# Automated Column Stand: Standard Operating Procedure

Ryan Hogan, Engineering Intern, Summer 2025

## Overview

This documentation was made to help operators using the automated test stand. The dashboard and control panel should be accessible on the iPad connected to the column stand. Use the following documentation when being trained on the new system and to help troubleshoot when problems may occur.

## Major Parts List

|  |  |
| --- | --- |
| **PART** | **LINK** |
| ESP32-DEVKIT-C | [ESP32-C6-DEVKITC-1-N8](https://www.digikey.com/en/products/detail/espressif-systems/ESP32-C6-DEVKITC-1-N8/17728861) |
| Flow Meter | [DMA20-AAAAA1 Endress+Hauser | Industrial Automation and Controls | DigiKey](https://www.digikey.com/en/products/detail/endress-hauser/DMA20-AAAAA1/17140769?gclsrc=aw.ds&gad_source=1&gad_campaignid=20243136172&gclid=CjwKCAjw4K3DBhBqEiwAYtG_9Nic6GV5G9iVCKkkcvaqPpTFLpSUF5Qw7qQC18XumS8sreTEZRJqrxoCE4MQAvD_BwE) |
| Pressure Gauges | [Cole-Parmer High-Accuracy Digital Gauge, 0 to 100 psig, 4-Digit LCD from Cole-Parmer](https://www.coleparmer.com/i/cole-parmer-high-accuracy-digital-gauge-0-to-100-psig-4-digit-lcd/6834906) |

## Features

### Calibration

Calibrating the sensors is done by setting the values for the high/low limits on each sensor. These values are accessible for the flow meter via the SmartBlue app on the iPad. For the pressure gauge, you can set/access the limits by following the instructions in Document A in the Appendix.

### Data Export

The data export feature allows operators to download data via .csv file or by copying and pasting a data table into an excel file.

### System Configuration

The system configuration window allows operators to set update intervals, the pressure threshold and set up the ESP32 with an IP address. The pressure threshold will determine when the test stops, and it is recommended to stay at 20 psi.

### Pause/Start/Reset Test

The pause/start/reset buttons allow operators to control the status of the current test. When pausing the test, the valve will close, and data collection will pause. The collected data will not clear until the reset button is clicked. If paused, the test will have the ability to be resumed with no clearing of data.

## How to Operate: Step by Step

### Connecting with ESP32

Connect with the ESP32 by typing in the IP address and clicking connect. Be sure that the computer/iPad connecting is also on the PRENGIOT Wi-Fi or else connection will not work. The IP address for the current ESP32 is 10.107.6.137.

### Pre-Test Setup (10 minutes)

1. System Check
   * Verify power connections
   * Check sensor connections
   * Confirm valve connection
   * Power on ESP32
2. Web Interface
   * Navigate to ESP32 IP in browser
   * Verify "Connected" status
   * Set pressure threshold (5-100 PSI)
   * Configure update interval (0.5-120 seconds)

### Test Setup (15 minutes)

1. Sample Preparation
   * Install test filter in flow path
   * Connect inlet/outlet pressure sensors
   * Position flow sensor
   * Verify leak-free connections
2. Calibration (Optional)
   * Use calibration menu for sensors
   * Apply known values at min/max points
   * Verify accuracy within ±2% (flow), ±1% (pressure)
   * Save calibration to EEPROM

### Test Execution

1. Start Test
   * Click "Start Test" in web interface
   * Verify valve opens and flow begins
   * Monitor initial readings for stability
   * Confirm data logging is active
2. Monitor Test (Up to 72 hours)
   * Check system every 4-8 hours
   * Verify stable flow and pressure readings
   * Monitor real-time charts and data
   * Watch for threshold warnings

### Test Completion (10 minutes)

1. Stop and Export
   * Click "Stop Test" when complete
   * Verify valve closes automatically
   * Export data via CSV download or clipboard copy
   * Click "Reset" to clear system
2. Shutdown
   * Disconnect test sample
   * Clean flow path components
   * Power down system
   * Archive test data and documentation

### Emergency Procedures

If pressure threshold exceeded:

* System automatically closes valve
* Investigate cause before restart

If connection lost:

* Check power and network
* Data retained locally on ESP32

If sensor errors:

* Verify connections and recalibrate

### Quality Checkpoints

#### Before Test:

* All sensors online
* Valve operates properly
* WiFi connected
* Configuration saved

#### During Test:

* Stable flow rate
* Data logging continuous
* No error messages

#### After Test:

* Data exported successfully
* Equipment cleaned
* Test documented

## Maintenance

### Sensor Calibration

Contact other engineers for help with flow meter and pressure gauge calibration. This should be done every \_\_\_\_\_\_ months and marked on the sensor when completed.

## FAQs/Troubleshooting

### How do I find the ESP32 IP Address?

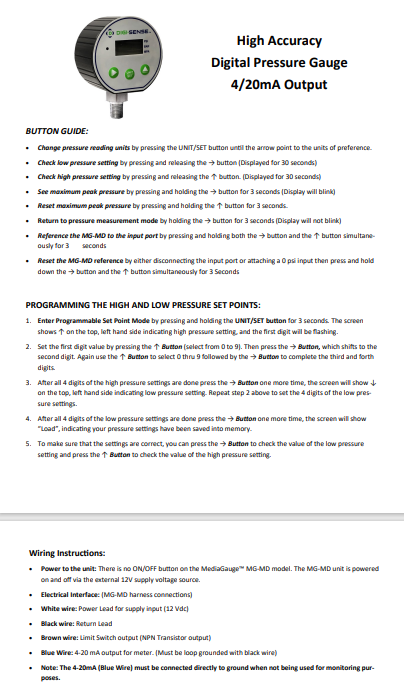
The IP address should be \_\_\_\_\_\_\_\_\_\_. If this is not working, you will need to download the backend software and run this on Arduino IDE (<https://www.arduino.cc/en/software/>) while connected to the ESP32 via USB-C cable. When you compile and run the code, the IP address will be printed on the serial monitor in the Arduino IDE.

### Why can’t I connect to the ESP32?

You need to be on the PNRENGIOT Wi-Fi to connect to the ESP32. If you do not have the password for this Wi-Fi, please contact IT (Greg Schraufnagel) for password.

## Appendix

### Document A: Pressure Gauge Documentation



### Document B: Flow Meter Documentation (<https://tinyurl.com/PicoMagIODoc>)

Diagram of a device plug

AI-generated content may be incorrect.

A close-up of a box

AI-generated content may be incorrect.

A white box with black text

AI-generated content may be incorrect.

A diagram of a circuit

AI-generated content may be incorrect.

A screenshot of a computer

AI-generated content may be incorrect.

A graph of a function

AI-generated content may be incorrect.

### Document C: How Does the System Work?

The communication flow begins when the ESP32 establishes a WiFi connection to the local network and starts an HTTP web server listening on port 80. Browser clients discover and connect to the ESP32 using its assigned IP address, initiating a client-server relationship where the ESP32 serves both static web content and dynamic API endpoints.

Real-time data acquisition occurs through continuous analog-to-digital conversion on three dedicated GPIO pins, where the ESP32 samples pressure and flow sensors at 12-bit resolution with 10-sample averaging for noise reduction. The raw voltage readings undergo scaling transformations using configurable minimum and maximum ranges to convert sensor outputs into engineering units.

During active test operations, the browser establishes a polling loop that sends GET requests to the ESP32's data endpoint at user-configurable intervals ranging from 500 milliseconds to 10 seconds. Each request triggers the ESP32 to perform immediate sensor readings, calculate derived values like pressure differential and cumulative volume, and package the results into structured JSON responses.

The ESP32 maintains internal test state including elapsed time, total volume calculations, and safety threshold monitoring. When pressure limits are exceeded, the system automatically triggers safety responses by closing the solenoid valve and notifying connected clients through status updates in subsequent polling responses.

Configuration changes flow bidirectionally, where browser clients send POST requests containing updated parameters such as pressure thresholds, sensor scaling factors, and operational settings. The ESP32 processes these requests, updates internal variables, and optionally persists critical settings to EEPROM for retention across power cycles.

The system implements graceful degradation where connection losses are detected through failed HTTP requests, triggering automatic reconnection attempts on the client side and fail-safe valve closure on the ESP32 side. Multiple clients can connect simultaneously to monitor the same test session, with each receiving independent data streams through their individual polling cycles.

Emergency stop functionality operates through immediate HTTP POST requests that bypass normal polling intervals, ensuring rapid response times under 100 milliseconds for critical safety operations. The ESP32 processes these commands with highest priority, immediately updating hardware outputs and internal state before responding to the client.

Data export capabilities utilize browser-initiated requests that trigger the ESP32 to format historical data into CSV structures, which are then transmitted as downloadable content through standard HTTP file transfer mechanisms, allowing seamless integration with external analysis tools.