12th INTERNATIONAL SCIENTIFIC CONFERENCE ON DEFENSIVE TECHNOLOGIES
OTEH 2026
Serbia, Tara, 21 - 24 October 2026

IMPLEMENTATION OF A LOW-COST HP-IB (GPIB, IEEE 488) INTERFACE AND MATLAB GUI APPLICATION FOR HP 3582A SPECTRUM ANALYZER CONTROL

PETAR MIŠLJEN
Military Technical Institute, Belgrade, petarmisljen@gmail.com

MILOŠ PAVIĆ
Military Technical Institute, Belgrade, meelos.pavic@gmail.com

SAŠA GUNDELJ
Military Technical Institute, Belgrade, gundelj@yahoo.com

IGOR GORŠIĆ
Military Technical Institute, Belgrade, gorsic@gmail.com

Abstract: *This paper presents a low-cost implementation of an HP-IB (GPIB, IEEE 488) interface based on the open-source AR488 firmware running on an Arduino Nano microcontroller. HP-IB is a short-range, 8-bit parallel, multi-master digital interface bus used for remote control and data acquisition between laboratory instruments and controllers. In this work, the HP 3582A spectrum analyzer serves as a case study.*

The developed system converts data from the GPIB bus to USB serial communication. A graphical user interface (GUI) application was designed in the MATLAB App Designer environment. The application utilizes HP-IB commands to enable instrument connection, parameter configuration, data acquisition, graphic spectrum display, and export of the measured data in ASCII format.

The proposed solution demonstrates how the operational lifespan of legacy laboratory instruments can be successfully extended while allowing their seamless integration and automation in modern laboratory environments. The obtained results confirm reliable communication, accurate data transmission, and a substantial expansion of the HP 3582A's functionality under contemporary laboratory conditions.

Keywords: *HP-IB, GPIB, IEEE-488, AR488, Arduino Nano, HP 3582A, MATLAB App Designer, data acquisition, measurement automation.*

1. INTRODUCTION

The rapid advancement of modern measurement instrumentation has rendered many legacy laboratory devices obsolete, despite their continued technical reliability and high performance. The HP 3582A, a dual-channel dynamic signal analyzer from the late 1970s, exemplifies such equipment [1]. Based on the Fast Fourier Transform (FFT) algorithm, it offers excellent low-frequency spectral analysis capabilities and remains valuable for applications in audio engineering, vibration analysis, and acoustics [2]. However, the lack of native integration with contemporary computers and automated test systems limits its usability in modern laboratory environments. The primary barrier is the obsolescence of the Hewlett-Packard Interface Bus (HP-IB), later standardized as the General Purpose Interface Bus (GPIB) interface under IEEE 488 standard, which, although robust and widely adopted in the past, is no longer natively supported by current personal computers.

This paper presents a low-cost implementation of HP-IB interface based on the open-source AR488 [3] firmware running on an Arduino Nano microcontroller. The developed system enables bidirectional communication between the HP 3582A spectrum analyzer and a personal computer via USB, converting HP-IB commands and data into serial communication. A user-friendly graphical user interface (GUI) application was developed in MATLAB App Designer to facilitate instrument control, parameter configuration, spectrum visualization, and data export in ASCII format. By demonstrating reliable communication and extended functionality of legacy instrumentation, this work contributes to the preservation and modernization of valuable laboratory equipment at minimal cost, offering a practical solution for educational institutions and research laboratories.

2. GPIB (IEEE 488)

One of the longest-standing standards for instrument

Corresponding author: S. Gundelj;
E-mail: gundelj@yahoo.com
DOI: 10.5937/OTEH26149M

interconnection is the GPIB (IEEE 488) [4], [5]. Originally developed by Hewlett-Packard in the late 1960s under the name HP-IB. The GPIB standard was formalized in 1975 and subsequently updated as IEEE 488.1 (electrical and mechanical specifications) and IEEE 488.2 (command and data formats). GPIB (IEEE 488) is an 8-bit parallel bus with a maximum theoretical data rate of 1 MB/s, supporting up to 15 devices over short distances (typically up to 20 m with proper cabling). Its robust handshake mechanism and multi-master capability made it the de facto standard in automated test and measurement systems for decades [3]. Figure 1 illustrates the standard GPIB (IEEE 488) connector pinout.

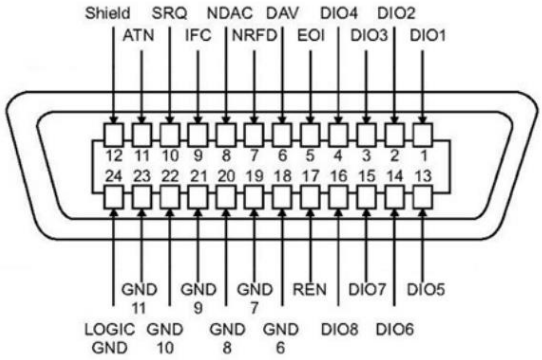


Figure 1. GPIB (IEEE 488) Fimale Connector Pinout [3]

Bus operation is based on dedicated control and handshake lines. The Attention (ATN) line differentiates between command and data modes, while the End Or Identify (EOI) line signals the end of data transfer or enables parallel polling. System control is maintained through Interface Clear (IFC) and Remote Enable (REN) lines, managed exclusively by the system controller. Asynchronous service requests are handled via the Service Request (SRQ) line. Data transfer is synchronized using the three-wire handshake protocol consisting of Data Valid (DAV), Not Ready for Data (NRFD), and Not Data Accepted (NDAC) lines, providing reliable communication even with devices of varying speeds [4], [5]. Figure 2 illustrates the functional grouping of the GPIB (IEEE 488) lines.

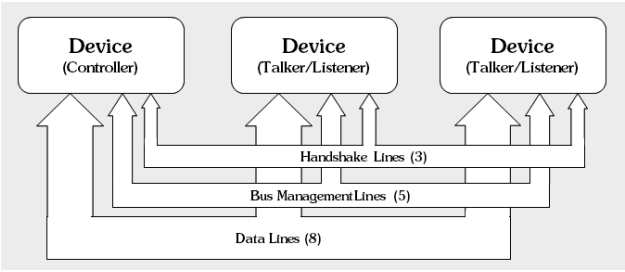


Figure 2. GPIB (IEEE 488) Functional Line Groups [6]

Devices on the bus assume one of three primary roles: Controller, Talker, or Listener. The Controller (usually a computer) manages bus arbitration, addressing, and overall system operation. Multiple controllers are supported, with clear distinction between the System Controller and the active Controller-in-Charge. Addressing is performed using primary addresses from 0 to 30, with Talk and Listen commands derived by adding fixed offsets to the device address. This flexible architecture, supporting configurations from simple point-to-point transfers to complex multi-controller systems, ensures deterministic behavior and remains highly relevant for the modernization of legacy laboratory equipment, as demonstrated in this work.

3. HP 3582A SPECTRUM ANALYZER

The HP 3582A is a classic dual-channel dynamic signal analyzer developed by Hewlett-Packard in the late 1970s. It is based on the Fast Fourier Transform (FFT) algorithm and provides high-resolution spectral analysis in the low-frequency range from 0.02 Hz to 25.5 kHz. Thanks to its advanced digital signal processing capabilities at the time, the instrument achieves a frequency resolution of up to 0.02 Hz with real-time spectrum display, making it particularly suitable for applications in audio engineering, vibration analysis, acoustics, and sub-audio measurements [1], [2]. Figure 3 shows the external appearance of the instrument.



Figure 3. HP 3582A Spectrum Analyzer

Within the HP-IB bus, the HP 3582A offers extensive remote control capabilities. Virtually all front-panel functions are programmable, and the instrument supports full read/write access to its internal memory, including time records, amplitude and phase spectra, transfer functions, coherence functions, and alphanumeric CRT display data [1]. Figure 4 presents a typical display of the analyzer during operation. These features enable efficient integration into automated measurement systems, where data can be systematically acquired, stored, post-processed, and compared with reference values.

From the signal processing perspective, the HP 3582A implements digital heterodyne processing followed by the FFT algorithm, allowing band-selectable analysis with exceptionally high frequency resolution. The instrument supports multiple window functions (rectangular, Hanning, and flat-top) to optimize the trade-off between spectral leakage and amplitude accuracy. Additionally, the built-in sine wave and pseudo-random noise generator (periodic and random) enables standalone testing and calibration of the measurement chain. The dual-channel configuration further supports advanced analyses such as transfer function and coherence measurements — key parameters for system identification and linearity assessment [2]. Due to these characteristics, the HP 3582A serves not only as a measurement instrument but

also as an excellent platform for studying dynamic systems in the frequency domain.

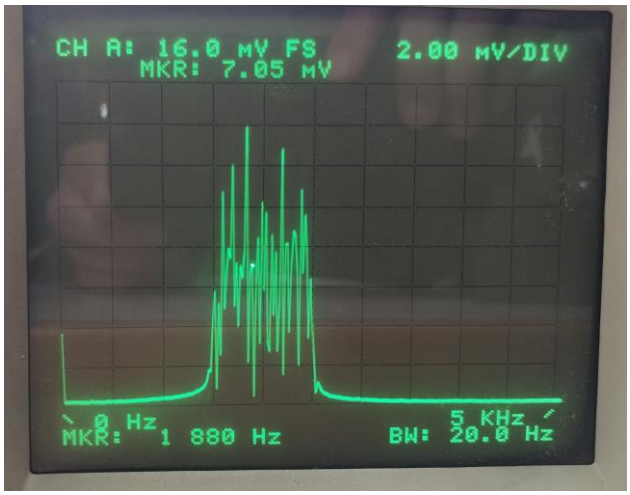


Figure 4. HP 3582A Display Example

4. AR488

The AR488 is an open-source Arduino-based GPIB (IEEE 488) controller developed by Twilight Logic [3], which provides a low-cost alternative to commercial GPIB (IEEE 488) interfaces. The firmware has been developed and tested on various Arduino boards, including the Uno, Nano, Mega 2560, Micro, and Leonardo, with additional support for certain ATmega and ESP32 variants [3]. It emulates Prologix command syntax (with minor exceptions) and enables seamless USB-to-GPIB communication.

The AR488 firmware supports SN75160 and SN75161 GPIB (IEEE 488) transceiver integrated circuits. These ICs provide a buffer between the Arduino and the GPIB (IEEE 488) bus and allow the full 48 mA drive current for a GPIB (IEEE 488) device. In addition, when powered down, these devices present a high impedance to the GPIB (IEEE 488) bus so that the connected device does not interfere with the operation of the bus. [3] In this proposed solution, the SN75160 and SN75161 integrated circuits were not used, as the HP 3582A demonstrated reliable operation in a configuration with one computer. However, for devices requiring higher current drive or full IEEE 488.1 compliance, the use of these IC's is recommended.

Configuration is performed by editing the AR488_Config.h header file, while firmware updates are done through the Arduino IDE. The interface was assembled by soldering short wires from an GPIB (IEEE 488) male connector directly to the Arduino Nano pins, as illustrated in Figure 5. The complete assembly was then placed in a custom 3D-printed enclosure (Figure 6) for mechanical protection and practical use.

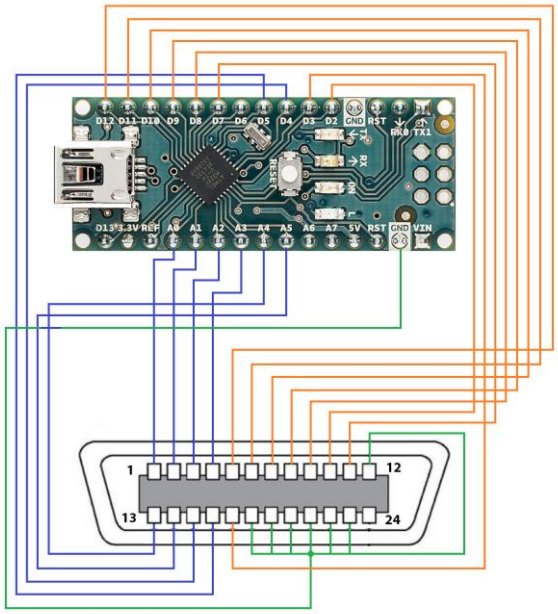


Figure 5. Wiring diagram (male connector)

Testing of the interface was performed using the CoolTerm terminal emulator. The controller version was first verified by issuing the command ++ver, which elicited the response AR488 GPIB controller, ver. 0.53.33, 12/12/2025. Verbose mode was subsequently enabled with the command ++verbose, resulting in the confirmation Verbose: ON.

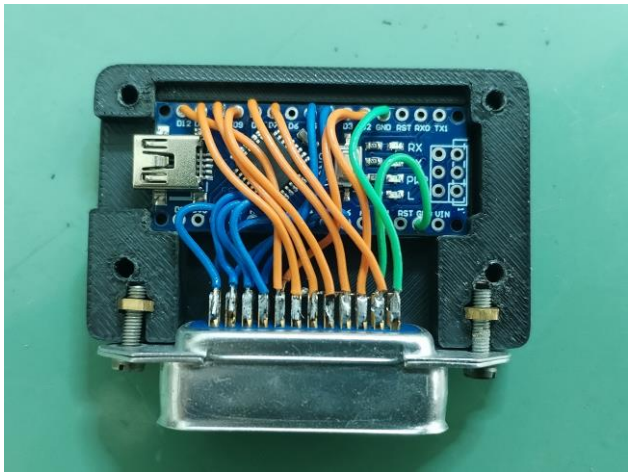


Figure 6. The complete assembly (male connector)

The minimum set of commands required to configure the controller and enable communication with the HP 3582A spectrum analyzer was then applied. The command ++mode 1 configured the AR488 to operate as Controller-in-Charge (CIC) on the HP-IB bus. This was followed by ++auto 1, enabling automatic reading of responses from the instrument, and ++addr 11, which set the HP-IB address of the HP 3582A to its factory default value of 11 (octal). Measured data were subsequently acquired from the spectrum analyzer by issuing the command LDS (Low Data Strobe), which returned a sequence of 256 values. The complete testing procedure is presented in Figure 7.

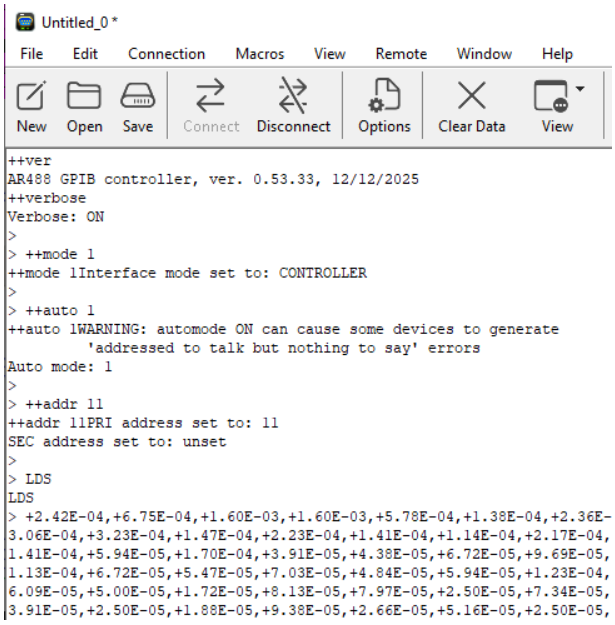


Figure 7. CoolTerm view

This low-cost and compact solution demonstrates the practicality of open-source hardware for extending the usability of legacy GPIB (IEEE 488) instruments.

5. MATLAB GRAPHICAL USER INTERFACE

Prior to developing the full graphical user interface, reliable communication between the AR488 GPIB adapter and the HP 3582A spectrum analyzer was established and tested using MATLAB's serialport interface. The AR488 serves as a USB-to-GPIB bridge, allowing transmission of HP-IB commands over a virtual serial port.

Figure 8 illustrates the initial MATLAB command window test sequence used to configure the adapter and retrieve measurement data.

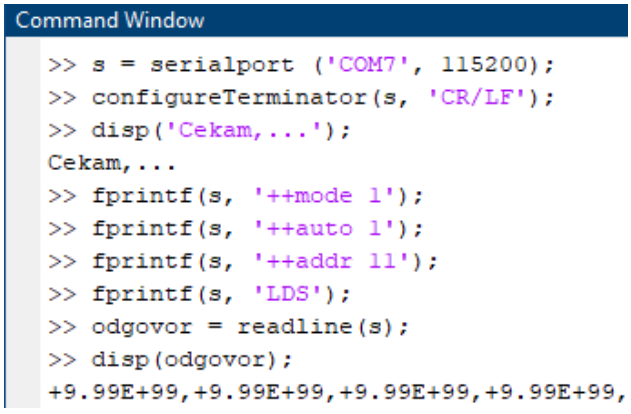


Figure 8. Matlab Command Window

After successful verification of low-level communication, the graphical user interface was designed in MATLAB R2023b App Designer using a modular layout. The main window is divided into three primary sections: (1) a connection and control panel at the top, (2) a central visualization area with axes for spectrum plotting, and (3) a status and data management panel at the bottom. This structure provides an intuitive workflow. Standard App Designer components (drop-down menus, buttons, labels, and axes) were used, with all functionality implemented through callback functions. Figure 9 presents the main window of the developed HP 3582A Control Application (version 1.2).

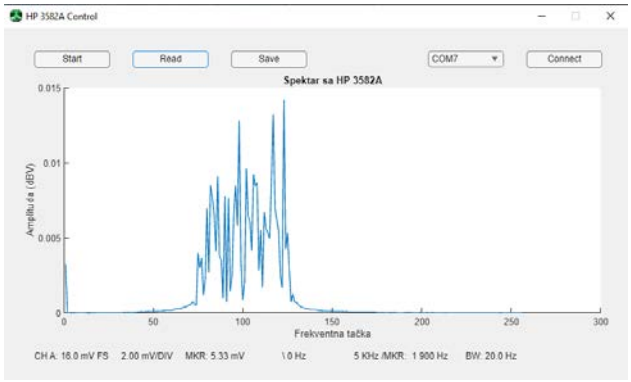


Figure 9. HP 3582A Application V1.2

The core operations are realized as follows. Instrument connection is handled by the ConnectButtonPushed callback function, as illustrated in Figure 10.

```
% Button pushed function: ConnectButton
function ConnectButtonPushed(app, event)
    try
        port = app.COMDropDown.Value;
        app.SerialPort = serialport(port, 115200);  % Baud iz AR488 default
        configureTerminator(app.SerialPort, 'CR/LF');  % ++eos 0 = CR+LF
        app.StatusLabel.Text = 'Čekam inicijalizaciju AR488 (3 sekunde)';
        pause(3.0);
        fprintf(app.SerialPort, '++ver');
        pause(1.0);
        app.StatusLabel.Text = readline(app.SerialPort);
        pause(3.0);
        app.StatusLabel.Text = ['Povezano na ' port '- AR488 inicijalizovan.'];
    catch e
        app.StatusLabel.Text = ['Greška pri povezivanju: ' e.message];
    end
end
```

Figure 10. 'ConnectButtonPushed' function

Upon pressing the Connect button, the application opens the selected COM port at 115200 baud, sets the appropriate terminator, and initializes the AR488 adapter. Comprehensive error handling using try-catch blocks ensures robust operation.

Configuration of the HP 3582A is performed in the StartButtonPushed function, which sequentially sends the necessary commands (++mode 1, ++auto 1, ++addr 11) with appropriate delays (Figure 11).

```
% Button pushed function: StartButton
function StartButtonPushed(app, event)
    try
        app.StatusLabel.Text = 'Setujem HP 3582A (9 sekundi).';
        fprintf(app.SerialPort, '++mode 1');
        app.StatusLabel.Text = 'Setujem Controller mode';
        pause(3.0);
        fprintf(app.SerialPort, '++auto 1');
        app.StatusLabel.Text = 'Setujem auto mode';
        pause(3.0);
        fprintf(app.SerialPort, '++addr 11');
        app.StatusLabel.Text = 'Setujem adresu 11';
        pause(3.0);
        app.StatusLabel.Text = 'Setovanje završeno.';
    catch
        app.StatusLabel.Text = 'Greška pri pokretanju setovanja.';
    end
end
```

Figure 11. 'StartButtonPushed' function

Measurement acquisition and visualization are implemented in the ReadButtonPushed function. When the Read button is activated, the application sends the LDS command, reads the returned data string, converts it into a numeric array using str2double and strsplit, and stores it in the application property SpectrumData. The spectrum is immediately plotted on the designated axes with appropriate labels and title. Additionally, the LAN command is sent afterward to return the instrument to local mode.

Finally, the SaveButtonPushed function enables data export. If spectral data is available, the user is prompted via uiputfile to select a save location. The data is saved as a .txt file in ASCII format with a descriptive header containing timestamp and metadata, ensuring traceability and ease of further processing.

The integration of these callback functions with the graphical components resulted in a robust, user-friendly application that significantly simplifies control of the HP 3582A spectrum analyzer while maintaining full transparency and reproducibility of the measurement process.

During repeated testing sessions, the system exhibited stable operation with consistent spectral data acquisition and no observed transmission errors. The MATLAB GUI provided intuitive and responsive control, noticeably simplifying the measurement workflow compared to manual operation of the HP 3582A.

6. CONCLUSION

This paper successfully demonstrated a low-cost HP-IB (GPIB, IEEE 488) interface based on the open-source AR488 firmware and Arduino Nano microcontroller, coupled with a custom MATLAB GUI application for controlling the HP 3582A spectrum analyzer. The developed system enables reliable bidirectional communication, remote parameter configuration, spectrum acquisition and visualization, as well as data export in ASCII format. By utilizing inexpensive and widely available components, the proposed solution significantly extends the operational life of legacy laboratory instrumentation while seamlessly integrating it into modern computer-controlled measurement environments. The obtained results confirm stable operation and accurate data transmission, validating the practical value of the approach for educational and research purposes.

The presented work opens several avenues for further development. The MATLAB application can be expanded to support virtually all HP-IB commands available on the HP 3582A, enabling more comprehensive automation. Additionally, integration with MATLAB and Simulink environments [7] would allow advanced post-processing of acquired data, such as enhanced filtering, statistical analysis, or system identification techniques not natively supported by the instrument itself. Future research may also explore alternative microcontrollers (e.g., ESP32) to overcome the speed limitations of the Arduino Nano, although the current platform proved fully sufficient for the intended low-frequency applications. Overall, this low-cost modernization strategy contributes to more sustainable and accessible laboratory infrastructure.

References

[1] HEWLETT-PACKARD: *Model 3582A Spectrum Analyzer Operating Manual. Part No. 03582-90002,* Loveland, Colorado, USA, 1981.

[2] HEWLETT-PACKARD: *A High-Resolution, Low-Frequency Spectrum Analyzer.* HP Journal. Palo Alto, California, USA, 1987.

[3] TWILIGHT LOGIC: *AR488 GPIB Controller User Manual*, Twilight Logic, 2025.

[4] THE INSTITUTE OF ELECTRIC AND ELECTRONICS ENGINEERS, INC: *488.1-1987 - IEEE Standard Digital Interface for Programmable Instrumentation*, IEEE, New York, 1988.

[5] THE INSTITUTE OF ELECTRIC AND ELECTRONICS ENGINEERS, INC: *488.2-1987 - IEEE Standard Codes, Formats, Protocols, and Common Commands*, IEEE, New York, 1988.

[6] THE HP 9845 PROJECT, www.hp9845.net, 2026.

[7] MATLAB OFFICIAL WEB PAGE, www.mathworks.com, 2026.

[8] HEWLETT-PACKARD: *HP-IB Command Library, Hewlett-Packard*, Corvallis, Oregon, USA, 1986.

[9] NATIONAL INSTRUMENTS: *GPIB – PC IEEE-488 Instrumentation interface,* National Instruments Corporation, Texas, 1984.

[10] S. AJKALO, O. BJELICA: *One Implementation of IEEE 488 USB Controller,* Elektrotehnički fakultet Istočno Sarajevo, Istočno Sarajevo, 2010.

[11] S. AJKALO, M. GAZIVODA, S. LUBURA, M. ŠOJA: *GPIB and Measurement Instrumentation – Control of Instruments, Acquisition and Presentation of Measurement Data*, Infoteh – Jahorina Vol 6, Istočno Sarajevo, 2007.