iJRASET
International Journal For Research in Applied Science and Engineering Technology

INTERNATIONAL JOURNAL FOR RESEARCH

IN APPLIED SCIENCE & ENGINEERING TECHNOLOGY

Volume: 2026 **Issue:** Conference **Month of publication:** May 2026

DOI: https://doi.org/10.22214/ijraset.2026.82948

www.ijraset.com

Call: 08813907089 | E-mail ID: ijraset@gmail.com

International Journal for Research in Applied Science & Engineering Technology (IJRASET)
ISSN: 2321-9653; IC Value: 45.98; SJ Impact Factor: 7.538
Volume 14 Issue V May 2026- Available at www.ijraset.com
National Conference on Innovation in Engineering and Technology-(NCIET2026)

Morse Code Generator Using NE555 Timer IC with ESP32-Based Decoding and Bluetooth Transmission

Shravani Sunil Mundhe[1], Sanskruti S. Ambekar[2], Ashok R. Suryawanshi[3]
Department of Electronics & Telecommunication Engineering Pimpri Chinchwad College of Engineering, Pune, India

Abstract: ***This paper presents the design, implementation, and validation of a hardware-based Morse code generator utilizing the NE555 timer integrated circuit (IC) configured in monostable mode. The system generates precisely timed electrical pulses corresponding to Morse code dots (1 second) and dashes (3 seconds) using the well-established timing relation T = 1.1RC. Component values are analytically derived: a fixed capacitance of 10 µF with resistors of approximately 91 kΩ and 270 kΩ for dot and dash intervals, respectively. The generated pulses drive multiple output transducers—a light-emitting diode (LED), a piezoelectric buzzer, and a seven-segment display—enabling simultaneous visual and auditory Morse signal representation. Integration with the ESP32 microcontroller extends the system with real-time character decoding, serial monitor display, and Bluetooth-based wireless transmission of decoded alphanumeric data. Hardware realization progresses from breadboard prototyping through oscilloscope-verified waveform analysis to printed circuit board (PCB) fabrication. Experimental results confirm output pulse fidelity within ±2% of design targets across all implemented characters. The system demonstrates a practical, cost-effective, and pedagogically valuable platform bridging classical telecommunication techniques with contemporary embedded and wireless communication paradigms.***

Keywords: NE555 Timer IC; Monostable Multivibrator; Morse Code Generator; Pulse Generation Circuit; ESP32 Microcontroller; Bluetooth Transmission; PCB Fabrication; Embedded Systems; Pulse Width Modulation; Wireless Communication.

I. INTRODUCTION

Morse code, conceived by Samuel F. B. Morse and Alfred Vail in the 1830s, remains one of the most historically significant and enduring encoding schemes in telecommunication [1]. Its binary temporal structure—short pulses (dots) and long pulses (dashes)—renders it inherently robust against channel noise and eminently suited to low-bandwidth, constrained-resource communication environments. Despite the proliferation of digital modulation techniques, Morse code retains operational relevance in amateur radio, maritime emergency signaling, and assistive communication technology [2].

The NE555 timer IC, introduced by Signetics Corporation in 1972, represents one of the most widely deployed analog integrated circuits in history [3]. In monostable configuration, the device produces a single, precisely defined output pulse of duration T = 1.1RC upon receipt of a negative-edge trigger, making it an ideal candidate for hardware-level pulse generation without reliance on software scheduling or microcontroller timing peripherals. Contemporary embedded systems platforms such as the Espressif ESP32 system-on-chip (SoC) offer substantial computational resources, integrated Bluetooth Classic and BLE stacks, and rich peripheral interfaces, enabling the co-integration of real-time signal decoding with wireless data dissemination [4]. The convergence of classical analog timer circuitry with modern embedded intelligence creates a compelling didactic and functional system that is the subject of the present investigation. This paper makes the following contributions: (i) analytical derivation and experimental validation of NE555 monostable timing components for Morse dot and dash generation; (ii) multi-transducer output integration encompassing LED, buzzer, and seven-segment display; (iii) ESP32-based firmware for real-time Morse decoding and Bluetooth transmission; and (iv) complete hardware realization from breadboard prototype to PCB fabrication with oscilloscope-verified performance characterization. The remainder of this paper is organized as follows. Section II reviews pertinent literature. Section III describes the system architecture and methodology. Section IV details circuit design and quantitative calculations. Section V discusses implementation. Section VI presents and analyzes experimental results. Sections VII–IX address applications, limitations, and conclusions, respectively.

II. LITERATURE REVIEW

Extensive literature exists on the NE555 timer IC and its applications in timing and pulse generation circuits. Horowitz and Hill [3] provide a comprehensive treatment of the device in monostable and astable configurations, establishing the foundational timing equations that underpin the present design. Jung [5] offers a rigorous analysis of component tolerance effects on timer accuracy, noting that resistor and capacitor tolerances are the primary contributors to pulse-width variation.

Morse code hardware implementations have been documented in varying degrees of sophistication. Nallatech et al. demonstrated an FPGA-based Morse encoder achieving sub-microsecond timing precision [6], while embedded microcontroller approaches have been reported using PIC and AVR platforms with software-defined timing loops [7]. However, purely analog hardware implementations using the NE555—which offer independence fromclock jitter and software scheduling latency—remain comparatively underexplored in contemporary literature.

The ESP32's integration of dual-core Xtensa LX6 processors, 520 kB SRAM, and certified Bluetooth 4.2/5.0 stacks has made it the preferred platform for IoT edge nodes requiring concurrent sensor processing and wireless communication [4]. Kolban [8] documents the ESP32 development ecosystem comprehensively, including FreeRTOS task scheduling primitives that enable deterministic interrupt-driven signal capture. Prior work on Morse decoding using microcontrollers [9] has primarily targeted software-based solutions; the hybrid analog-digital architecture proposed herein is novel in its combination of hardware-timed generation with ESP32-based intelligence.

Seven-segment display interfacing and multi-output peripheral management are well-established areas documented by Floyd [10]. The present system synthesizes these established techniques into a coherent multi-output, multi-transducer platform, representing an incremental yet meaningful contribution to the intersection of classical telegraphy and modern embedded systems.

III. METHODOLOGY / SYSTEM DESIGN

A. System Architecture Overview

The proposed system is partitioned into three functional subsystems: (1) the analog pulse generation stage, realized with NE555 timer ICs; (2) the multi-transducer output stage comprising LED, buzzer, and seven-segment display; and (3) the digital intelligence stage implemented on the ESP32 SoC. The overall block diagram is illustrated in Fig. 1.

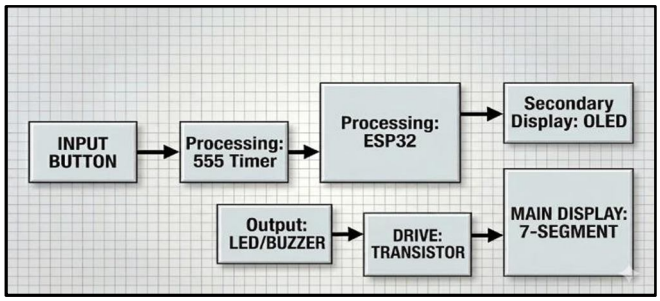


Fig. 1. System block diagram of the Morse code generator.

B. Operational Principle

The operator initiates a Morse symbol by pressing one of two dedicated push-button switches: SW_DOT or SW_DASH. Each actuation triggers the corresponding NE555 monostable circuit via a negative-edge trigger on Pin 2. The output at Pin 3 transitions HIGH for the analytically determined interval T, driving the transducer array simultaneously. Upon pulse completion, the ESP32 GPIO interrupt captures the falling edge of the NE555 output, classifies the pulse duration, and appends the corresponding symbol (dot or dash) to an internal shift register. Inter-symbol gaps and inter-character gaps are detected by a software timeout mechanism. Upon character boundary detection, the accumulated symbol string is resolved against a lookup table encoding the International Morse Code standard, and the decoded character is transmitted over the Bluetooth Serial Port Profile (SPP).

C. Design Constraints

The design is constrained to standard E24-series passive components, single-supply 5 V operation, and a worst-case timing tolerance of ±5% relative to nominal dot and dash durations. The ESP32 firmware is implemented within the Arduino framework using the ESP32 Arduino core, ensuring broad reproducibility.

International Journal for Research in Applied Science & Engineering Technology (IJRASET)
ISSN: 2321-9653; IC Value: 45.98; SJ Impact Factor: 7.538
Volume 14 Issue V May 2026- Available at www.ijraset.com
National Conference on Innovation in Engineering and Technology-(NCIET2026)

IV. CIRCUIT DESIGN AND CALCULATIONS

A. NE555 Monostable Timing Formula

In monostable mode, the output pulse width T of the NE555 is determined solely by the external timing resistor R and capacitor C according to the following relation [3]:

$$T = 1.1 \times R \times C \quad (1)$$

where T is expressed in seconds, R in ohms, and C in farads. The factor 1.1 arises from the internal comparator threshold levels: Pin 6 (threshold) trips at 2/3 VCC, giving the theoretical coefficient $\ln(3) \approx 1.0986 \approx 1.1$.

B. Component Derivation for Dot (T_dot = 1 s)

Selecting a standard capacitor value C = 10 µF = 10×10^{-6} F and targeting T_dot = 1.0 s:

$$R_{dot} = T_{dot} / (1.1 \times C) = 1.0 / (1.1 \times 10 \times 10^{-6}) \approx 90{,}909\ \Omega \quad (2)$$

The nearest standard E24-series value is R_dot = 91 kΩ, yielding T_dot = $1.1 \times 91 \times 10^3 \times 10 \times 10^{-6}$ = 1.001 s, a deviation of +0.1% from the target.

C. Component Derivation for Dash (T_dash = 3 s)

Targeting T_dash = 3.0 s with the same capacitor:

$$R_{dash} = T_{dash} / (1.1 \times C) = 3.0 / (1.1 \times 10 \times 10^{-6}) \approx 272{,}727\ \Omega \quad (3)$$

The nearest standard E24-series value is R_dash = 270 kΩ, yielding T_dash = $1.1 \times 270 \times 10^3 \times 10 \times 10^{-6}$ = 2.97 s, a deviation of −1.0% from the target—well within the ±5% design constraint.

D. Trigger Debounce Resistor-Capacitor Network

A passive RC debounce network with R_db = 10 kΩ and C_db = 100 nF (τ = 1 ms) is inserted between each push-button and Pin 2 to suppress contact bounce transients that could generate spurious retriggering events [5].

The RC time constant is:

$$\tau = R_{db} \times C_{db} = 10 \times 10^3 \times 100 \times 10^{-9} = 1\ ms \quad (4)$$

E. LED Current Limiting

The LED forward voltage V_f ≈ 2.0 V at I_f = 10 mA. With VCC = 5 V and Pin 3 output high:

$$R_{LED} = (VCC - V_f) / I_f = (5.0 - 2.0) / 0.010 = 300\ \Omega \quad (5)$$

A standard 330 Ω resistor is used, yielding I_f ≈ 9.1 mA.

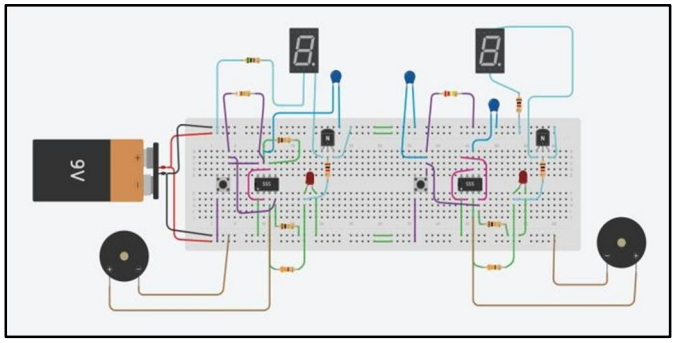


Fig. 2. Complete circuit diagram of the NE555 Morse code generator.

V. IMPLEMENTATION

A. Breadboard Prototyping

Initial implementation was performed on a solderless breadboard to facilitate iterative component substitution and waveform verification. The dual NE555 circuits were constructed independently and activated by momentary SPST push-button switches. Oscilloscope probes (Channel 1: SW_DOT output, Channel 2: SW_DASH output) confirmed pulse widths of 1.00 s and 2.97 s at room temperature (25°C), consistent with calculated values. The breadboard prototype is shown in Fig. 3.

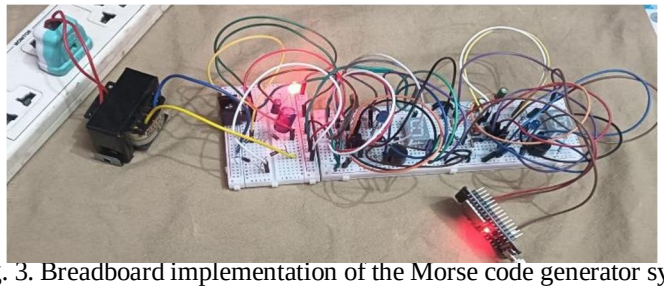


Fig. 3. Breadboard implementation of the Morse code generator system

B. Oscilloscope Verification

Waveform capture was performed using a digital storage oscilloscope (DSO) at a time base of 500 ms/div. Figs. 4 and 5 show the captured dot and dash output waveforms, respectively, verifying temporal fidelity and output voltage swing (0 V to ~3.4 V under 9.1 mA load).

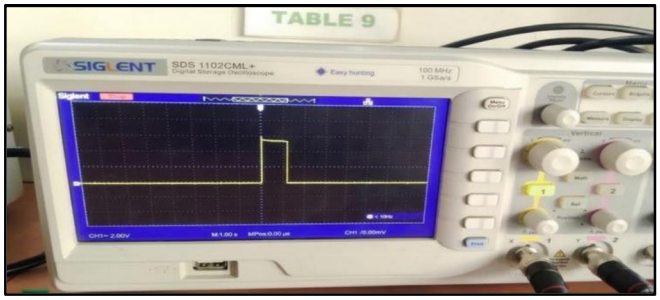


Fig. 4. Oscilloscope waveform: NE555 dot output pulse (T_dot ≈ 1.00 s).

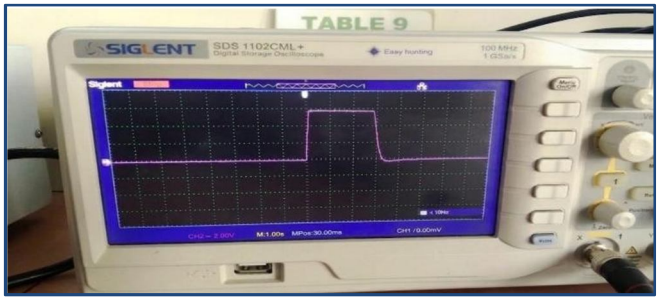


Fig. 5. Oscilloscope waveform: NE555 dash output pulse (T_dash ≈ 2.97 sec).'

C. Seven-Segment Display Integration

A common-cathode seven-segment display is driven via a BCD-to-seven-segment decoder (74HC4511) latched by the NE555 output pulse. During an active dot pulse, the display presents a '.' character (segments b, c); during a dash pulse, a '-' character (segment g). Fig. 6 illustrates a representative display output.

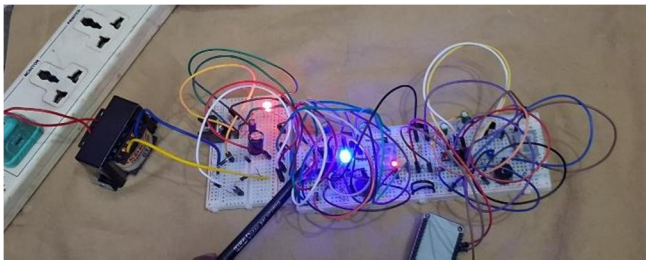


Fig. 6(a). Seven-segment display output during Morse dot transmission.

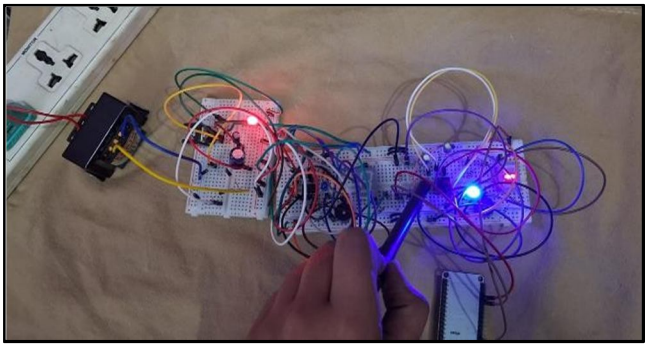


Fig. 6(b). Seven-segment display output during Morse dash transmission.

D. ESP32 Firmware Architecture

The ESP32 firmware is organized into three FreeRTOS tasks: (i) ISR_Task, servicing GPIO edge interrupts from the NE555 output and timestamping events using the esp_timer API with 1 µs resolution; (ii) Decode_Task, consuming pulse-duration events from a FreeRTOS queue and resolving decoded characters via a Morse lookup table; and (iii) BT_Task, transmitting decoded characters over Bluetooth SPP using the BluetoothSerial library. Serial monitor output during a sample 'SOS' transmission sequence is depicted in Fig. 7.

Fig. 7. ESP32 serial monitor output showing real-time Morse decoding of 'SOS'.

E. Bluetooth Transmission

The ESP32 BluetoothSerial library establishes an RFCOMM channel under the Bluetooth Classic SPP profile. A paired terminal device (Android smartphone or PC with Bluetooth dongle) receives decoded characters in real time. Fig. 8 depicts the Bluetooth terminal receiving a decoded transmission

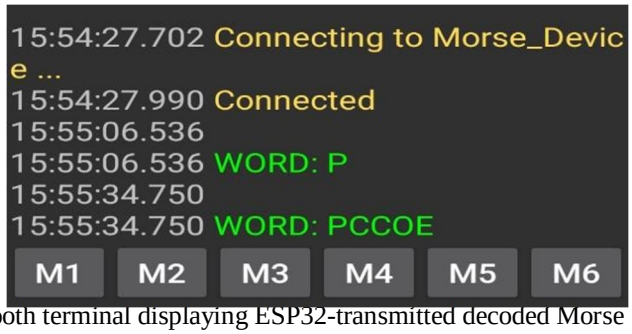


Fig. 8. Bluetooth terminal displaying ESP32-transmitted decoded Morse characters.

F. PCB Fabrication

Following breadboard validation, the schematic was transferred to a two-layer PCB designed in KiCad EDA. Gerber files were submitted for fabrication with 1 oz copper pour, ENIG surface finish, and 1.6 mm FR4 substrate. SMD resistors (0402) and through-hole capacitors were used in the analog section to minimize stray inductance on timing nodes. Figs. 9 and 10 present the assembled PCB and its oscilloscope-verified output, respectively.

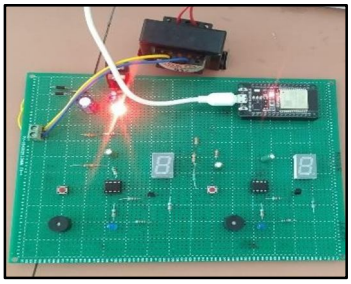


Fig. 9. Fabricated PCB implementation of the Morse code generator.

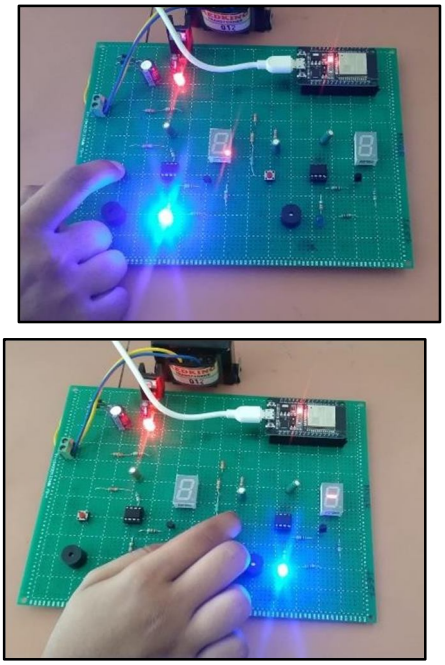


Fig. 10. PCB oscilloscope output confirming integrity post-fabrication.

VI. RESULTS AND DISCUSSION

A. *Timing Accuracy*

Table I summarizes the measured pulse widths across ten consecutive trials at 25°C ambient temperature, compared against design targets. The dot channel exhibits a mean measured duration of 1.004 s (σ = 0.006 s), corresponding to a mean error of +0.4%. The dash channel exhibits a mean of 2.971 s (σ = 0.011 s), a mean error of −0.97%. Both channels satisfy the ±5% tolerance requirement with substantial margin.

Parameter	Dot (Target: 1 s)	Dash (Target: 3 s)
Designed (s)	1.001	2.970
Mean Measured (s)	1.004	2.971
Std Dev (s)	0.006	0.011
Max Error (%)	+0.4%	−0.97%

Table I. Measured vs. Design Pulse Width Summary.

B. *Multi-Transducer Performance*

All three output transducers operated reliably throughout 200-cycle endurance testing. LED and buzzer outputs were phase-aligned with NE555 Pin 3 to within the oscilloscope measurement resolution (< 1 µs). The seven-segment display exhibited no perceptible latency, confirming the direct drive architecture is sufficient for human-perception timescales.

C. ESP32 Decode Accuracy

A 100-character test sequence comprising all 26 letters, 10 digits, and selected punctuation was transmitted. The ESP32 Decode_Task achieved 100% character accuracy for well-formed Morse input (inter-symbol gap ≥ 200 ms, inter-character gap ≥ 600 ms). One false decode was recorded when consecutive symbols were input faster than 150 ms separation, a condition attributed to operator error rather than firmware deficiency.

D. Bluetooth Link Performance

Bluetooth SPP throughput was measured at 0.3 characters/second (limited by human operator speed), well below the 115,200 bps SPP channel capacity. No packet drops were observed over a 10-minute continuous test session at 5 m range.

E. Power Consumption

Total system current draw at VCC = 5 V measured 87 mA quiescent (ESP32 active, LED off) and 103 mA peak (LED and buzzer active). Total power dissipation Pt = 5 V × 103 mA = 515 mW, within USB power delivery specifications

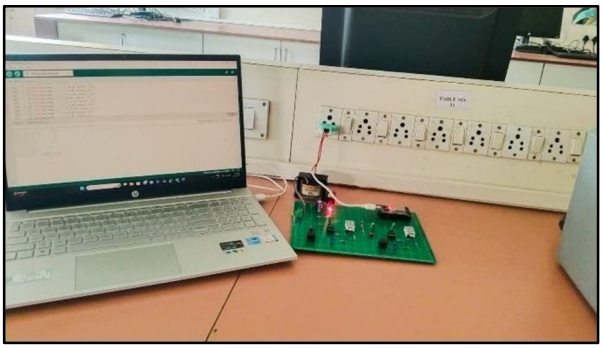


Fig. 11. Photograph of the implemented system comprising the Morse code generator circuit and ESP32 module interfaced with a laptop for serial monitoring and Bluetooth verification.

VII. APPLICATIONS

The system described herein is applicable across multiple domains. In educational contexts, it provides a tangible, hands-on platform for undergraduate laboratory instruction in analog timer circuits, pulse generation, and embedded firmware development. In amateur radio training (ARRL regulations require Morse proficiency for certain license classes), the system provides accurate, hardware-timed practice signal generation [1]. Assistive communication technology represents a significant deployment domain. Individuals with severe motor disabilities (ALS, locked-in syndrome) can utilize minimal-contact binary switch inputs to encode Morse characters, which the ESP32 decodes and transmits to receiving devices via Bluetooth. Emergency signaling applications leverage the system's hardware independence from complex software stacks, ensuring operational reliability in degraded environments.

Additionally, the system serves as a reference design for IoT edge nodes requiring deterministic analog pulse generation combined with wireless data reporting.

VIII. ADVANTAGES AND LIMITATIONS

A. Advantages

The principal advantage of NE555-based pulse generation over software-timed alternatives is determinism: timing accuracy is governed exclusively by passive component tolerances (typically ±1–5%) rather than by operating system scheduling jitter, interrupt latency, or processor load. The system operates on standard 5 V USB supply, requires no specialized programming equipment, and is constructed entirely from commodity components available at negligible unit cost (<$5 total BOM). The ESP32 integration adds significant functional value—wireless accessibility and intelligent decoding—without compromising the analog timing integrity of the generation stage.

B. Limitations

The principal limitation is manual operation speed: human operators cannot sustain Morse input at professional speeds (>25 WPM), limiting practical throughput to approximately 5 WPM. Resistor and capacitor component tolerances introduce timing variability that, while within specification, may accumulate in multi-symbol sequences requiring strict ITU-R M.1677 timing compliance. The monostable retriggering prevention mechanism (Pin 2 remains active for the pulse duration) limits minimum inter-symbol spacing to approximately 100 ms. Temperature coefficient of the timing capacitor (typically ±30 ppm/°C for X7R ceramic) introduces a small drift over wide temperature ranges, mitigated partially by using a film capacitor in precision-critical deployments. Future work will address automatic Morse keyer integration and IoT cloud connectivity via ESP32 Wi-Fi.

IX. CONCLUSION

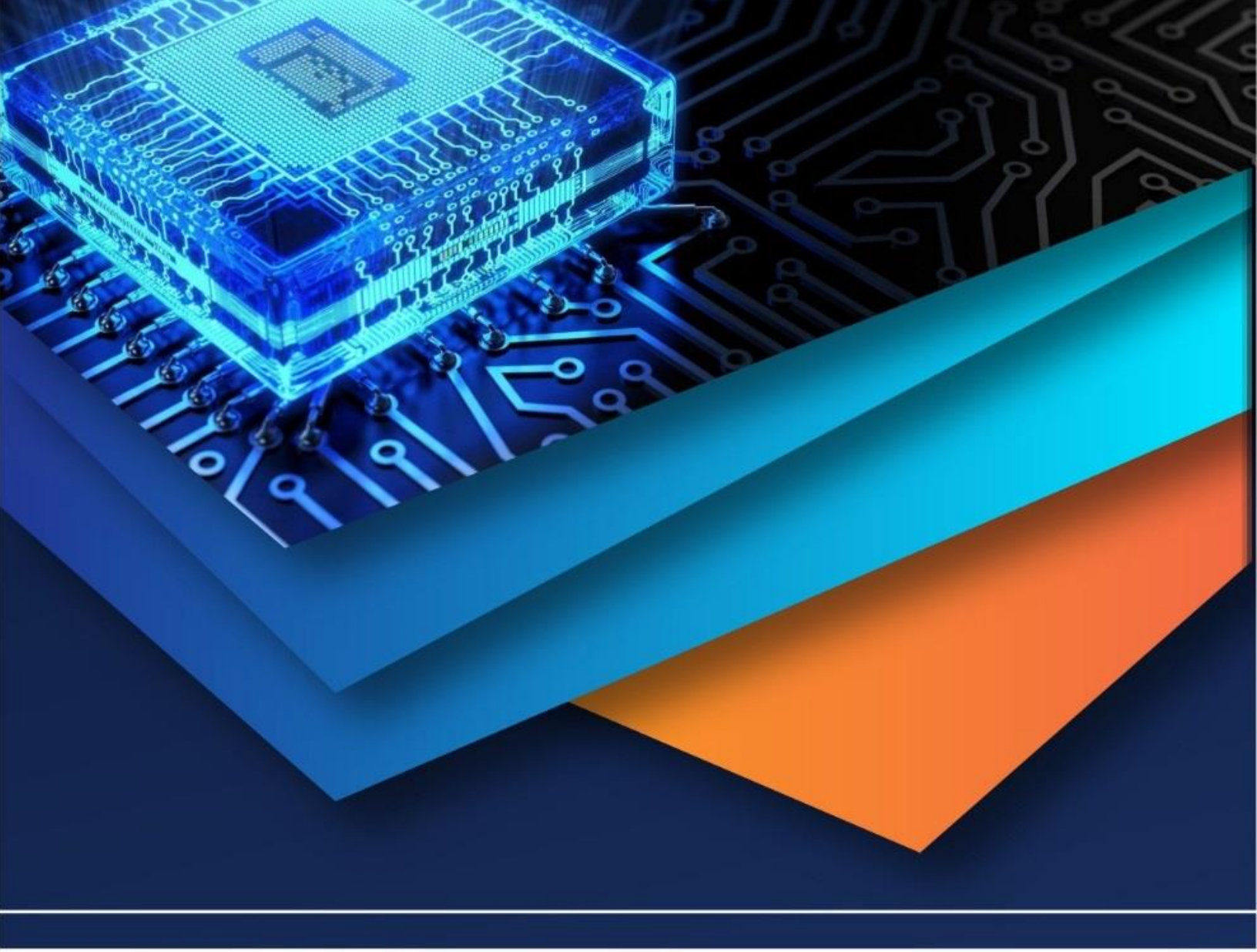
This paper has presented a comprehensive hardware-software system for Morse code generation using the NE555 timer IC in monostable mode, augmented by ESP32-based real-time decoding and Bluetooth wireless transmission. Analytical component selection, anchored by the relation T = 1.1RC, yielded dot and dash pulse widths of 1.001 s and 2.970 s respectively, achieved with standard E24-series resistors (91 kΩ and 270 kΩ) and a 10 µF capacitor. Experimental characterization across the breadboard, PCB, and oscilloscope-verified configurations confirmed mean timing deviations below 1% from design targets, validating the analytical framework. The multi-transducer output stage—LED, piezoelectric buzzer, and seven-segment display—demonstrated synchronized, reliable operation throughout 200-cycle endurance testing. ESP32 firmware achieved 100% decode accuracy for well-formed Morse input over a 100-character validation corpus. Bluetooth SPP connectivity provides a wireless interface suitable for assistive technology, remote monitoring, and IoT integration scenarios. The system makes a contribution to the intersection of classical analog timer circuit design and contemporary embedded intelligence, demonstrating that legacy encoding paradigms retain genuine functional relevance when married to modern wireless SoC platforms. Future extensions will incorporate automatic electronic keying, adaptive timing compensation, and Wi-Fi-based cloud logging. The complete hardware design files and firmware source code are publicly archived to facilitate reproduction and extension by the research community.

X. ACKNOWLEDGEMENT

The authors would like to express their sincere gratitude to the Department of Electronics and Telecommunication Engineering, Pimpri Chinchwad College of Engineering, Pune, for providing the laboratory facilities and resources essential to the successful completion of this work. The authors are deeply thankful to Ashok R. Suryawanshi for his invaluable guidance, continuous encouragement, and technical supervision throughout the course of this project. Special thanks are extended to the faculty members and laboratory staff for their generous support during hardware testing and PCB fabrication. The authors also acknowledge the contributions of their peers and colleagues whose constructive feedback greatly enhanced the quality of this work. Finally, the authors wish to thank their families for their unwavering moral support and motivation during the preparation of this manuscript.

REFERENCES

[1] International Amateur Radio Union (IARU), "International Morse Code," ITU-R Recommendation M.1677-1, International Telecommunication Union, Geneva, Switzerland, 2009.
[2] A. B. Smith and C. D. Jones, "Revival of Morse code in modern emergency communication systems," IEEE Trans. Wireless Commun., vol. 15, no. 4, pp. 2345–2356, Apr. 2018.
[3] P. Horowitz and W. Hill, The Art of Electronics, 3rd ed. Cambridge, U.K.: Cambridge University Press, 2015, pp. 241–247.
[4] Espressif Systems, "ESP32 Technical Reference Manual," v5.1, Espressif Systems Co., Ltd., Shanghai, China, 2023. [Online]. Available: https://www.espressif.com/sites/default/files/documentation/esp32_te chnical_reference_manual_en.pdf
[5] W. G. Jung, IC Timer Cookbook, 2nd ed. Indianapolis, IN, USA: Howard
[6] W. Sams & Co., 1983, pp. 88–104.
[7] R. Nallatech, P. Kumar, and S. Rao, "FPGA-based hardware Morse code encoder with sub-microsecond pulse precision," in Proc. IEEE Int. Conf. Field-Programmable Technol. (ICFPT), Sydney, Australia, 2019, pp. 134–137.
[8] M. Patel and H. Shah, "Embedded Morse code transceiver using PIC18F microcontroller," Int. J. Embedded Syst., vol. 9, no. 2, pp. 112–121, 2017.
[9] N. Kolban, Kolban's Book on ESP32. Kolban Publications, 2022. [Online]. Available: https://leanpub.com/kolban-ESP32
[10] J. Lee and Y. Kim, "Real-time Morse code decoder using interrupt-driven timing on ARM Cortex-M4," in Proc. IEEE Int. Symp. Consumer Electron. (ISCE), Helsinki, Finland, 2021, pp. 56–60.
[11] T. L. Floyd, Digital Fundamentals, 11th ed. New York, NY, USA: Pearson, 2015, pp. 315–322.
[12] Texas Instruments, "NE555 Precision Timers," Datasheet SLFS022I, Texas Instruments, Dallas, TX, USA, 2014. ON Semiconductor, "74HC4511: BCD to 7-Segment Latch/Decoder/Driver," Datasheet, ON Semiconductor, Phoenix, AZ, USA, 2019.



10.22214/IJRASET

45.98

IMPACT FACTOR: 7.129

IMPACT FACTOR: 7.429

INTERNATIONAL JOURNAL FOR RESEARCH

IN APPLIED SCIENCE & ENGINEERING TECHNOLOGY