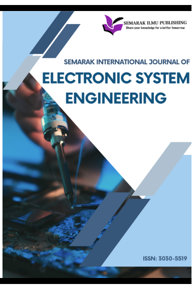
Semarak International Journal of Electronic System Engineering

Journal homepage: https://semarakilmu.my/index.php/sijese/index
ISSN: 3030-5519



Low-Cost Software Defined Radio Transmitter using Raspberry Pi and HackRF One

Shujahat Qamar[1], Ehtesham Ali[2,*] Syed Muzzamil Ali Shah[3]

[1] Department of Electronics, The University of Haripur, Haripur 22620, Pakistan
[2] Faculty of Electrical and Electronics Engineering Technology, University Malaysia Pahang, Al-Sultan Abdullah, Pekan, Pahang, Malaysia
[3] Department of Electrical Engineering, Lappeenranta-Lahti University of Technology LUT Finland

ARTICLE INFO

Article history:
Received 10 July 2025
Received in revised form 23 August 2025
Accepted 28 August 2025
Available online 9 September 2025

ABSTRACT

Software Defined Radio (SDR) enables flexible and reconfigurable wireless communication by replacing most hardware-based signal processing with software, reducing cost and complexity. This paper aims to design and implement a low-cost SDR transmitter for short-range FM audio broadcasting using a Raspberry Pi 4 and HackRF One module. The system operates within the 1 MHz to 500 MHz range and utilizes GNU Radio for real-time modulation, filtering, and transmission control, adopting a homodyne architecture for simplicity, compactness, and energy efficiency. Hardware components, software configuration, and signal processing flowgraphs were developed and integrated into a functional prototype. Experimental results demonstrated stable FM transmission over distances up to 50 meters, delivering clear audio at short range and acceptable quality at maximum distance. These findings highlight the system's potential as an affordable platform for education, experimentation, and rapid prototyping in wireless communications.

Keywords:
Software Defined Radio (SDR), Raspberry Pi FM audio

1. Introduction

Software Defined Radio (SDR) is a radio communication system where traditional hardware components such as mixers, filters, amplifiers, modulators/demodulators, and detectors are implemented through software on a general-purpose computing platform or embedded system, rather than fixed-function analog hardware [1].

The idea behind SDR emerged in the late 1980s and early 1990s, initially driven by military requirements for radios that could quickly adapt to different communication standards without requiring physical hardware replacements [2]. By replacing hardware-based signal processing with Software algorithms, SDR allows for high flexibility, easy reconfiguration, and compatibility with multiple frequency bands and modulation schemes.

[*] *Corresponding author.*
E-mail address: Ehteshamali@gmail.com

https://doi.org/10.37934/sijese.7.1.19

A typical SDR system uses an RF front-end for basic signal conversion (analog to digital or vice versa) and a digital signal processor (DSP), general-purpose processor (GPP), or field-programmable gate array (FPGA) for implementing the signal processing tasks [3]. This design makes it possible to update or enhance a radio system through software upgrades rather than building new hardware.

The fig 1 illustrates the diverse applications of Software Defined Radio (SDR) across multiple domains. In wireless communications, SDR enables seamless operation over cellular networks, Wi-Fi, Bluetooth, and satellite systems. It serves as a powerful tool in research and education, supporting experimentation, prototyping, and learning in communication technologies. In military and emergency communication systems, SDR provides adaptable, secure, and reliable communication solutions for mission-critical operations [4]. Amateur radio operators utilize SDR for its ability to access multiple frequency bands and modes. Additionally, SDR is instrumental in spectrum monitoring and signal intelligence, allowing efficient detection, analysis, and management of radio frequency activity.

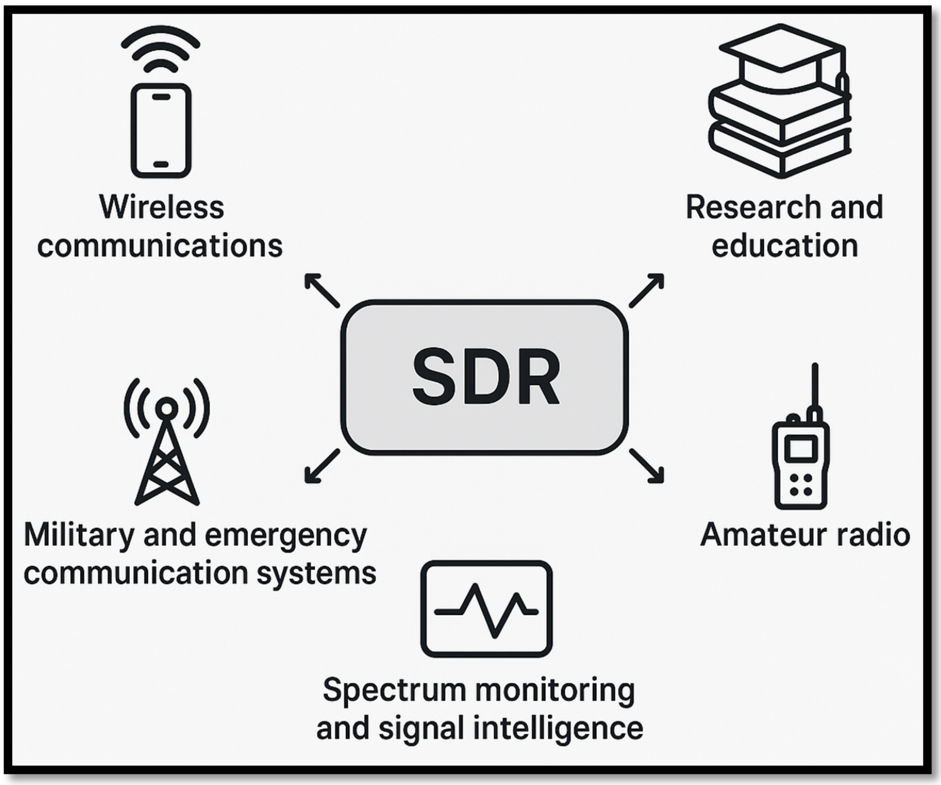

Wireless communications
Research and education
SDR
Military and emergency communication systems
Amateur radio
Spectrum monitoring and signal intelligence

Fig. 1. Key Applications of Software Defined Radio (SDR)

1.1 Advantages of SDR Over Traditional Radios

Compared to conventional hardware-defined radios, Software Defined Radios (SDRs) offer significant advantages, including reconfigurability, where protocols, modulation schemes, and operating frequencies can be modified through software updates without altering the hardware. They also provide cost efficiency by eliminating the need for multiple dedicated radios to support different standards [5]. With multi-standard capability, SDRs can operate on protocols such as GSM, LTE, Wi-Fi, and Bluetooth using the same hardware platform. Additionally, they facilitate rapid prototyping, enabling quick testing and deployment of new communication algorithms. Their flexibility also holds strong educational value, allowing students and hobbyists to explore and experiment with wireless communication concepts without the need for costly laboratory equipment [6].

1.2 Project Motivation

Traditional hardware radios are often limited to fixed functions and require complex, costly modifications to support new communication standards. This limitation is particularly challenging for researchers, students, and hobbyists who need a flexible and affordable platform for testing various wireless protocols [7]. Pairing a Raspberry Pi with an SDR device such as the HackRF One offers a cost-effective, portable, and energy-efficient alternative. This combination provides a powerful experimental platform for implementing, testing, and analyzing radio systems without the high investment required for commercial-grade SDR platforms such as the USRP.

1.3 Focus of This Work

This study focuses on developing a transmitter-only SDR system using a Raspberry Pi 4 and HackRF One. The main objectives are:

- To design a compact and portable SDR transmitter suitable for experimental and educational purposes.
- To demonstrate FM signal generation and transmission using GNU Radio on a low-cost hardware platform.
- To provide a clear, replicable implementation method for students, researchers, and hobbyists.

The design emphasizes simplicity, affordability, and reconfigurability, ensuring accessibility for a broad audience interested in wireless communication experimentation.

1.4 SDR vs Traditional Radio

In traditional radios, the signal captured by the antenna passes through multiple stages of analog hardware, including amplifiers, filters, and mixers, to isolate and process the desired frequency. The conversion from analog to digital occurs later in the chain, meaning that most modulation, demodulation, and filtering tasks are carried out using fixed-function electronic circuits. Because these tasks depend on dedicated components, changing the system to support a new frequency band, modulation scheme, or communication standard often requires replacing or redesigning hardware [8]. This makes traditional radios less adaptable, slower to upgrade, and more costly to modify.

In SDRs, the design shifts most of the processing workload from hardware to software. After minimal analog front-end processing, the signal is converted to digital form early using high-speed analog-to-digital converters (ADCs). Once in digital form, all core processing such as modulation, demodulation, filtering, and protocol handling, is performed by software running on general-purpose processors (GPPs), digital signal processors (DSPs), or field-programmable gate arrays (FPGAs) [9]. For transmission, the reverse occurs, with software generating the digital waveform, which is then converted back to analog using a digital-to-analog converter (DAC) [10].

This architecture offers unparalleled flexibility: the same SDR hardware can support multiple communication standards simply by updating the software. As a result, SDRs enable rapid prototyping, easy reconfiguration, multi-standard capability, and lower long-term costs compared to traditional radios, which rely heavily on dedicated, non-programmable circuitry.Fig 2 compares the architectures of traditional hardware-defined radios and software-defined radios (SDRs), highlighting their fundamental differences in signal processing.

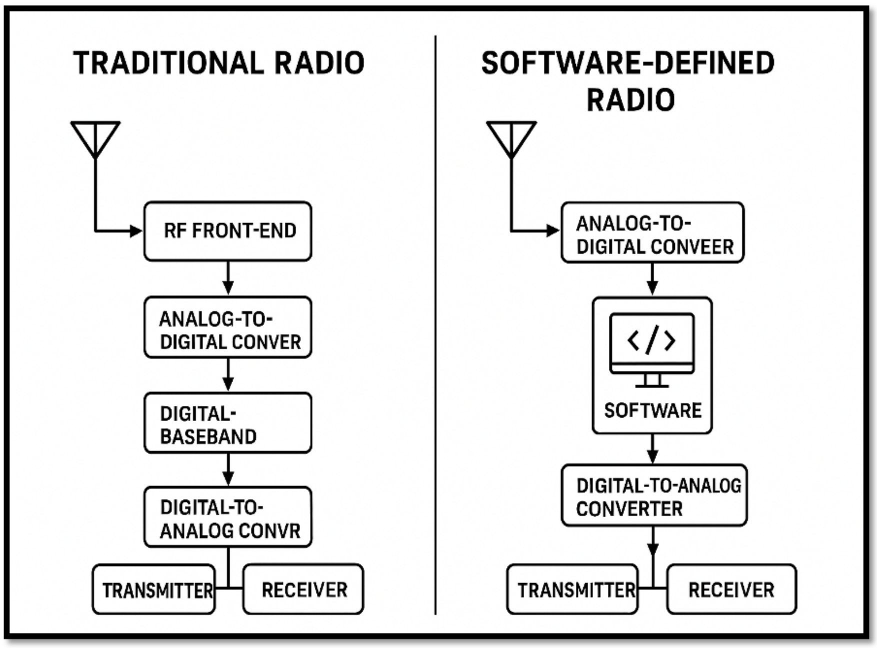


Fig. 2. Compares the architectures of traditional hardware-defined radios and software-defined radios (SDRs), highlighting their fundamental differences in signal processing

Despite the proven advantages of SDR, many existing implementations rely on costly, high-end platforms such as USRP, which limits accessibility for budget-conscious researchers, educational institutions, and hobbyists. Affordable SDR solutions are often receiver-focused, with fewer documented, reproducible designs for transmitter-only systems that can be easily deployed for learning and experimentation. Furthermore, limited resources exist that integrate low-cost single-board computers like the Raspberry Pi with SDR hardware to deliver a portable, power-efficient, and functional transmission platform.

To address this gap, the present study aims to design and implement a compact, low-cost SDR transmitter using a Raspberry Pi 4 and HackRF One, capable of generating and transmitting FM signals via GNU Radio. The project prioritizes affordability, portability, and ease of replication, ensuring that students, educators, and independent researchers can leverage SDR technology without substantial financial or technical barriers.

The significance of this work lies in its contribution to democratizing access to SDR experimentation. By offering a practical, step-by-step implementation of a functional transmitter platform, this study supports hands-on learning, fosters rapid prototyping in wireless communication research, and encourages innovation in resource-constrained environments.

2. Literature Review

Early concepts of Software-Defined Radio (SDR), first introduced by Mitola in the 1990s, marked a paradigm shift in wireless communication by replacing fixed analog circuitry with flexible, software-driven signal processing [11]. Initially, SDR technology found its primary applications in military and defense systems, where the ability to adapt rapidly to changing communication standards and frequencies offered a strategic advantage [12]. Over time, advancements in processing power, miniaturization, and open-source development enabled the migration of SDR into commercial, educational, and research domains [13].

The early low-cost implementations by developing a Raspberry Pi-based SDR for disaster communication. Their work demonstrated that portable, energy-efficient, and cost-effective SDR systems could serve as viable tools for emergency networks in resource-constrained environments. This breakthrough paved the way for broader experimentation in non-military applications [14].

More recent studies have explored SDR's potential in diverse areas such as Open Radio Access Networks (Open RAN), Unmanned Aerial Vehicle (UAV) communication systems, and cognitive radio networks, which dynamically adjust spectrum usage to optimize performance. However, much of this research relies on sophisticated and high-cost platforms like the Universal Software Radio Peripheral (USRP), which, while offering exceptional flexibility and performance, can be financially prohibitive for widespread deployment [15].

Building on this foundation, the present work proposes a simplified, transmitter-only SDR configuration tailored for affordable hardware platforms. The design focuses on preserving adaptability for multiple modulation schemes and frequency bands while significantly reducing complexity and cost, making it more accessible for educational projects, rural communication networks, and small-scale experimental deployments.

3. Methodology

The system is built around a Raspberry Pi 4 (4 GB RAM), which serves as the central processing platform for all signal handling and system control tasks. A HackRF One device, capable of operating across a wide frequency range from 1 MHz to 6 GHz, is connected to the Raspberry Pi to handle the conversion of the processed baseband signal into an RF signal suitable for over-the-air transmission. A standard microphone connected to the Raspberry Pi provides live audio input, which forms the content of the transmission. Finally, an external antenna attached to the HackRF One radiates the RF signal into the air, allowing it to be received by compatible radio receivers.

Figure 1 illustrates the signal flow and functional components of the proposed Software-Defined Radio (SDR) transmitter system

The software environment for the system is based on GNU Radio 3.10, an open-source digital signal processing toolkit that allows for the creation of signal flowgraphs through a modular block-based interface. Within GNU Radio, the incoming audio is first processed through the Wideband FM (WBFM) transmit block, which applies frequency modulation to encode the audio signal onto a carrier wave, making it suitable for FM broadcast. Following this, a low-pass filter is applied to suppress unwanted high-frequency noise and harmonics that could degrade transmission quality or cause interference in adjacent frequency channels.

Once filtered, the baseband FM signal is passed to the Osmocom Sink block, which interfaces directly with the HackRF One. The HackRF One then upconverts the modulated signal from its baseband form to the target RF carrier frequency (for example, around 100 MHz for FM radio broadcasting). The upconverted RF signal is output to the connected antenna, which converts the electrical signal into electromagnetic waves, enabling wireless transmission over a considerable range depending on antenna characteristics and output power.

For monitoring and verification, the system uses applications such as GQRX and SDRangel, which can visualize the transmitted spectrum, measure signal strength, and confirm modulation accuracy in real time. This ensures that the transmission adheres to the intended frequency, modulation parameters, and signal quality requirements.

Remote operation is supported via SSH or PuTTY, allowing the Raspberry Pi to be configured and controlled over a network connection without a direct physical interface. This enables flexible deployment of the SDR transmitter in various environments, including locations where local monitoring is not feasible.

Overall, the integration of low-cost, general-purpose computing hardware (Raspberry Pi) with a versatile SDR peripheral (HackRF One) and open-source DSP software (GNU Radio) results in a flexible, reconfigurable, and portable FM transmission system. The modular design allows for quick reconfiguration to support different modulation schemes, frequencies, and input sources, making it suitable for educational, experimental, and communication applications.

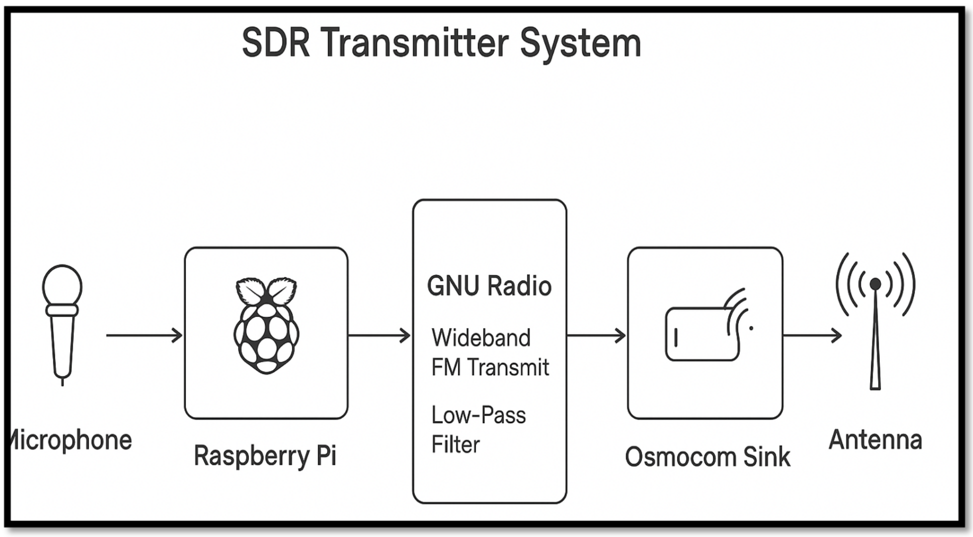


Fig. 3. Block Diagram of SDR Transmitter

4. Results

The SDR transmitter system was evaluated for its FM audio transmission performance across three distinct distances: 5 meters, 20 meters, and 50 meters. For each test, the system was configured to broadcast an audio signal generated through GNU Radio on the Raspberry Pi 4, with the HackRF One serving as the RF front-end. A standard consumer-grade FM radio receiver, tuned to the transmission frequency, was used to assess the clarity, stability, and strength of the received audio signal. The tests were conducted in an open outdoor environment to minimize multipath interference and physical obstructions, ensuring a consistent line-of-sight between the transmitter and receiver. At each distance, qualitative observations of audio quality and background noise were recorded, while signal strength was monitored using the radio’s built-in signal meter. Figure 4 illustrates the experimental setup used for the transmission performance evaluation.

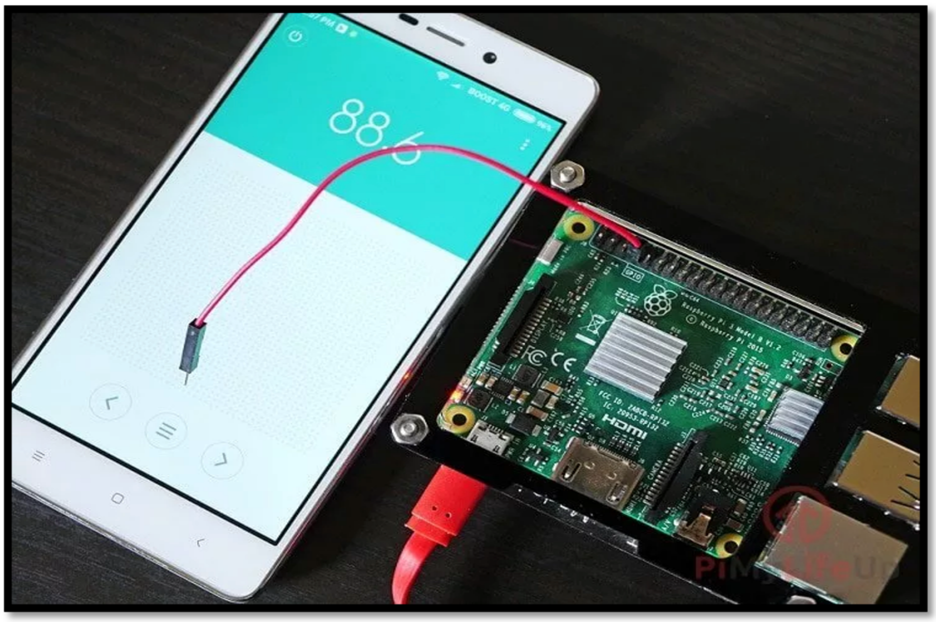


Fig. 4. Overview of SDR transmitter system

I. Transmission Performance

- 5 Meters: At a short distance of 5 meters, the received signal strength was excellent, and the audio was clear without noticeable distortion or background noise.
- 20 Meters: At 20 meters, the signal remained strong, but minor noise interference was observed, likely due to environmental obstacles and the HackRF One's limited transmit power.
- 50 Meters: At the maximum tested distance of 50 meters, the signal was still detectable, and the audio was intelligible; however, noticeable distortion and background hiss were present.

Table 1
Signal vs Audio Quality

Distance (m)	Signal Strength	Audio Quality
5	Excellent	Clear
20	Good	Minor noise
50	Acceptable	Slight distortion

II. Power Consumption

During continuous transmission, the combined Raspberry Pi 4 and HackRF One setup consumed approximately 6.5 W, making it energy-efficient and suitable for battery-powered or portable applications.

III. Observations

The system successfully demonstrated FM audio transmission over short to moderate distances. However, the HackRF One's inherently low output power limits the achievable range without the use of an external RF power amplifier. Environmental factors such as building walls, interference from nearby electronics, and antenna positioning also affected the quality of received signals. Despite these limitations, the setup proved effective for educational demonstrations, SDR experimentation, and low-power communication tests.

5. Conclusion

This work presents the successful development of a low-cost Software Defined Radio (SDR) transmitter using a Raspberry Pi 4 paired with a HackRF One module. The system effectively transmits FM audio signals over distances of up to 50 meters, delivering satisfactory signal strength and audio clarity. Powered by GNU Radio for real-time modulation and signal processing, the setup is particularly well-suited for educational and experimental use. The findings underscore the SDR's value as an accessible platform for students, researchers, and hobbyists to explore wireless communication concepts without the high expenses of conventional radio systems. While the transmission range is limited by the HackRF One's modest output power, the design's simplicity, flexibility, and affordability make it a practical resource for learning and experimentation. Future enhancements could include the integration of external RF power amplifiers to extend range and improve performance. Overall, this work adds to the growing body of SDR research by delivering a practical, adaptable, and cost-effective solution for wireless communication experimentation.

Acknowledgement
This research was not funded by any grant

References

[1] Abttan, Rana Ali, Adnan Hasan Tawafan, and Samar Jaafar Ismael. "Economic dispatch by optimization techniques." *International Journal of Electrical and Computer Engineering* 12, no. 3 (2022): 2228-2241. https://doi.org/10.11591/ijece.v12i3.pp2228-2241.

[2] Chen, Y., H. Liu, and R. Feng. 2021. "Multi-Objective Optimization Methods for Economic Dispatch: A Comparative Study." Sustainable Energy Solutions 56 (1): 48–65.

[3] Chen, H., and Q. Liu. 2022. "Bio-Inspired Optimization for Environmental Considerations in Power Dispatch." Renewable Energy Research 29 (3): 230–244.

[4] Ailani, G. n.d. "Multi-Objective Optimization." WallStreetMojo. Accessed January 19, 2024. https://www.wallstreetmojo.com/multi-objective-optimization/.

[5] Hassan, S. S., M. A. Zohdy, and M. A. Ahmed. 2020. "Multi-Objective Barnacles Mating Optimization for Solving Economic Dispatch Problems." Soft Computing 24 (5): 3855–3869.

[6] Ismail, Nor Laili, Ismail Musirin, Nofri Yenita Dahlan, Mohd Helmi Mansor, and A. V. Sentilkumar. "Integrated Optimization Algorithm in Solving Economic Dispatch Problems." In *2023 IEEE International Conference on Artificial Intelligence in Engineering and Technology (IICAIET)*, pp. 129-134. IEEE, 2023.https://doi.org/10.1109/IICAIET59451.2023.10291341.

[7] Marzbani, Fatemeh, and Akmal Abdelfatah. "Economic dispatch optimization strategies and problem formulation: A comprehensive review." *Energies* 17, no. 3 (2024): 550. https://doi.org/10.3390/en17030550.

[8] Mirjalili, Seyedali, Amir H. Gandomi, Seyedeh Zahra Mirjalili, Shahrzad Saremi, Hossam Faris, and Seyed Mohammad Mirjalili. "Salp Swarm Algorithm: A bio-inspired optimizer for engineering design problems." *Advances in engineering software* 114 (2017): 163-191.https://doi.org/10.1016/j.advengsoft.2017.07.002.

[9] Mouassa, S., M. Rahli, and B. Bendjedia. 2021. "Barnacles Mating Optimization: A New Bio-Inspired Algorithm for Optimization Problems." Engineering Applications of Artificial Intelligence 97: 104054.

[10] Patel, A., and P. Singh. 2020. "Sustainable Dispatch Solutions and Their Implications." International Journal of Power and Energy Systems 45 (2): 182–190.

[11] Rahman, S., and R. Singh. 2020. "Addressing the Dynamic Constraints in Real-Time Economic Dispatch." IEEE Transactions on Power Systems 35 (4): 1420–1430.

[12] Savic, Dragan. "Single-objective vs. multiobjective optimisation for integrated decision support." (2002).

[13] Sulaiman, Mohd Herwan, Zuriani Mustaffa, Mohd Mawardi Saari, and Hamdan Daniyal. "Barnacles mating optimizer: a new bio-inspired algorithm for solving engineering optimization problems." *Engineering Applications of Artificial Intelligence* 87 (2020): 103330. https://doi.org/10.1016/j.engappai.2019.103330.

[14] Touma, Haider J. "Study of the economic dispatch problem on IEEE 30-bus system using whale optimization algorithm." *International Journal of Engineering Technology and Sciences* 3, no. 1 (2016): 11-18.. https://doi.org/10.15282/ijets.5.2016.1.2.1041.

[15] Rao, M. V., C. District, and I. M. V. Rao. n.d. “Economic Load Dispatch.” Unpublished.

[16] Wood, A. J., B. F. Wollenberg, and G. B. Sheble. 2013. Power Generation, Operation, and Control. 3rd ed. Wiley.

[17] Zhou, Xinyang, Chin-Yao Chang, Andrey Bernstein, Changhong Zhao, and Lijun Chen. "Economic dispatch with distributed energy resources: Co-optimization of transmission and distribution systems." *IEEE Control Systems Letters* 5, no. 6 (2020): 1994-1999. https://doi.org/10.1109/LCSYS.202