - to provide smart libraries that can be combined to form WSN applications
  - to simplify the development process
  - to be efficient
- It should be possible to write simple pieces of code such as:

```
SenseTemp -> [collect] RoutingTree;
SenseLight -> [collect] RoutingTree;
```

Sensor Network Application Construction Kit

- Syntax of SNACK code:

```
service Service {
   src :: MsgSrc;
   src [send:MsgRcv] -> filter :: MsgFilter -> [send] Network;
   in [send:MsgRcv] -> filter;
}
```

- *n::T* declares an instance named *n* of a component type *T* (i.e., an instance is an object of a given type)
- *n[i:τ]* indicates an output interface on component *n* with name *i* and interface type *τ* (similarly, *[i:τ]n* is an input interface)
- A component *provides* its input interfaces and *uses* its output interfaces

Sensor Network Application Construction Kit

- SNACK library
  - variety of components for sensing, aggregation, transmission, routing, and data processing
  - several core components supported
    - *Network*: receives/sends messages from/to TinyOS radio stack
    - *MsgSink*: ends inbound call chains and destroys received buffers
    - *MsgSrc*: generates periodic empty SNACK messages and passes them to outbound interface
    - *Timing*:
      - *TimeSrc*: generates a timestamp signal sent over signal interface at specified minimum rate
      - *TimeSink*: consumes such signals
    - *Storage*: implemented by components such as Node-Store64M, which implements an associative array of eight-byte values keyed by node ID
    - *Service*: variety of services (e.g., *RoutingTree* implements a tree designed to send data up to some root)

Thread-Based Model

- Multiple tasks allowed to make progress in execution without concern that a task may block other tasks (or be blocked) indefinitely
- Task scheduler manages task execution
  - example: time-slicing approach, where tasks execute for certain amount of time
- MANTIS (MultimodAl system for NeTworks of In-situ wireless Sensors)
  - thread-based operating system for WSNs
  - memory-efficient
    - requires less than 500 bytes of RAM
    - 14 kbytes of flash memory
  - energy-efficiency
    - microcontroller switches to low-power sleep state after all active threads have called the *sleep()* function

Thread-Based Model

- TinyThread
  - adds support for multithreaded programming to TinyOS and nesC
  - procedural programming of sensors
  - includes suite of interfaces that provide blocking I/O operations and synchronization primitives
- Protothreads
  - lightweight stackless type of threads; all protothreads share the same stack
  - context switch is done by stack rewinding
  - variables must be saved before calling a blocking wait (variables with function-local scope that are automatically allocated on stack are not saved across wait calls)
- Y-Threads
  - preemptive multithreading (distinguish preemptable from nonpreemptable code)
  - shared stack for nonblocking parts, thread-specific stack for blocking calls
    - blocking portions of code require only small amounts of stack, leading to better memory utilization compared to other preemptive approaches

Macroprogramming

- Focus on programming group of nodes, instead of individual nodes
- Abstract Regions
  - focuses on group-level cooperation
    - group of nodes working together to sample, process, and communicate sensor data
  - region-based collective communication interface
  - defines neighborhood relationship between nodes
    - *"the set of nodes within distance d"*
  - type of definition of abstract region depends on the type of application
  - examples of implementations of abstract regions
    - N-radio hop (nodes within N radio hops)
    - k-nearest neighbor (k nearest nodes within N radio hops)
    - spanning tree (rooted at a single node, used for data aggregation over entire network)

Macroprogramming

- Abstract Region (contd.)
  - example:
    - regions defined using hop distances
    - discovery of region members using periodic broadcasts (advertisements)
    - data can be shared between region members using a "push" (broadcasting) or "pull" (issue a fetch message) approach

Macroprogramming

- EnviroTrack
  - object-based middleware library
  - geared toward target-tracking sensor applications
  - free developer from details of
    - interobject communication
    - object mobility
    - maintenance of tracking objects and their state
  - also uses the concept of groups, which are formed by sensors which detect certain user-defined entities in the physical environment
  - groups are identified by context labels (logical addresses that follow the external tracked entity around in the physical environment)

Macroprogramming

- EnviroTrack (contd.)
  - tracking objects: objects can be attached to context labels to perform context-specific operations; executed on the sensor group of the context label
  - type of context label depends on entity (e.g., context label *car* is created wherever a car is tracked)
  - context label of some type *e*:
    - function $sense_e()$: describes sensory signature identifying tracked environmental target (car: magnetometer and motion sensor readings)
      - also used to track group membership
    - aggregation function $state_e()$: environmental state shared by all objects attached to a context label
      - acts on the readings of all sensors for which $sense_e()$ is true
    - aggregation is performed by sensor node acting as group leader

Macroprogramming

- Database approaches treat entire WSN as a distributed database that can be queried
- TinyDB
  - network is represented logically as a table (called *sensors*)
    - one row per node per instant in time
    - each column corresponds to sensor readings (light, temperature, pressure, …)
    - new record is added when a sensor is queried
    - new information is usually stored for a short period of time only
  - queries are like SQL-based queries (SELECT, FROM, WHERE, etc.)

```
SELECT nodeid, light, temp
   FROM sensors
   SAMPLE PERIOD 1s FOR 10s
```

    - initiates data collection at beginning of each epoch (specified in SAMPLE PERIOD clause); results are streamed to the root of the network

Macroprogramming

- TinyDB (contd.)
  - supports group aggregation queries
  - example: monitoring microphone-equipped rooms; look for rooms where the average volume is over a certain threshold:

```
SELECT AVG(volume), room FROM sensors
  WHERE floor = 6
  GROUP BY room
  HAVING AVG(volume) > threshold
  SAMPLE PERIOD 30s
```

- Similar projects
  - Cougar: resource-efficient database appraoch
  - SINA: models WSN as collection of distributed objects; supports SQTL scripts in SQL queries
  - MiLAN: sensor applications can specify QoS needs

Dynamic Reprogramming

- Sometimes necessary to disseminate code to all or some sensor nodes
- Virtual machines
  - Maté
    - small VM on top of TinyOS
    - capsules (sequence of 24 instructions) inside a single TinyOS packet
    - every capsule includes type and version information
      - *message send* capsules
      - *message receive* capsules
      - *timer* capsules
      - *subroutine* capsules
    - programs execute in response to events (e.g., timer firing, packet being sent/received)
    - each event has a capsule and an execution context

Dynamic Reprogramming

- Virtual machines (contd.)
  - Maté (contd.)
    - jumps to first instruction of capsule and executes until **halt** instruction
    - when subroutine called, return address is pushed onto a return address stack; upon return, address is taken from the stack
- Trickle
  - controlled flooding protocol for disseminating small pieces of code
  - uses metadata to describe code (allows node to determine if code update needed)
  - metadata is exchanged among neighbors via broadcast
    - periodic time intervals, each node randomly selects broadcast time during each interval
  - when a node hears outdated metadata, it broadcasts its own code, giving outdated node chance to update
  - when a node overhears newer metadata, it broadcasts its own metadata, triggering the neighbor to broadcast newer code

Dynamic Reprogramming

- Melete
  - similar to Maté and Trickle
  - supports multiple concurrent applications
  - supports selective dissemination by limiting dissemination range
    - code is only forwarded within a forwarding region
- Deluge
  - occasionally advertises the most recent code version using broadcasts
  - if a node receives an update with old code, it responds with new code version (allowing neighbor to request new code)
  - eliminates redundant advertisements and request messages
  - provides robustness
    - uses a three-phase handshake to ensure that only symmetric links are used
    - allowing a node to search for a new neighbor to request code if it has not completely received the code after *k* requests
  - dynamically adjusts rate of advertisements for quick propagation when needed, but consuming few resources in steady state

Dynamic Reprogramming

- Pump Slowly, Fetch Quickly (PSFQ)
  - slowly pace propagation of packets (pump slowly)
  - aggressively fetch lost packets (fetch quickly)
  - nodes do not relay packets out of order
  - prevents loss events from propagating downstream
  - *localized recovery* allows nodes to recover lost packets from immediate neighbors (reduces recovery costs)
- Push Aggressively with Lazy Error Recovery (PALER)
  - based on observation that pushing data downstream and recovering lost packets simultaneously leads to excessive contention
  - eliminates in-order reception requirement
  - pushes all data aggressively
  - nodes keep list of missing packets and request retransmission after the broadcast period
  - retransmission requests are handled by neighbors (if they don't have a copy of missing data, they issue their own request to their neighbors)

Sensor Network Simulators

- Large scale of sensor networks makes implementation and experimentation difficult and expensive
- Instead, simulation is used to evaluate novel WSN tools, mechanisms, protocols, and applications
- Quality of simulations depends on choice of appropriate models for
  - sensor node hardware/software characteristics
  - wireless communication
  - physical environment
  - node mobility
- Simulators typically come with tools for collecting, analyzing, and visualizing sensor data

ns-2

- Discrete event simulator called network simulator (ns-2)
- Written in C++ and Otcl
- Highly extensible, many extensions have been developed, e.g.:
  - extension adding the concept of a phenomenon (physical event)
    - uses broadcast packets over designated channel to represent physical phenomena (fire, moving vehicle, chemical cloud)
    - uses PHENOM routing protocol: emits packets with certain configurable pulse rate and whose arrival triggers a receive event
  - many routing protocols
  - many MAC-layer protocols
  - variations of packet contents
  - models for multi-homed devices
  - mobility models

GloMoSim and QualNet

- GloMoSim
  - based on the PARSEC (PARallel Simulation Environment for Complex systems) simulation environment
    - C-based simulation language
    - represents sets of objects in physical environment as logical processes
    - represents interactions among these objects as time-stamped message exchanges
  - supports variety of models at different protocol layers
  - supports different mobility models
  - intended for academic use
- QualNet
  - commercial version of GloMoSim
  - produced by Scalable Network Technologies, Inc.

JiST/SWANS

- Java in Simulation Time (JiST)
  - discrete event simulator
  - efficient
    - run program in parallel
    - dynamically optimize simulator configuration
  - transparent
    - transform simulation programs automatically to run with simulation time semantics (instrument simulations such that no programmer intervention or calls to specialized libraries are needed to support concurrency, consistency, reconfiguration, etc.)
- Scalable Wireless Ad hoc Network Simulator (SWANS)
  - built on top of JiST
  - collection of independent SW components that can be aggregated to form complete wireless (ad hoc) simulations

OMNeT++

- Objective Modular Network Testbed
  - discrete event simulator for simulating communication networks, multiprocessors, and distributed systems
  - open-source based on C++
  - models consist of modules that communicate using message passing
    - simple modules and compound modules
  - uses topology description language NED to define structure of a module
  - includes graphical editor
  - lacks protocol models

TOSSIM

- Simulator for TinyOS-based networks
- Generates discrete event simulations directly from TinyOS components (i.e., runs the same code that runs on the sensors)
- Replaces low-level components (e.g., interrupts) with events in the simulations
- Simulator event queue delivers these events to components
- Works at bit level, i.e., event is generated for each sent or transmitted bit
  - allows for experimentation with low-level protocols
- TinyViz: visualization tool
- Very scalable and extensible
- Lacks energy profiling and use is limited to TinyOS systems

EmStar

- Targeted at high capability nodes called microservers (e.g., cluster heads)
- Consists of a Linux microkernel extension, libraries, and several tools
- EmSim: operates many virtual nodes in parallel in a simulation that models radio and sensor channels
- EmCee: runs the EmSim core and is an interface to real low-power radios
- EmView: graphic visualizer

Avrora

- Flexible simulator framework in Java
- Each node is its own thread and code is executed instruction-by-instruction
- Event queue:
  - targets nodes operating in long sleep modes
  - event queue takes advantage of that to boost performance
  - when node sleeps, only a time-triggered event that causes an interrupt can wake up the node
  - such an event is inserted into event queue of the node to be woken up at a certain time
  - simulator processes events in order until one of them triggers a hardware interrupt, which re-awakes a node
- Fast and scalable simulator; can simulate down to the level of individual clock cycles