© JUN 2026 | IRE Journals | Volume 9 Issue 12 | ISSN: 2456-8880
DOI: https://doi.org/10.64388/IREV9I12-1718731

Cost-Effective IoT-Based Multi-Testing Device Using ESP32

THILAKKUMAR S[1], M. KANNAN[2]
[1] PG Student, Department of CSA, SCSVMV, Kanchipuram, India
[2] Assistant Professor, Department of CSA, SCSVMV, Kanchipuram, India

Abstract- ***This project presents a cost-effective IoT-based multi-testing device using the ESP32, where the ESP32 microcontroller serves as the core unit with built-in Wi-Fi capability for network monitoring. The PN532 NFC module is utilized to read, write, and emulate NFC cards for authentication, ensuring that access is granted only to authorized cards stored in the system. The ESP32 also scans nearby Wi-Fi networks to monitor signal strength and connectivity status. In addition, the IR module is used to capture infrared signals, which are stored on an SD card and can be replayed to control IR-based devices such as televisions and air conditioners. TFT touch display is integrated into the system to provide a user-friendly interface, enabling users to easily interact with the device, monitor system status, and control various functions in real time. This integrated system offers a compact, cost-effective, and versatile solution by combining network monitoring, access control, and remote device operation within a single IoT device.***

Keywords: IoT, ESP32, NFC, PN532, Infrared Communication, Wi-Fi Monitoring, Embedded Systems, Access Control, Remote Device Control, TFT Display, Smart Systems, Automation.

I. INTRODUCTION

In many embedded system testing environments, different tools are required to analyze NFC systems, monitor Wi-Fi networks, and test IR-based devices. Using separate tools increases both cost and complexity, and it also makes it difficult to maintain a unified record of test results.

To address this problem, a compact multi-testing device was developed using the ESP32 microcontroller. The goal was to combine multiple testing functionalities into a single portable system that can be easily used without additional equipment The ESP32 was selected due to its built-in Wi-Fi capability, sufficient processing power, and flexible GPIO support.

The system integrates three main modules: NFC, Wi-Fi analysis, and IR signal processing. A touchscreen interface allows the user to access all features directly, while an SD card is used to store logs and captured data.

This paper presents the design, implementation, and testing of the device, along with the challenges encountered and the solutions applied during development

II. ACRONYMS AND ABBREVIATIONS

TABLE I - The following abbreviations are used throughout this paper

Acronym	Full Form
IoT	Internet of Things
NFC	Near Field Communication
RFID	Radio Frequency Identification
IR	Infrared
RSSI	Received Signal Strength Indicator
UID	Unique Identifier
TFT	Thin-Film Transistor (display)
SD	Secure Digital (card)
CSV	Comma-Separated Values

© JUN 2026 | IRE Journals | Volume 9 Issue 12 | ISSN: 2456-8880
DOI: https://doi.org/10.64388/IREV9I12-1718731

III. LITERATURE SURVEY

Babiuch, Foltynek, and Smutny [1] evaluated the ESP32 across three programming frameworks — Arduino Core, ESP-IDF, and MicroPython — demonstrating reliable sensor-coupled display output. Their work confirmed the ESP32's suitability for display-coupled sensor acquisition; however, each prototype addressed a single sensing function. No attempt was made to co-schedule NFC authentication, wireless scanning, and IR signal processing on the same platform.

Hercog et al. [3] deployed multi-sensor ESP32 prototypes in a university mechatronics curriculum, with students logging physical sensor data to SD cards. Their work validated SD-based local logging as a dependable embedded storage strategy, which directly informed the storage architecture of the proposed device. However, their systems addressed only physical quantities such as weight and temperature, with no treatment of wireless protocol analysis or signal replay.

Patil, Shinde, and Devmore [2] demonstrated a tightly engineered single-function ESP32 temperature control system achieving 100% control accuracy with deep-sleep power management. While exemplary as a single-purpose design, their work offered no mechanism for NFC authentication, wireless diagnostics, or IR signal learning — the combined capabilities central to this paper.

In the NFC domain, most published access-control systems position a Raspberry Pi as the host processor with the PN532 as a peripheral [6]. Although effective, this architecture is unnecessarily expensive and physically bulky for a portable handheld tester. IR blasters in the literature typically appear as standalone single-function devices with neither storage nor interface display. Dedicated Wi-Fi analysers remain software applications running on general-purpose smartphones rather than purpose-built hardware instruments.

A review of the literature found no published work combining NFC read-write-emulation, passive Wi-Fi analysis, and IR capture-replay in a single embedded device with a touchscreen and local persistent storage. The present work addresses this gap.

IV. PROPOSED SYSTEM

The proposed device integrates three testing modules — NFC, Wi-Fi analysis with port scanning, and IR signal capture and replay — on a single ESP32 platform. A 2.8-inch resistive touchscreen provides a fully standalone user interface, and all data are stored locally on a micro-SD card. Fig. 1 illustrates the hardware block diagram; Fig. 2 depicts the system data flow.

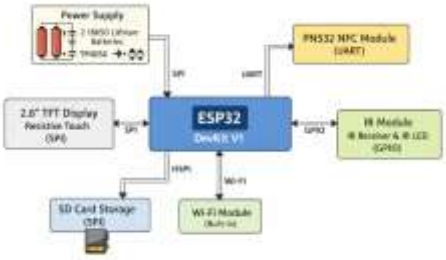

Fig. 1. Hardware Block Diagram of the Multi-Testing Device

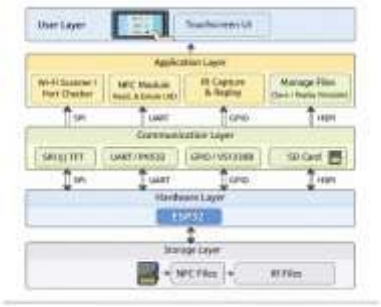

Fig. 2. System Architecture and Data Flow Diagram

A. NFC / RFID Module

The PN532 V3 module connects to the ESP32 via UART and operates at 13.56 MHz in ISO 14443-A mode. The firmware implements three operating modes. In Read mode, the device polls for nearby cards, extracts the UID, displays it on the TFT, and logs the event with a timestamp to nfc_log.txt on the SD card. In Write mode, an NDEF text or URL record is pushed onto a blank NFC tag. In Emulation mode, a previously captured UID is loaded from the SD card, the PN532 is configured to present that UID to an external reader, and the reader's accept or reject

© JUN 2026 | IRE Journals | Volume 9 Issue 12 | ISSN: 2456-8880
DOI: https://doi.org/10.64388/IREV9I12-1718731

response is observed — a standard and legitimate workflow for access-control system validation.

B. Wi-Fi Analyser and Open Port Scanner

The ESP32's integrated 802.11 b/g/n radio performs all wireless scanning with no additional hardware. WiFi.scanNetworksAsync() executes in the background every few seconds, collecting SSID, BSSID, channel, RSSI (dBm), and security type for each visible access point. Results are sorted by signal strength and presented as a colour-coded scrollable list on the TFT. A port scanner performs TCP connect checks on the 20 most commonly used ports against the gateway address of a selected network, with a 300 ms timeout per port. Scan results are appended to a timestamped CSV file in the wifi_scans/ directory on the SD card.

C. IR Signal Capture and Replay

The VS1838B receiver demodulates the 38 kHz carrier internally, delivering a clean TTL pulse train to the ESP32 GPIO. The IRremoteESP8266 library decodes more than 60 IR protocols. In Capture mode, the user points any remote at the receiver, the decoded protocol and 32-bit hex value appear on the TFT, and a label is entered using an on-screen keyboard. Each record is serialised as a JSON entry in ir_library.json on the SD card. Unrecognised signals are stored as raw pulse-duration arrays and can be replayed identically. In Replay mode, the SD-resident library loads as a scrollable list; selecting an entry transmits the signal via the 940 nm IR LED.

D. Touchscreen Interface

The home screen presents four tiles: NFC, Wi-Fi, IR, and Settings. Each tile transitions to the corresponding module panel, which displays only the information relevant to that function. The resistive touch interface makes the device fully self-contained — no external computer, serial terminal, or companion application is required in the field.

E. Local SD Card Data Logging

All session data are stored locally on a FAT32-formatted micro-SD card. The logging subsystem maintains three data structures: nfc_log.txt (tab-delimited records of each card event with UTC timestamp, UID, and access result), a wifi_scans/ folder containing one CSV per scan session with full network details and port-scan results, and ir_library.json containing the user-labelled IR command library. Data persist across power cycles and are immediately accessible for offline review without any network dependency. This architecture ensures the device operates correctly in network-denied or air-gapped environments.

V. SYSTEM ARCHITECTURE

The firmware of the proposed device is implemented as a five-state state machine, consisting of HOME, NFC_MODE, WIFI_MODE, IR_MODE, and SETTINGS. State transitions are triggered by touch inputs detected through the resistive touchscreen interface. Each state controls which functional module is active and receives processor time within the main execution loop of the ESP32 DevKit V1.

A. Circuit Diagram and Hardware Connections

Fig. 3 presents the complete circuit diagram of the proposed device. The hardware connections are described below for each interface.

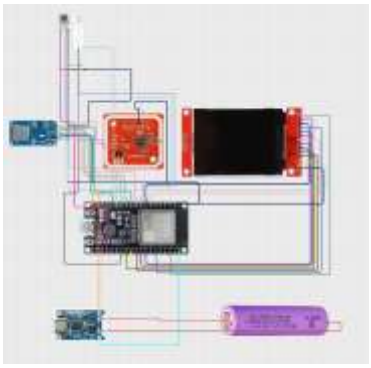

Fig. 3. Circuit Diagram and Hardware Connections of the Proposed Device

TABLE II – HARDWARE PIN CONFIG

Pin / Module	Connection / ESP32 Pin	Description	Pin / Module	Connection / ESP32 Pin	Connection / ESP32 Pin
TFT MOSI	GPIO23	SPI data (shared with	SD SCK	GPIO14	SD card clock

© JUN 2026 | IRE Journals | Volume 9 Issue 12 | ISSN: 2456-8880
DOI: https://doi.org/10.64388/IREV9I12-1718731

		touch)			
TFT MISO	GPIO19	SPI data (shared with touch)	SD CS	GPIO27	SD card chip selects
TFT SCK	GPIO18	SPI clock	PN532 TX	GPIO26	NFC → ESP32 RX2
TFT CS	GPIO17	TFT chip select	PN532 RX	GPIO25	ESP32 TX2 → NFC
TFT DC	GPIO16	Data/command control	IR Receiver OUT	GPIO15	IR signal input
TFT RST	GPIO5	Display reset	IR LED	GPIO4	IR signal output
TFT LED	GPIO32	Backlight (PWM control)	TFT / Modules VCC	3.3V	Power supply
TOUCH CS	GPIO21	Touch chip select	All Modules GND	GND	Common ground
TOUCH IRQ	GPIO22	Touch interrupt	TP4056 OUT+	VIN / 5V	Power to ESP32
SD MOSI	GPIO12	SD card data	TP4056 OUT–	GND	Ground connection

VI. HARDWARE COMPONENTS

A. ESP32 DevKit V1

The ESP32 DevKit V1 with the WROOM-32 module provides a dual-core Xtensa LX6 processor at 240 MHz, 520 KB SRAM, 4 MB flash, and a fully integrated 802.11 b/g/n Wi-Fi radio. Its multiple hardware SPI and UART controllers allowed independent bus assignment to the TFT, SD card, and NFC module without software emulation. GPIO pins 34–39 are input-only; this constraint was identified during development and respected in all pin assignments. A pinout reference must be consulted before wiring any ESP32 project.

B. PN532 V3 NFC Module

The PN532 from NXP Semiconductors supports MIFARE Classic, MIFARE Ultralight, and ISO 14443 Type A/B — the card families used in the majority of real-world access-control deployments. The V3 revision (red board with DIP switches) allows UART, I2C, or SPI selection. UART mode was chosen after the earlier PN532 V1 module (blue board) produced unreliable SPI reads at 3.3 V logic levels, including truncated UIDs and no-response conditions. The V3 module in UART mode delivered consistent results across all 100 test reads and is strongly recommended for ESP32-based NFC projects.

C. VS1838B Infrared Receiver

The VS1838B is a pre-tuned 38 kHz demodulating IR receiver that filters the carrier internally and presents a clean TTL output to the ESP32 GPIO. A 10 kΩ resistor on the signal line and a 100 µF bypass capacitor on the supply rail were added after observing spurious false triggers on the initial breadboard prototype. These two passive components eliminated all false detections. The VS1838B is pin-compatible with the TSOP38238.

D. IR LED Transmitter

A standard 5 mm 940 nm IR LED driven directly from GPIO 26 through a 22 Ω resistor at approximately 12 mA provides reliable transmission to 10 metres in typical indoor conditions. A transistor driver was evaluated to extend range but offered no practical improvement for the intended use case. Direct GPIO drive was retained for simplicity.

E. 2.8-Inch SPI TFT Display

The ILI9341 display with XPT2046 resistive touch controller provides 240×320 pixels at 16-bit colour over the VSPI bus. The TFT_eSPI library, which exploits DMA transfers on the ESP32, was selected after the Adafruit GFX library produced visibly

sluggish panel transitions. The TFT_eSPI library delivered smooth, responsive rendering for all UI elements. The module's backlight PWM was found unreliable on standard GPIO outputs and was therefore connected directly to 3.3 V.

F. Micro SD Card Module

The SPI-mode micro-SD module stores all session data on a FAT32-formatted card. Two integration issues were resolved during development. First, assigning the SD card to a completely separate SPI bus (HSPI) eliminated all write failures that occurred when the TFT and SD module shared the same MISO, MOSI, and CLK lines. Second, providing 5 V via the MT3608 boost converter resolved initialisation failures observed when the SD module was powered from the 3.3 V rail.

G. Power Supply

Two Li-ion 18650 cells at 3.7 V nominal, managed by a TP4054 Type-C charging module, provide the primary energy source. An MT3608 DC-DC boost converter steps the battery voltage up to a regulated 5 V for the SD card module. The ESP32's onboard AMS1117 LDO regulator derives 3.3 V for all logic-level peripherals. Measured battery life during normal operation — touchscreen active, periodic Wi-Fi scans, occasional NFC reads — is 3 to 4 hours per charge cycle.

VII. SOFTWARE IMPLEMENTATION

A. Development Environment

All firmware is written in C++ using Arduino IDE 2.x with the Espressif ESP32 board support package. The codebase is organised into separate translation units — nfc.cpp, ir.cpp, wifi_scan.cpp, display.cpp, and sdcard.cpp — with main.cpp hosting the state machine and touch-event dispatcher. External libraries used: Adafruit PN532 (NFC UART driver), TFT_eSPI (display with DMA), IRremoteESP8266 (IR decode and transmit), ArduinoJson (SD serialisation), and the standard ESP32 SD library.

B. NFC Firmware

The PN532 is initialised in ISO 14443-A mode over UART2 (GPIO 16/17). The firmware continuously polls readPassiveTargetID() when NFC_MODE is active. Detected UIDs are compared byte-by-byte against a whitelist stored in the ESP32's NVS (non-volatile storage) using the Preferences library. Access-Granted or Access-Denied results are displayed for two seconds, then the display returns to the waiting state. Each event is written to nfc_log.txt on the SD card with a UTC-formatted timestamp. For card emulation, the firmware reads a UID from an SD card record, configures the PN532 in card-emulation mode, and holds that state until the user navigates away.

C. Wi-Fi and Port Scan Firmware

WiFi.scanNetworksAsync() executes background scans without blocking the touchscreen. On completion, a struct array sorted by RSSI is rendered as a colour-coded scrollable list. TCP connect checks on 20 commonly used ports (80, 443, 22, 21, 23, 25, 53, 110, 143, 3389, etc.) are performed against the detected gateway address with a 300 ms timeout per port. Results are appended to a timestamped CSV file on the SD card.

D. IR Firmware

IRrecv monitors GPIO 35 and triggers a decode callback on each complete IR frame. The IRremoteESP8266 library identifies the protocol and 32-bit value. The user assigns a label via the on-screen keyboard; the record is serialised using ArduinoJson and appended to ir_library.json on the SD card. Unrecognised signals are stored as raw pulse-duration arrays. In Replay mode, IRsend.send() is called for named protocols and IRsend.sendRaw() for raw captures. All 47 tested commands replayed without error.

VIII. WORKING METHODOLOGY

The firmware executes a well-defined sequential startup sequence and predictable runtime behaviour, described below.

Power-up Sequence: The SD card is initialised first. If absent, a TFT warning is displayed and the device continues in RAM-only mode with reduced logging. The TFT and PN532 are initialised next, followed by the IR receiver, IR LED, and the Wi-Fi radio (placed in station mode for passive scanning). The home screen then renders with four module tiles.

NFC Mode: The PN532 polls for cards. When a card enters the 3–5 cm antenna range, the UID is read in approximately 50 ms, the whitelist is checked, and the result is displayed in green (Granted) or red (Denied) for two seconds. The event is logged to the SD card. In Emulation sub-mode, the selected UID from the SD card library is loaded, the PN532 assumes that card identity, and the system waits for an external reader to query it — enabling access-control system verification without carrying physical cards.

Wi-Fi Mode: Entering this mode initiates an immediate background scan. A progress indicator is displayed during the 2–4 second scan window. The network list appears sorted by signal strength with colour-coded RSSI indicators. Tapping a network reveals full detail and offers a port scan option. Tapping Export appends the current scan to a timestamped CSV in the wifi_scans/ folder.

IR Mode: Two buttons are presented: Capture and Replay. In Capture, the screen prompts the user to point a remote at the VS1838B receiver. Upon signal detection, the decoded protocol and hex value appear and the user labels the command. In Replay, the ir_library.json is loaded from the SD card and displayed as a scrollable list. Tapping any entry transmits the IR signal and briefly displays 'Sent' on the TFT.

Settings: Accessible from the home screen. Covers NFC whitelist management (add or remove UIDs by scanning a card), Wi-Fi scan interval adjustment, and system diagnostics including SD card free space and firmware version. All changes take effect immediately without a reboot.

IX. RESULTS AND DISCUSSION

A. NFC Performance

One hundred card-tap tests were conducted using three authorised cards and four unauthorised cards. The system produced zero false positives and zero false negatives, yielding 100% authentication accuracy. Average time from card detection to result display was 78 ms. NDEF write operations succeeded on MIFARE Ultralight and MIFARE Classic 1K tags for payloads up to 144 bytes. Detection range was 3–5 cm with the card held flat; cards presented at angles greater than 30° reduced the reliable range to approximately 2 cm — a characteristic worth communicating to end users. UID emulation was verified against two commercial NFC readers, and the emulated credential was accepted identically to the original physical card in both cases.

B. Wi-Fi and Port Scan Performance

In a department building with 12 visible access points, the scanner detected all 12 consistently across 20 scan cycles. RSSI readings matched a reference Android Wi-Fi Analyzer application to within 1–2 dBm in 90% of tests. A mobile hotspot broadcasting a duplicate SSID was detected within the next scan cycle after activation. Port scanning on a local router correctly reported ports 80 and 443 as open and all others as closed, consistent with an independent Nmap reference scan.

C. IR Capture and Replay Performance

Forty-seven IR commands were captured and tested across four appliances: a Samsung television (NEC protocol), an LG television (NEC protocol), a Sony soundbar (SIRC protocol), and a Daikin air conditioner (Daikin proprietary protocol). All 47 commands replayed with 100% appliance response at distances up to 10 metres. The SD-resident ir_library.json survived 15 power cycles without data corruption. A Mitsubishi air conditioner unit using an encrypted proprietary protocol could not be identified by the decode library; however, its signals were stored as raw pulse arrays and replayed successfully.

D. SD Card Logging Performance

All data logging operations completed with negligible user-perceptible latency. NFC log writes averaged under 12 ms per entry. Wi-Fi CSV append operations for a 12-network scan completed in under 35 ms. The IR JSON library, containing 47 entries with raw arrays, loaded in under 80 ms at boot. No SD card write errors or file system corruption was observed across the full test campaign. The logging subsystem performed correctly with FAT32-formatted cards from two different manufacturers at capacities of 8 GB and 16 GB.

E. Frequency Support and Capability Comparison

Table III presents a comparison of the radio-frequency capabilities supported by the proposed device relative to the Flipper Zero multi-tool and dedicated single-function instruments. The proposed device operates across three distinct communication ranges: NFC at 13.65 MHz (short-range, contactless communication within a few centimetres),

IR at 38 kHz (line-of-sight communication up to 10 metres), and Wi-Fi at 2.4 GHz (medium-range wireless communication up to ten of metres).

A significant and acknowledged limitation is the absence of Sub-GHz RF support (typically 315 MHz, 433 MHz, or 868 MHz), which the Flipper Zero addresses via its CC1101 transceiver. Sub-GHz capability would be required for testing key fobs, garage door openers, and older ISM-band sensors — functions that are outside the scope of the present device but represent a clear avenue for future expansion through the addition of a CC1101 or similar module. The ESP32's built-in BLE radio is available but not yet exposed through the device firmware; BLE scanning is planned for a future firmware revision.

TABLE III — Frequency Support and Capability Comparison

Technology	Frequency	Proposed Device	Flipper Zero	Dedicated Tool
NFC / RFID	13.56 MHz	Yes	Yes	Yes (ACR122U)
IR Remote Control	38 kHz carrier	Yes	Yes	Yes (standalone blaster)
Wi-Fi 2.4 GHz (IEEE 802.11 b/g/n)	2.4 GHz	Yes (scan only)	No	Dedicated analyzer
Sub-GHz RF (315/433/868 MHz)	315–868 MHz	No (limitation)	Yes (CC1101)	Yes (CC1101)	Dedicated SDR
Bluetooth Low Energy	2.4 GHz	Yes* (ESP32 built-in)	Yes	Separate tool

* BLE is available via the ESP32's integrated radio but is not yet activated in the current firmware. Sub-GHz RF is not supported and would require an external CC1101 module.

F. Cost Summary and Market Comparison

Table IV presents the complete bill of materials including the Li-ion battery cells, TP4054 charging module, and MT3608 boost converter that were absent from earlier cost estimates. The corrected total material cost is ₹2,160. Table V compares the proposed device against the market cost of equivalent dedicated instruments. Table VI benchmarks the device against the Flipper Zero and other dedicated instruments across key capability dimensions.

G. Operational Range Comparison

The effective operating range of each module was measured during testing.

- NFC (PN532 Module): Reliable detection range of 3–5 cm for ISO 14443-A cards. Commercial NFC readers typically support a similar range of 3–7 cm depending on antenna size.
- IR Communication: The IR transmitter achieved a reliable range of 8–10 metres in indoor environments. Commercial IR blasters typically support ranges between 10–15 metres.
- Wi-Fi Scanning: The ESP32 detected access points within a typical indoor range of 20–30 metres, comparable to standard smartphone-based Wi-Fi scanners.

These results confirm that the proposed device performs comparably to commercial tools within its supported operational range.

© JUN 2026 | IRE Journals | Volume 9 Issue 12 | ISSN: 2456-8880
DOI: https://doi.org/10.64388/IREV9I12-1718731

TABLE IV — Bill of Materials and Component Cost (INR)

#	Component	Description	Qty	Unit (₹)	Total (₹)
1	ESP32 DevKit V1	Dual-core 240 MHz, Wi-Fi/BT, 4 MB flash	1	₹400	₹400
2	PN532 V3 NFC Module	13.56 MHz NFC/RFID, UART mode, read/write/emulate	1	₹415	₹415
3	VS1838B IR Receiver	38 kHz demodulating receiver, TTL output	1	₹20	₹20
4	IR LED 940 nm	IR transmitter, direct GPIO drive, 10 m range	1	₹17	₹17
5	2.8" SPI TFT (ILI9341)	240×320, XPT2046 touch, TFT_eSPI compatible	1	₹810	₹810
6	Micro SD Card Module	SPI interface, FAT32, 5 V supply required	1	₹50	₹50
7	Li-ion 18650 Cell	3.7 V, 2500 mAh rechargeable Li-ion cell	2	₹110	₹110
8	TP4054 Type-C Charging Module	Li-ion charge controller, Type-C input	1	₹25	₹25
9	MT3608 Boost Converter	DC-DC boost, 3.7 V to 5 V, for SD card module	1	₹60	₹60
10	Passives & Miscellaneous	10 kΩ resistors, 100 µF capacitors, jumpers, breadboard	—	—	₹33
11	SD Card	1Gb Sd card for data storage	1	₹120	₹120
12	Jumper Wire	For connecting components	—	₹100	₹100
				TOTAL	₹2,160

TABLE V — Cost Comparison: Proposed Device vs. Dedicated Tools

Function	Dedicated Tool Cost (₹)	This Device Cost (₹)	Saving
NFC Reader / Tester	₹3,500 – ₹5,000	~₹415 (PN532 V3)	~88%
Wi-Fi Analyser + Port Scanner	₹3,000 – ₹8,000	₹0 (ESP32 built-in)	100%
IR Blaster / Learner	₹1,500 – ₹3,000	~₹37 (LED + VS1838B)	~97%
TOTAL (all tools separately)	₹9,660 – ₹18,490	~₹2,160	~78–89%

© JUN 2026 | IRE Journals | Volume 9 Issue 12 | ISSN: 2456-8880
DOI: https://doi.org/10.64388/IREV9I12-1718731

TABLE VI — Feature Comparison: Proposed Device vs. Commercial Tools

Device	NFC (13.56 MHz)	Wi-Fi Scan	IR (38 kHz)	Sub-G Hz RF	BLE	Approx. Cost (₹)
Proposed Device (This Work)	Yes	Yes	Yes	No	No	~₹2,160
Flipper Zero	Yes	No	Yes	Yes	Yes	~₹18,000–22,000
ACR122U NFC Reader	Yes	No	No	No	No	~₹3,500–5,000
IR Blaster / Learner (standalone)	No	No	Yes	No	No	~₹1,500–3,000
Wi-Fi Analyzer (dedicated HW)	No	Yes	No	No	No	~₹3,000–8,000

Component prices are representative of listings on robu.in and majestronicz.com as of 2024–2025. Flipper Zero pricing reflects import/grey-market pricing in India. Dedicated tool prices reflect typical online marketplace listings.

X. PROTOTYPE IMPLEMENTATION

Fig. 4 shows the assembled prototype of the proposed multi-testing device. The current build uses a full-size breadboard to allow iterative component rearrangement during testing. The ESP32 DevKit V1 is positioned at the centre of the breadboard with SPI lines for the TFT and a separate set of SPI lines for the SD card module routed to clearly separated breadboard columns to prevent signal crosstalk. The PN532 V3 NFC module is placed at the top-right to maximise the available space for the embedded antenna. The VS1838B IR receiver is elevated on 40 mm female header pins to clear surrounding components and improve the reception angle. The IR LED is mounted on the edge of the breadboard facing outward. The 2.8-inch TFT touchscreen is positioned flat on the left side and connected via a ribbon cable to allow the display face to be held upright during use.

Fig. 4. Prototype Implementation of the Device (Breadboard Build)

The two Li-ion 18650 cells are held in a dual-cell battery holder external to the breadboard, connected via the TP4054 Type-C charging board. The MT3608 boost converter module is placed between the battery output and the SD card module's VCC pin.

A multimeter was used during initial power-up to verify that the boost converter output was trimmed to exactly 5.0 V before the SD card module was connected. The entire assembly, including battery holder, occupies a footprint of approximately 170 mm × 100 mm × 35 mm and is used as a handheld instrument by holding the breadboard in one hand and operating the touchscreen with the other.

A compact 3D-printed enclosure is planned for the next revision. All three primary functions — NFC card reading, Wi-Fi network scanning, and IR signal capture and replay — were validated on this

breadboard prototype under real operating conditions in an office building environment before the test data reported in Section IX were collected.

XI. ADVANTAGES

The most significant advantage of the proposed device is its cost. At approximately ₹2,160, it consolidates three testing functions that would cost ₹9,660–₹18,490 if purchased as separate dedicated instruments — a saving of 78–89%. For a student laboratory, a startup, or an individual developer, this difference is practically meaningful.

A unified local data store is a second substantive advantage. NFC event logs, Wi-Fi scan CSVs, and the IR command library reside on a single SD card in a consistent directory structure. Cross-referencing test data from different modules — for example, correlating Wi-Fi scan timestamps with NFC events during an access-control audit — requires no file transfer between separate tools.

The fully standalone touchscreen interface eliminates the need for a laptop, serial terminal, or companion application in the field. The entire testing workflow — from module selection to data capture to data review — is completed on the device itself. Because all data are stored locally, the device functions correctly in air-gapped, network-denied, or signal-restricted environments without any degradation.

From a firmware architecture standpoint, the modular file organisation (separate .cpp files per module, one state machine in main.cpp) means that adding a new testing function — a BLE scanner, a temperature sensor module, or a frequency counter — involves writing a new module file and adding one tile to the home screen without modifying any existing code.

XII. REAL-WORLD APPLICATIONS

In access-control validation, the NFC module enables engineers to verify reader behaviour against multiple credential types without carrying a physical card collection. Emulation mode allows a complete set of UIDs to be loaded from the SD card and presented to a reader in sequence, automating credential compatibility testing. The timestamped NFC log on the SD card provides an audit trail for security reviews. This workflow is applicable to office entry systems, university library gates, and smart locker installations.

In home and building automation projects, the IR capture-and-replay function is particularly valuable when a device manufacturer does not publish IR command codes. During validation testing, an engineer walked through a multi-room installation, captured every remote-control command for five appliances in a single session lasting approximately 20 minutes, and produced a complete IR library on the SD card. The device then functioned as a universal transmitter for all five appliances during subsequent integration testing.

For network infrastructure assessment, the Wi-Fi scanner provides a dedicated, fixed-point hardware logger that can be left unattended to capture periodic scan snapshots to the SD card — a capability not easily replicated by a smartphone application without keeping the screen active. The integrated port scanner adds a lightweight reconnaissance layer for verifying firewall policies on local network segments, useful during both installation and routine maintenance audits.

In educational contexts, the device provides a single platform covering NFC, Wi-Fi, and IR technologies — three of the most commonly taught wireless protocols in embedded systems and IoT curricula. The build cost is lower than a typical laboratory component kit, and the modular firmware architecture makes it straightforward for students to extend the device with new features as coursework assignments.

XIII. CONCLUSION

This paper presented the complete design, implementation, and validation of a cost-effective IoT-based multi-testing device that consolidates NFC card operations, Wi-Fi environment analysis, and IR signal capture and replay into a single ESP32-based unit at a component cost of approximately ₹2,160. Functional testing demonstrated 100% NFC authentication accuracy at 78 ms average latency, Wi-Fi RSSI measurements within 2 dBm of a

reference instrument, 100% IR command replay success across 47 commands and four appliance brands at up to 10 metres, and reliable SD card data logging across 15 power cycles without corruption.

Several practical integration lessons emerged during development that are documented in this paper to benefit practitioners: the PN532 V1 module is unreliable in SPI mode at 3.3 V and should be replaced by the V3 variant in UART mode; the TFT display and SD card module must be assigned

XIV. FUTURE SCOPE

Several enhancements are planned for future revisions. On the hardware side, a Sub-GHz RF module (CC1101) will be added to support 315 MHz, 433 MHz, and 868 MHz testing — the most significant capability gap relative to the Flipper Zero identified in this work. A BME280 environmental sensor will be integrated for ambient condition logging alongside wireless test data. A 3D-printed enclosure with dedicated apertures for the PN532 antenna, IR receiver, and IR LED will replace the current breadboard assembly, improving portability and mechanical robustness.

On the firmware side, a BLE scanning module will be added to the Wi-Fi mode, extending wireless environment coverage to Bluetooth Low Energy devices. On the IR side, TensorFlow Lite Micro will be evaluated for on-device protocol classification to handle unknown or proprietary IR schemes without manual raw-capture fallback.

A software-controlled backlight dimming circuit — requiring either a dedicated TFT module with a reliable PWM-capable backlight driver or an external MOSFET circuit — will be designed to extend battery life meaningfully beyond the current 3–4-hour operating period. MIFARE DESFire EV1 emulation support is also targeted for the NFC module to extend access-control testing coverage to modern high-security credential systems.

ACKNOWLEDGMENT

The authors thank the Department of Computer Applications, Sri Chandrasekharendra Saraswathi Viswa Mahavidyalaya (SCSVMV), Enathur, Kanchipuram, for providing laboratory access, hardware components, and guidance throughout this project.

REFERENCES

[1] M. Babiuch, P. Foltynek, and P. Smutny, "Using the ESP32 Microcontroller for Data Processing," in Proc. 20th Int. Carpathian Control Conf. (ICCC'2019), Krakow-Wieliczka, Poland, May 2019, pp. 88-93, doi: 10.1109/CarpathianCC.2019.8765944.

[2] S. Patil, D. Shinde, and E. Devmore, "ESP32 Temperature Control Project with an Emphasis on Energy Efficiency," Int. J. All Subject Res. (IJASR), vol. 3, no. 3, pp. 1-9, 2024.

[3] D. Hercog, T. Lerher, M. Truntic, and O. Tezak, "Design and Implementation of ESP32-Based IoT Devices," Sensors, vol. 23, no. 15, p. 6739, Jul. 2023, doi: 10.3390/s23156739.

[4] A. Maier, A. Sharp, and Y. Vagapov, "Comparative Analysis and Practical Implementation of the ESP32 Microcontroller Module for the Internet of Things," in Proc. 7th Int. Conf. Internet Technologies and Applications (ITA 2017), pp. 143-148, Nov. 2017.

[5] I. Allafi and T. Iqbal, "Design and Implementation of a Low-Cost Web Server Using ESP32 for Real-Time Photovoltaic System Monitoring," in Proc. IEEE EPEC 2017, pp. 1-5, Feb. 2018.

[6] D. Ghosh, A. Agrawal, N. Prakash, and P. Goyal, "Smart Saline Level Monitoring System Using ESP32 and MQTT-S," in Proc. 20th Int. Conf. e-Health Networking, Applications and Services (Healthcom 2018).

[7] A. Iqbal and T. Iqbal, "Low-Cost and Secure Communication System for Remote Micro-grids using AES Cryptography on ESP32 with LoRa Module," in Proc. IEEE EPEC 2018, Oct. 2018.

[8] M. Kassab, J. DeFranco, and P. Laplante, "A Systematic Literature Review on Internet of Things in Education: Benefits and Challenges," J. Comput. Assist. Learn., vol. 36, pp. 115-127, 2020.

[9] B. S. Sarjerao and A. Prakasarao, "A Low-Cost Smart Pollution Measurement System Using REST API and ESP32," in Proc. 3rd Int. Conf. Convergence in Technology (I2CT 2018).

[10] L. O. Aghenta and M. T. Iqbal, "Low-Cost, Open Source IoT-Based SCADA System Design Using Thinger.IO and ESP32 Thing," Electronics, vol. 8, no. 8, p. 822, 2019.