Design and Implementation of WSN for Home Automation

Benchekroun Amine
Diani Badr
Chagot Alix
Marce Clément
Louis Rousset

5 ISS
2024 / 2025

Table of content:

Introduction **2**
Requirements Analysis **3**
System Overview **4**
Table 1: Periodic Collectors 4
Table 2: Actuators 5
Table 3: Event Sensors 5
Table 4: Emergency Sensors 6
Table 5:Definition of Needs 7
Physical Layer Design **8**
1- Introduction to Binary Phase Shift Keying (BPSK) 8
2- Simulation of the BPSK without PlutoSDR 9
3- Simulation of the BPSK with PlutoSDR 10
MAC Layer Design **12**
1 - CSMA/CA Implementation 12
2 - Priority Management for Critical Sensors 12
3 - Energy Efficiency Considerations 13
4 - Performance Metrics and Evaluation 13
5 - Security Considerations 13
6 - Integration with GNU Radio 14
7 - Future Developments 15
Conclusion **16**

Introduction

WSNs are a cornerstone in modern IoT applications, enabling devices to communicate without wires in varied environments. In this paper, we present the development and implementation of a WSN for home automation. The project aims at achieving a system that efficiently manages sensors and actuators with secure, reliable, and fast communication while effectively optimizing energy consumption.

The typical wireless sensor network for smart homes includes sensors and actuators connected over some gateway for intelligent monitoring and efficient operation of the ambient real-time home. The types of nodes to be accommodated may range from energy usage to different emergency situations related to a case of falls by using motion detectors and detection alarm devices. Such integration challenges arise particularly within a constrained environment for maintaining latency bounds, assurance over the security of the data, and power consumptions in a tight and often limited resource area.

To address these challenges, our project leverages a multi-layered design approach focusing on the Physical (PHY) and Medium Access Control (MAC) layers. The PHY layer is responsible for transmitting data using a robust modulation scheme, while the MAC layer is responsible for ensuring efficient channel access and collision avoidance. This layered design is implemented and tested using SDR platforms such as GNU Radio and USRP.

The main objectives of this work are to prove the feasibility of deploying WSNs for smart homes by taking into consideration security, quality of service, and energy efficiency. The knowledge and results gained from this research will contribute to developing intelligent, sustainable, and user-friendly home automation.

In the subsequent sections, we present the requirements, system architecture, design decisions, and performance evaluation that give a comprehensive overview of the WSN implementation for home automation.

Requirements Analysis

In our laboratory dedicated to wireless sensor networks (WSN) for a smart medical home, we designed and implemented the physical, MAC and application layers. The aim was to create a network capable of efficiently managing sensors and actuators, ensuring reliable communication, optimal energy management and responsiveness adapted to a medical environment.

Key requirements include :

- Mobility: Connectivity provided in a fixed (home) environment.
- Security: Use of a unique key for cryptographic operations and protection against replay attacks.
- Availability: Reserves for emergency sensors, guaranteeing priority transmission of critical information.
- Energy consumption: Flexibility for connected or regularly charged sensors.
- Quality of service: Low latency, optimized transmission, and throughput adapted to sensor type.

The network includes 10 to 100 sensors/actuators, a central communication point (gateway) for local connectivity and a link to the cloud for remote control.

This includes modulation schemes, frequency management and robustness against interference.

System Overview

Types of sensors/actuators :

- Temperature sensor
- Accelerometer
- Humidity sensor

Table 1: Periodic Collectors

Measurement	Description	Type of Sensor	Frequency	Latency	Security
Temperature	Measures the temperature in each room	Temperature sensor	Every 10 minutes	Low	Basic encryption
Indoor brightness	Measures the light level inside the rooms	Light sensor	Every 5 minutes	Low	Basic encryption
Outdoor brightness	Measures the light level outside the house	Light sensor	Every 5 minutes	Low	Basic encryption
Weather conditions	Measures humidity, wind, rain, and sunlight	Weather sensor	Every 30 minutes	Low	Basic encryption

Table 2: Actuators

Action	Description	Type of Sensor	Frequency	Latency	Security
Opening shutters	Controls external shutters	Motorized actuator	On demand	Medium (1-2 s)	Medium (authentication)
Heating control	Adjusts the temperature	Heating actuator	Based on temperature	Medium (2-5 s)	Medium (authentication)
Light control	Controls the lamps	Lamp actuator	On demand	Low (<1 s)	Medium (authentication)
Door opening	Controls motorized doors	Motorized actuator	On demand	Medium (1-2 s)	High (authentication)

Table 3: Event Sensors

Measurement	Description	Type of Sensor	Frequency	Latency	Security
Presence detection	Detects movements in rooms	Motion sensor	Real-time	Low (<1 s)	Medium
Button control	Turns devices on or off	Button	On demand	Low (<1 s)	Medium
Remote control	Allows remote system control	Remote control	On demand	Low (<1 s)	Medium

Table 4: Emergency Sensors

Measurement	Description	Type of Sensor	Frequency	Latency	Security
Fall detection	Detects a fall in a room	Fall sensor	Real-time	Very low (<1 s)	High
Emergency call button	Allows the user to request help	Emergency button	On demand	Very low (<1 s)	High
Intrusion detection	Detects unwanted presence in the house	Motion sensor	Real-time	Very low (<1 s)	High

Table 5:Definition of Needs

Measurement	Description	Type of Sensor	Frequency	Latency	Security
Temperature	Ambient temperature sensor	Periodic collector	1 measure / h	None	Low
Humidity	Ambient humidity sensor	Periodic collector	1 measure / h	None	Low
Presence sensor	Detects presence in the room	Event-driven sensor	High frequency	Low	Low
Light switch	Turns on light when presence is detected	Actuator	Triggered by presence sensor	Low	Low
Fall sensor	Detects patient falls	Emergency sensor	High frequency	Low	High

Physical Layer Design

1- Introduction to Binary Phase Shift Keying (BPSK)

We chose Binary Phase Shift Keying (BPSK) for its simplicity and robustness against noise. Although its throughput is limited, it remains suitable for applications such as temperature readings, which do not require frequent updates. This modulation reduces equipment complexity, and is well suited to the specific needs of an intelligent environment.
Binary Phase Shift Keying (BPSK) is a technique used in communication systems to transmit information over a communication channel. In BPSK, the carrier signal is modified by alternating its phase by 180 degrees for each symbol. A phase shift of 180 degrees represents a binary bit 0, while no shift indicates a binary bit 1. This modulation process is simple and efficient, making it suitable for environments where the communication channel is subject to noise and interference.

In binary phase shift keying (BPSK), symbols are represented by phase shifts in the carrier signal. A binary bit 1 is transmitted with no phase change, while a binary bit 0 is sent with a phase shift of 180 degrees. This phase shift information is encoded in the carrier signal to enable data transmission.

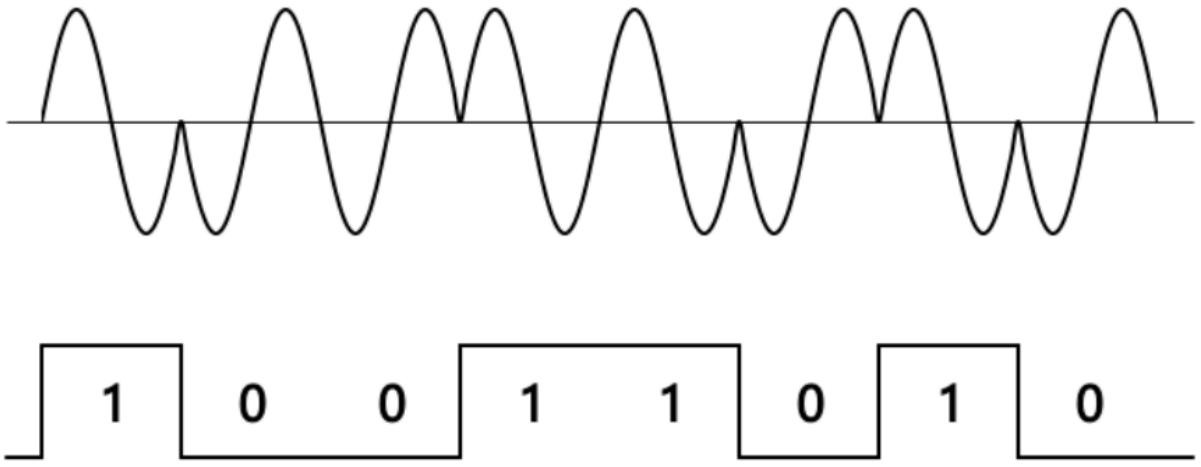


Figure 1: *Illustration of BPSK modulation: binary data and corresponding modulated signal*

The constellation diagram for BPSK shows two constellation points located along the x axis (in phase). No points are projected along the y axis (quadrature), as BPSK relies on a single basic function. As a result, all transmitted information is carried by the phase of the carrier wave.

2- Simulation of the BPSK without PlutoSDR

To better understand BPSK modulation, we chose to use GNU Radio, a tool that allows for simulating, visualizing, and experimenting with this modulation in real-time. We started by creating a flowchart that does not involve the PlutoSDR, in order to gain an initial understanding of the concepts and familiarize ourselves with the software, as shown below:

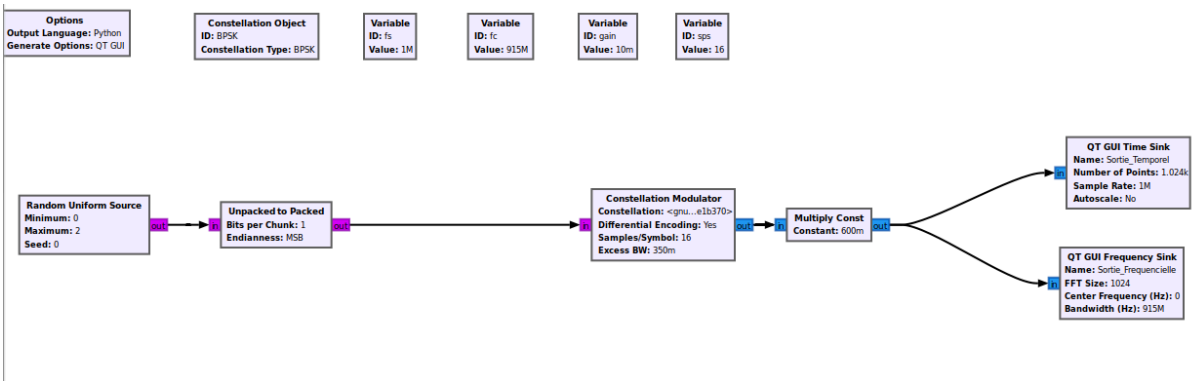


Figure 2: *Generation of BPSK signal*

- **This schematic uses the following components :**

The *Random Uniform Source* block generates random bits, simulating a data source. These bits are formatted using the *Unpacked to Packed* block to make them compatible with the modulation. The BPSK modulation is applied by the *Constellation Modulator*, configured with oversampling (SPS = 16), filtering (Excess BW = 350m), and differential encoding to avoid phase ambiguities. The signal is amplified using the *Multiply Const* block (gain = 600m) to adjust its amplitude. Finally, the signal is visualized using the *QT GUI Time Sink* and *QT GUI Frequency Sink* blocks, enabling temporal and spectral analysis to observe the signal's shape and spectrum, thereby validating the performance of the BPSK modulation.

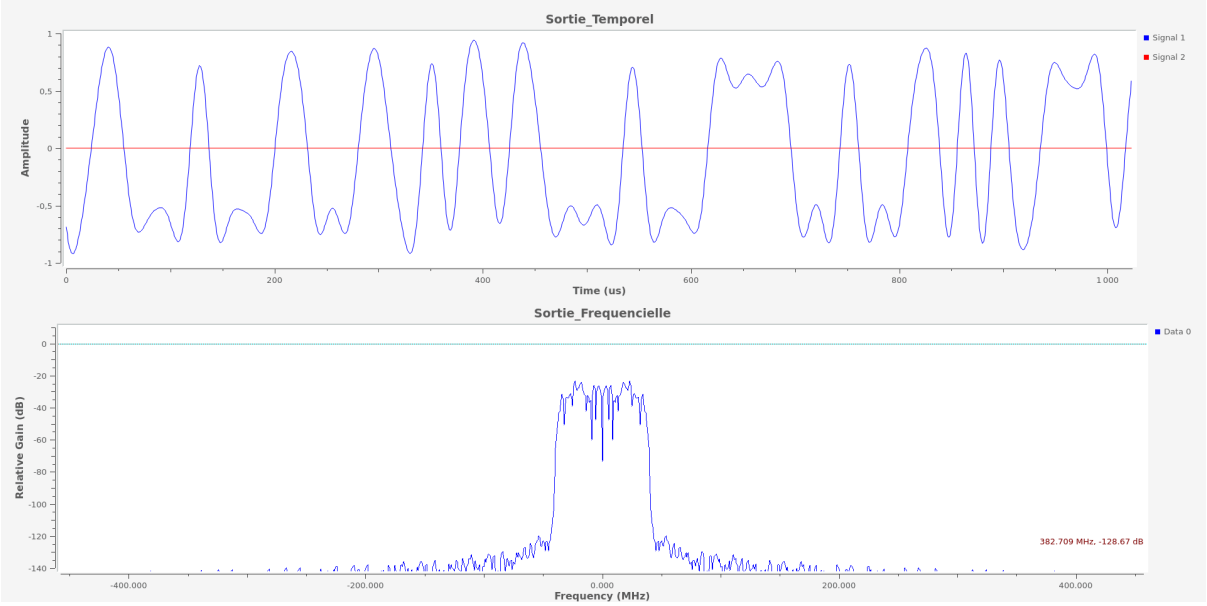


Figure 3: ***Time-domain and frequency-domain representation of the transmitted BPSK signal***

- **The obtained outputs align with the theoretical characteristics of BPSK modulation:**

In the time domain, the signal alternates between two levels (+1 and -1), corresponding to the two distinct phases (0° and 180°) of BPSK. In the frequency domain, the energy is concentrated around the center frequency with a bandwidth consistent with the configured parameters (SPS and filtering), thereby validating the correct operation of the schematic.

3- Simulation of the BPSK with PlutoSDR

We are now involving the PlutoSDR in the implementation of BPSK modulation. The use of the PlutoSDR allows us to transition from software simulation to real hardware experimentation, providing a deeper understanding of the practical aspects of BPSK modulation. By integrating the PlutoSDR, we can transmit and receive real signals, observe the effects of interference, noise, and hardware imperfections, and validate our system in a physical environment. This step is essential to test the robustness and reliability of BPSK modulation under real-world conditions.

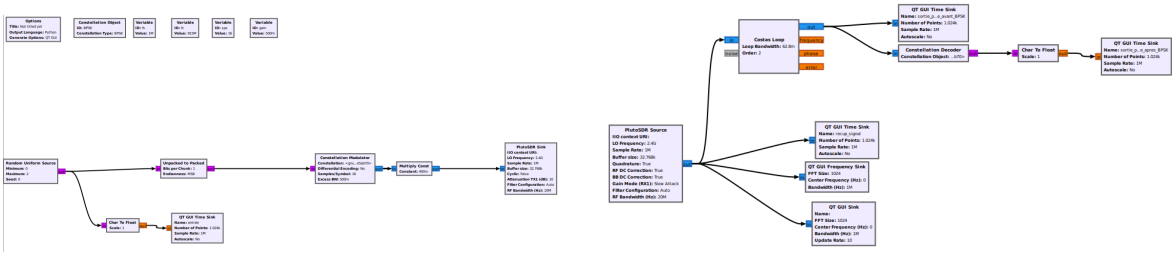


Figure 4: *Complete BPSK communication system*

To complete the simulation, we designed a BPSK communication system comprising a transmitter and a receiver. The transmitter sends the amplified signal through a PlutoSDR Sink, while the receiver, using a PlutoSDR Source, captures and analyzes the signal in both the time and frequency domains. A Costas Loop ensures phase synchronization to correct errors, and the signal is then demodulated and decoded using a Constellation Decoder to recover the original bits. Visualization tools are used to observe corrections, particularly in phase and noise, validating the overall performance of the communication system.

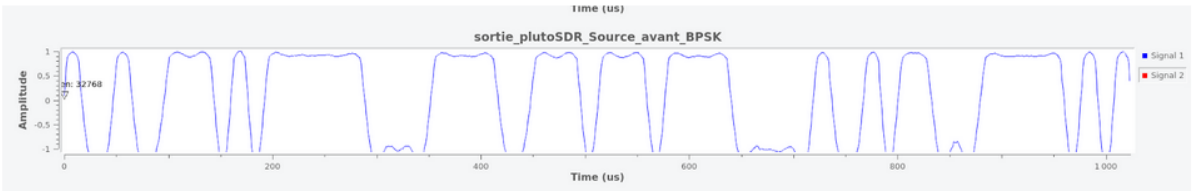


Figure 5: ***Signal received before BPSK demodulation (noisy and modulated)***

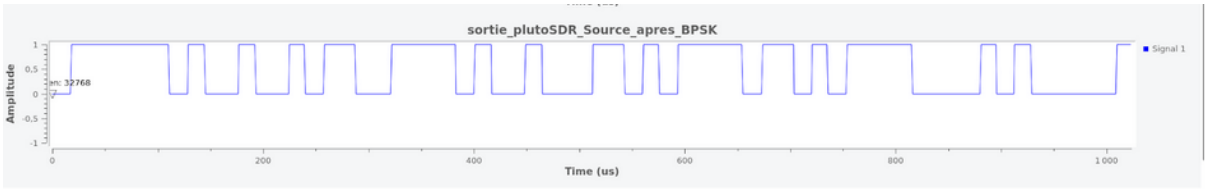


Figure 6:***Signal after BPSK demodulation (clean binary levels)***

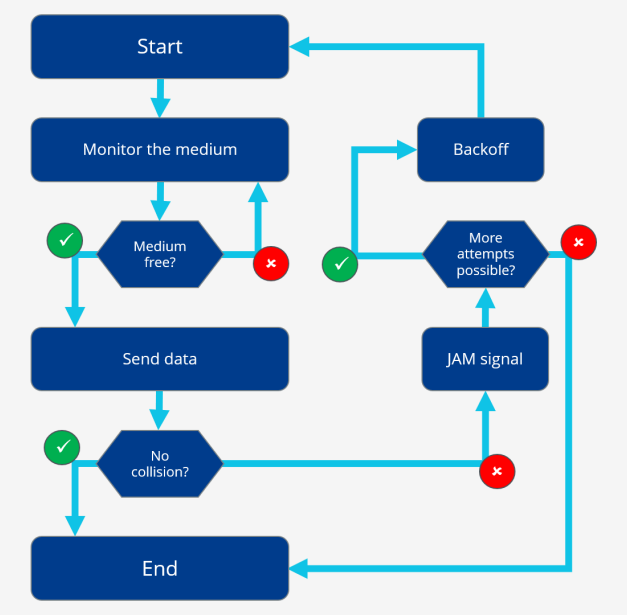
The results show the received signal before and after BPSK demodulation. Before BPSK, the signal exhibits continuous variations due to noise, phase shifts, and transmission effects inherent to the radio channel, which are typical of a modulated signal captured by the SDR receiver. After BPSK, thanks to the phase synchronization and demodulation performed by the *Costas Loop* and the *Constellation Decoder*, the signal is restored to a clean binary form, alternating between +1 and -1, corresponding to the logical levels expected for BPSK modulation. This confirms the proper operation of the transmission and reception system.

MAC Layer Design

The MAC is a core of the OSI model data link layer responsible for managing how devices share and access communication channels in wireless networks. It is critically used for addressing, error detection, and frame validation, and for assuring that the transmission of data packets in an environment with no fixed connections is performed effectively.

1 - CSMA/CA Implementation

The access method used is Carrier-Sense Multiple Access with Collision Avoidance. This protocol ensures fewer collisions by forcing devices to listen to the channel before transmission. If the channel is busy, it introduces a random backoff time to prevent simultaneous retries. CSMA/CA works very well in a smart home environment with high densities since there are numerous sensors and devices that send frequent traffic. Its implementation ensures reliable communication with reduced retransmissions and efficient use of the channel.

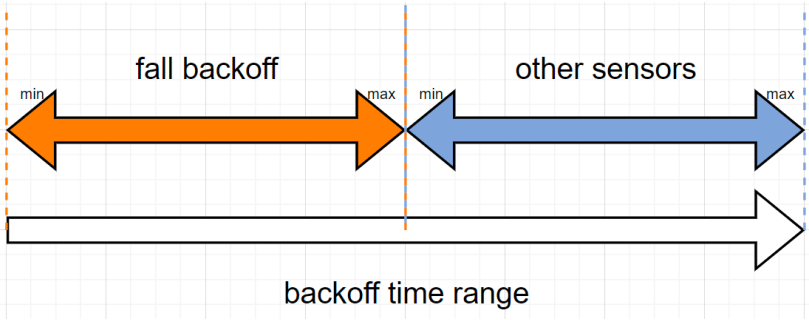


2 - Priority Management for Critical Sensors

Some sensors, like those of fall detection or emergency alarms, need more priority towards getting network access. Listed below are strategies that enhance their responsiveness:

- Time Reservations: Dedicated time slots ensure critical sensors can communicate without delays.
- Backoff Adaptation: High-priority sensors use shorter back off times, reducing their wait time.

- Dedicated Channels: Assigning unique frequencies to critical nodes avoids interference with other devices.



For example, in the event of a fall, the backoff maximum time is lower than the backoff minimum time of all the other events. By doing this, we are sure that the fall information will be sent before the others.

3 - Performance Metrics and Evaluation

To validate the MAC layer's effectiveness, the following metrics are monitored:

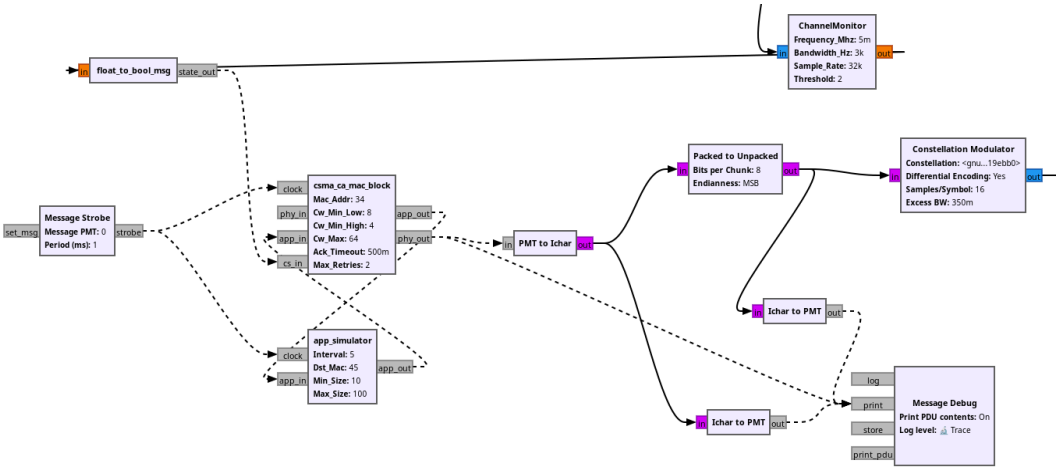
- Collision Rate: Measuring the frequency of data collisions and their impact on retransmissions.
- Latency: Assessing delays for both priority and non-priority traffic.
- Throughput: Evaluating the data transfer rate under varying network loads.
- Energy Consumption: Monitoring power usage across devices, ensuring efficiency for battery-powered nodes.

4 - Security Considerations

The use of CSMA/CA optimizes the access to the communication medium and reduces the risk of collision and thus, of packet loss in the network.

Furthermore, we would like to implement an FCS (Frame Check Sequence) to our MAC frames in order to ensure integrity of the data along exchanges.

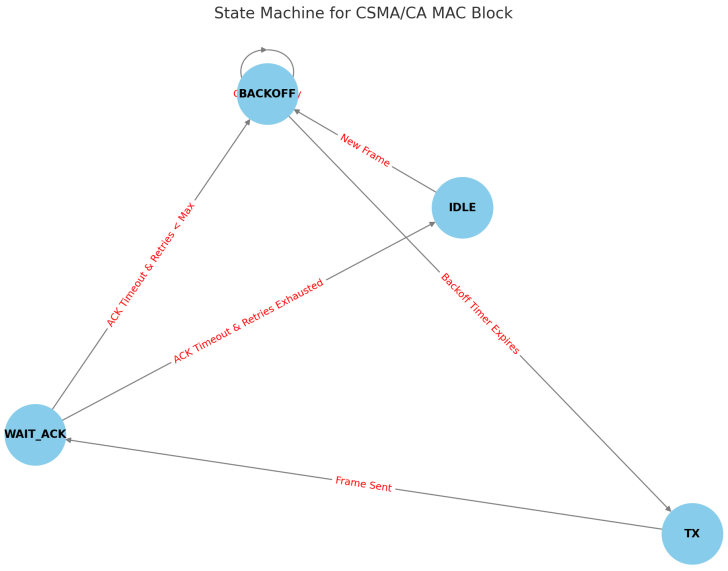
5 - Integration with GNU Radio



The CSMA/CA protocol is implemented in GNU Radio by instantiating a csma_ca_mac_block. This block contains :

- Configurable contention windows for adapting backoff times
- ACK timeouts and retry limits to ensure reliable delivery channel sensing via a carrier-sense signal for real-time contention resolution.
- Application-layer traffic is emulated and packet generation takes place in variable size to closely emulate realistic conditions. Processed packets are being transformed to a condition that enables physical layer modulation, hence meeting complete compatibility.

The python script inside is described by the following state machine diagram which follows the rules of the CSMA/CA graph we have seen earlier :



The usage of PDU to Stream block didn’t work so we tried to use custom Ichat to Stream block running the associated python script but we did not manage to make it work either.

6 - Future Developments

Protocol Interoperability: integrate the MAC layer with 6LoWPAN or IPv6 for broader IoT compatibility and scalability.
It represents the MAC layer that provides an appropriate balance between robust channel access strategies and energy efficiency, security, and adaptability as the reliable base for wireless communication management in smart homes.

Conclusion

The project discusses developing a WSN system for home automation, focusing on energy efficiency, low latency, and security in communication. The application of BPSK at the physical layer and CSMA/CA at the MAC layer of the proposed system shows the potential of this system to provide strong and prioritized communication, specifically for critical sensors like fall detectors.

However, despite these advances in designing and simulating each component, the project development still had its limitations. Particularly, we were unable to implement an application-level simulation for the entire line of communication. This blocked validation of performance and functionality from end to end for the system. Moreover, we couldn't have the MAC encapsulation protocol working and we were not able to fully implement our components and test the whole system.

Although the project was not completely successful, a lot has been learnt from this regarding WSN design, and many key issues have been found on which further research should be directed. Further efforts are required for simulations and integrating all layers for complete realization of smart home automation.