LoRa: Long Range Communication System for IoT
LoRa IoT Networks

- **LoRa Component**
  - Composed of end devices, gateways, and a network server
    - End devices are connected with one or more gateways through one-hop LoRa link
    - Gateways forward all successfully decoded messages from end devices to their associated network server through standard IP link

- **LoRa Topology**
  - Star-of-Stars

![LoRa IoT Networks Diagram]
Classification of Long Range IoT Applications

- **Traffic pattern**
  - Uplink
    - A control station collects the data produced by the end devices
    - Smart metering, monitoring applications
  - Uplink & Downlink
    - A control station sends the command to the end devices according to the collected data
    - Environmental control

- **Energy constraints**
  - With strict energy constraints
    - Most end devices have strict energy constraints
  - Without strict energy constraints
    - Some end devices connected to the power have less strict energy constraints
Short Summary of LoRa Features

- **Low bandwidth on sub-GHz band**
  - Regional ISM bands
    - 863-870 MHz (EU), 902-928 MHz (US), 917-923.5 MHz (KR, SKT)
    - Bandwidth: 125 kHz (250 kHz, 500 kHz, 150 kHz)

- **Low data rate**
  - Range from 0.3 kbps to 50 kbps

- **Long range**
  - Rural: 10-15 km
  - Urban: 3-5 km

- **Long Life Time**
  - < 10 years

- **Low Cost**
  - < 1 USD
LoRa End Devices

❖ Three Classes

- **Class A (All)**
  - Default functions which should be supported by all LoRa devices
  - Asynchronous uplink transmission (transmission at any time according to pure ALOHA), followed by **two downlink receive windows**
  - Intended for monitoring applications

- **Class B (Beacon)**
  - End devices synchronize with the server by using beacon packets
  - Open **extra receive windows** at scheduled times
  - Intended for control applications

- **Class C (Continuously listening)**
  - End devices **continuously open receive windows** except for time when transmitting
  - Intended for devices without strict energy constraints
Standard Protocol Stack

Application

LoRa MAC
- Class A (baseline)
- Class B (beacon)
- Class C (Continuous)

LoRa Modulation
- EU 868
- EU 433
- US 915
- AS 430
- ...

Regional ISM band

MAC
MAC options
Modulation
Any gateway that receives an uplink message relays it to network server. => multiple copies of an uplink message can be delivered to the server. A downlink message is sent to an end device by only one gateway.
Receive Windows (1)

Class A Receive Windows

First receive window (RX1)
- Opens RECEIVE_DELAY1 after the end of uplink modulation
- Use same frequency channel as the uplink channel
- With the same data rate as last uplink data rate

Second receive window (RX2)
- Opens RECEIVE_DELAY2 after the end of uplink modulation
- Use fixed configurable frequency channel
- Does not open when the frame for the end device is detected and demodulated during the first receive window
• If a preamble is detected during one of the receive windows, the radio receiver stays active until the downlink frame is demodulated.

• If a frame was demodulated during the first receive window and the frame was intended for this end-device after address and MIC checks, the end-device does not open the second receive window.
LoRa Channels

- **Regional ISM Band**
  - **In Korea**
    - 8 default channels with 125 kHz bandwidth
    - 1 channel for RX2 and 7 channels for RX1

<table>
<thead>
<tr>
<th>Channel ID</th>
<th>Channel Freq</th>
<th>RX1/RX2</th>
</tr>
</thead>
<tbody>
<tr>
<td>25</td>
<td>921.9 MHz</td>
<td>RX2 (DR0, SF12)</td>
</tr>
<tr>
<td>26</td>
<td>922.1 MHz</td>
<td></td>
</tr>
<tr>
<td>27</td>
<td>922.3 MHz</td>
<td></td>
</tr>
<tr>
<td>28</td>
<td>922.5 MHz</td>
<td></td>
</tr>
<tr>
<td>29</td>
<td>922.7 MHz</td>
<td>RX1</td>
</tr>
<tr>
<td>30</td>
<td>922.9 MHz</td>
<td></td>
</tr>
<tr>
<td>31</td>
<td>923.1 MHz</td>
<td></td>
</tr>
<tr>
<td>32</td>
<td>923.3 MHz</td>
<td></td>
</tr>
</tbody>
</table>

- **In Europe**
  - 3 default channels with 125 KHz : 868.1 MHz, 868.3 MHz, 868.5 MHz
  - At maximum, 5 channels can be added at association
LoRa Data Rates

- Achievable bit rate for each data rate by using different spreading factors (SF) and LoRa modulation on 125 KHz bandwidth

<table>
<thead>
<tr>
<th>Data rate</th>
<th>SF</th>
<th>Physical bit rate (bps)</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>12</td>
<td>250</td>
</tr>
<tr>
<td>1</td>
<td>11</td>
<td>440</td>
</tr>
<tr>
<td>2</td>
<td>10</td>
<td>980</td>
</tr>
<tr>
<td>3</td>
<td>9</td>
<td>1760</td>
</tr>
<tr>
<td>4</td>
<td>8</td>
<td>3125</td>
</tr>
<tr>
<td>5</td>
<td>7</td>
<td>5470</td>
</tr>
</tbody>
</table>

- Rate 6: 11 kbps by using SF 7 and Lora modulation on 250 KHz bandwidth
- Rate 7: 50 kbps using GFSK modulation on 150 KHz bandwidth
LoRa Modulation

LoRa uses **chirp spread spectrum (CSS)** for modulation

- Spread the spectrum by generating a chirp signal
- Chirp signal: frequency increases or decreases over time
- Length of the spreading code is $2^{SF}$ ($SF : Spreading Factor$)

![Up-Chirp Diagram](image)
Example Modulation techniques:

- **On-Off-Keying (OOK)**
  - Up-Chirp = “1”
  - Null = “0”

- **Superposed Chirps (4 possible states)**
  - Null/Up-Chirp/Down-Chirp/Superposition of Up-Chirp and Down-Chirp
  - allows one network double the data rate

- LoRa uses different modulation technique
LoRa modulation using CSS (1)

One symbol: $2^{SF}$ chips
A chip duration (fixed): $1/BW$

SF bits are encoded by a symbol
Data Rate: $BW \times SF / 2^{SF}$
Baseband signal bandwidth: $BW \times SF / 2^{SF}$
CSS signal bandwidth: $BW$

Symbol time: $2^{SF} / BW$

Spectrum spreading: $2^{SF} / SF$
LoRa modulation using CSS (2)

Example: SF=7, BW=125 kHz

- one symbol: 128 chips = 128 x 8 $\mu$s = 1,024 $\mu$s, encoding seven bits

<table>
<thead>
<tr>
<th>Symbol</th>
<th>0</th>
<th>64</th>
<th>32</th>
<th>95</th>
</tr>
</thead>
<tbody>
<tr>
<td>$f_{\text{high}}$</td>
<td>0</td>
<td>64</td>
<td>32</td>
<td>95</td>
</tr>
<tr>
<td>$f_{\text{center}}$</td>
<td>0</td>
<td>64</td>
<td>32</td>
<td>95</td>
</tr>
<tr>
<td>$f_{\text{low}}$</td>
<td>0</td>
<td>64</td>
<td>32</td>
<td>95</td>
</tr>
</tbody>
</table>

Bit stream: 0000000100000001000001011111

0 64 32 95

1.024 ms
Variable Data Rate by adjusting SF

125kHz bandwidth channel:
• one chip: 8 μs
• a symbol: $8 \times 2^{\text{SF}}$ => (SF 7) 1.024 ms, (SF 8) 2.048 ms, … , (SF 12) 32.768 ms
**Message Types (1)**

- **PHY Message**
  - **Format**
    - **Preamble**: used to synchronize the receiver with the incoming data flow
    - **PHY header**: payload length, FEC coding rate of payload
    - **Payload CRC**: exists only for the uplink
  - **Preamble**, **header CRC**, **payload CRC**: inserted by radio transceiver
Message Types (2)

MAC Message (PHY payload)

Format

- MHDR: header which specifies the message type
- MACPayload: data frame with header and Fport
  - FHDR: device address, ADR control bits, ACK bit, Pending bit, ...
  - Fport: 0 (MAC Command), application-specific (destination)
- MIC: Encrypted version of message (for security), based on AES-128

Type

- Join request: Request from an end node, for joining to the network
- Join accept: Acceptance that end node can join to the network
- Rejoin request
- Unconfirmed data up/down: Data transmission without acknowledgment
- Confirmed data up/down: Data transmission with acknowledgment
LoRa End-Device Activation

- To participate in a LoRaWAN network, each end device has to obtain
  - A unique id of the device within a current network (a device address)
  - Some keys for security (eg., key for calculating and verifying MIC)

- End device sends *join-request* message to a server, and server replies
  a *join-accept* message if the join request is accepted

![Diagram of LoRa End-Device Activation](image-url)
An end device transmits frames by using pure ALOHA protocol

- **Confirmed Data Up**
  - When the network receives the frame, the network generates a downlink frame with the ACK bit set RECEIVE_DELAY1 seconds later
  - ACK frame can also contain data or MAC commands
LoRa Class-A Procedure (2)

- **Confirmed Data Up (cont.)**
  - If an end device does not receive an ACK frame in one of two receive windows, the end device may resend the same frame ACK_TIMEOUT seconds after the second receive window.
  - Retransmission must be done on another channel.

![Diagram of LoRa Class-A Procedure](image)
LoRa Class-A Procedure (3)

- **Data Rate Adaptation**
  - Data rate can be adjusted to ensure reliable packet delivery and optimal network performance
  - Rate adaptation is allowed (individually for each device) when ADR bit (within FHDR) is set
  - ADR bit can be set by an end device or network server

  - Server can adjust the data rate of each end device
    - Server sends a `LinkADRReq` message to the end device
      - Contains requested data rate, TX output power, and usable channels
    - End device answers to server with a `LinkADRAns` message
  - Uplink rate control by an end device (next two slides)
LoRa Class-A Procedure (4)

- Rate Adaptation for Confirmed Up Transmission

- Every two confirmed up transmission fails, end device lowers the data rate and try again
- If frame has not been acknowledged after 8 transmissions, end device re-initiates the transmission a little later

<table>
<thead>
<tr>
<th>Transmission nb</th>
<th>Data Rate</th>
</tr>
</thead>
<tbody>
<tr>
<td>1 (first)</td>
<td>DR</td>
</tr>
<tr>
<td>2</td>
<td>DR</td>
</tr>
<tr>
<td>3</td>
<td>max(DR-1,0)</td>
</tr>
<tr>
<td>4</td>
<td>max(DR-1,0)</td>
</tr>
<tr>
<td>5</td>
<td>max(DR-2,0)</td>
</tr>
<tr>
<td>6</td>
<td>max(DR-2,0)</td>
</tr>
<tr>
<td>7</td>
<td>max(DR-3,0)</td>
</tr>
<tr>
<td>8</td>
<td>max(DR-3,0)</td>
</tr>
</tbody>
</table>
Rate Adaptation for Unconfirmed UP Transmission

1. If the data rate of end device is higher than its default data rate, the end device needs to validate the network still receives the uplink frame.

2. Validation procedure:
   - For each new uplink frame, the device increments ADR_ACK_CNT counter.
   - If any downlink frame is received, end device resets ADR_ACK_CNT.
   - After ADR_ACK_LIMIT uplink frames without any downlink response, the end device sets the ADRACKReq bit (within FHDR).
   - The network is required to respond with a downlink frame within the next ADR_ACK_DELAY uplink frames.
   - If no reply is received within next ADR_ACK_DELAY uplink frames, the end device switches to next lower data rate.
LoRa Class-A Procedure (6)

- **Confirmed Data Down**
  - Server sends downlink data frame after receiving frame from end device
  - Uplink ACK is transmitted like any standard uplink frame
  - Channel may be different
LoRa Class-A Procedure (7)

**Frame Pending**

- Server sets FPending bit (within FHDR) to inform the end device that the network server has several frames pending for the ED
- Then, the end device opens another receive window as soon as possible by sending another empty uplink message
Frame Pending (cont.)

- FPending bit, the ACK bit, and payload data can all be present in the same downlink message.
- If end device does not receive data from one of two receive windows, end device sends empty uplink message without ACK bit set after ACK_TIMEOUT
LoRa Class-A Procedure (9)

- **Duty Cycle Control**
  - The network limits the maximum transmit duty cycle over all channels of an end device.
  - Server sets the maximum aggregated transmit duty cycle by sending DutyCycleReq message:
    - Contains MaxDCycle value which is between 0 and 15
    - Maximum end device transmit duty cycle:
      \[
      \text{aggregated duty cycle} = \frac{1}{2^{\text{MaxDCycle}}}
      \]
  - End device answers to server with a DutyCycleAns message.
LoRa Class-A Procedure (10)

Channel Control

- Server can create a new channel or modify the current existing channel
  - Device is able to handle at least 16 different channels
  - Default channels cannot be modified

- Server sends *NewChannelReq* message to the end device
  - Contains channel frequency and allowed data rate range information
  - Channel frequency can be set from 100 MHz to 1.67 GHz in 100 Hz steps

- End device answers to server with *NewChannelAns* message
LoRa Class-B Procedure (1)

- End devices open additional receive windows at fixed time intervals
  - for the purpose of enabling server initiated downlink message
- End device in class B
  - should start and join the network as an end device of class A
  - can switch to class B
- Gateway sends a beacon on a regular basis to synchronize the all end devices in the network
- End device application can decide to switch to class B
  - The end-device application requests the LoRaWAN layer to switch to Class B mode.
  - The LoRaWAN layer in the end-device searches for a beacon
  - Based on the beacon strength and the battery life constraints, the end-device application selects a ping slot data rate and periodicity
- If end device in class B does not receive a beacon for a given period, the end device switches back to class A
LoRa Class-B Procedure (2)

- **Beacon Timing**

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Duration</th>
</tr>
</thead>
<tbody>
<tr>
<td>Beacon_period</td>
<td>128 s</td>
</tr>
<tr>
<td>Beacon_reserved</td>
<td>2.120 s</td>
</tr>
<tr>
<td>Beacon_guard</td>
<td>3.000 s</td>
</tr>
<tr>
<td>Beacon-window</td>
<td>122.880 s</td>
</tr>
</tbody>
</table>

- The beacon window interval is divided into 4096 ping slots of 30 ms each numbered from 0 to 4095.
LoRa Class-B Procedure (3)

- **Beacon Reception Slots and Ping Slots**
  - In Class B, all gateways synchronously broadcast a beacon every `BEACON_PERIOD` seconds.
  - Each end device opens additional receive windows called “ping slots” every `PING_PERIOD` seconds.
LoRa Class-B Procedure (4)

- **Switch from Class A to Class B**
  - End device sends *PingSlotInfoReq* message to the server when it wants to switch to Class B
    - Contains ping slot period and expected data rate of ping
      \[ pingSlotPeriod = 2^{Periodicity} \quad (Periodicity = [0...7]) \]
  - Server sends *PingSlotInfoAns* to the end device.
  - Server sets the unicast ping channel of an end-device by sensing *PingSlotChannelReq* message and the end device acknowledges with *PingSlotFreqAns* message.
  - At each beacon period, the end-device and the server compute a new pseudo-random offset (ping offset) to align the ping slots
    - Key = 16 x 0x00
    - \( Rand = aes128\_encrypt(\text{Key}, \text{beaconTime} \mid \text{DevAddr} \mid \text{pad16}) \)
    - \( pingOffset = (Rand[0] + Rand[1] \times 256) \mod \text{pingSlotPeriod} \)
LoRa Class-C Procedure

- Class-C end device will continually listen with RX2 windows when it is not either sending or receiving on RX1
- End device can receive a downlink at nearly any time
Uplink signal from an end device can be received at multiple GWs which are time-synchronized with each other.

**Triangulation** estimates the position of a target object (ED) by measuring a distance between the ED and several reference points (GWs)

- Time of arrival (ToA)
- Time difference of arrival (TDoA)

**ToA**
- Calculates the distance between the ED and GW by measuring the one-way propagation time
- Is applied, when the ED is also time-synchronized with GWs

**TDoA**
- Determines the relative position of the ED by examining the difference of arrival times at multiple (three or more) GWs
- Only the GWs are time-synchronized.
LoRa Localization (2)

- **ToA and TDoA**
  - A LOS path should exist and be able to be easily searched (ToF)
  - Three ToF values are needed => three GWs

- **Chirp Spread Spectrum (CSS)**
  - In multipath environment, the LOS path can be more easily searched

\[
S(t - \tau_0) + S(t - \tau_1) + S(t - \tau_2) + \cdots
\]

\[
\cdots \text{ by multiplying } S^*(t - \tau_X)
\]
LoRa Localization (3)

- **ToA**
  - \[ d_i = c \ (t_i - t_0) \]
    - \( c \): propagation speed of light
    - \( t_i \): signal reception time at GW1
    - \( t_0 \): signal transmission time at a target ED

- **TDoA**
  - \[ d_i - d_j = c \ (t_i - t_0) - c \ (t_j - t_0) \]
    - \[ = c \ (t_i - t_j) \]

The position of each is fixed and known.
Low-cost, Long-range Open IoT for Smarter Rural African Villages

- Some typical IoT applications where real-time data collection could greatly increase quality and productivity in rural environments.

- Livestock farming
- Irrigation
- Fish farming
- Storage & Logistics
- Agriculture
- Fresh water
Application Example-1

- **Low-cost, Long-range Open IoT for Smarter Rural African Villages**
  - Fully autonomous gateway scenario (No communication Infra)

  - The gateway collects data from remote devices and stores data locally.
  - After post processing, the data can be viewed by the gateway used as an end computer by just attaching a keyboard and a display.
  - The gateway can also interact with the end-users’ smartphone through WiFi or Bluetooth.
Application Example-2

- **Low power wide area Bat communication networks**
  - Observing the behavior of large animal populations in their natural habitat
  - Conventional Approach
    - employ radio telemetry utilizing directional antennas and manual triangulation.
    - To localize a single individual the receivers must be hand-operated by scientific staff resulting in considerable amount of human effort and costs.
    - Just few tracking samples are preserved
Application Example-2

- Low power wide area Bat communication networks
  - IoT-based Approach
    - Short range and long range communication networks
    - Sensors: bats
    - Observation of entire groups of bats in their natural habitat