Simulation-Driven Design of a Microcontroller-Based Programmable Voltage Stabilizer with Relay Control

Md Mubdiul Hasan*, *Electronics Design Engineer*, Chandan Chaudhary †, *Student Member, IEEE*
* Flextronics International AB, Kalmar, Sweden
†Electrical and Computer Engineering, Michigan State University, East Lansing, MI, USA 48824
Emails:mhasannoor@gmail.com, chaudharychandan98096@gmail.com

***Abstract*—Power quality issues and voltage fluctuations continue to pose significant challenges to the reliable operation of electrical equipment. This paper develops a microcontroller-based programmable voltage monitoring and protection system using a Proteus virtual environment. The design incorporates a virtual Arduino controller, an AC source with adjustable load conditions, and an integrated signal-conditioning module to feature display and relay-based control components. A real-time graphical interface and an LCD module are used to continuously display input and regulated output voltages during operation. The system intelligently detects deviations beyond an acceptable voltage range and initiates protective shutdown to safeguard connected loads. The complete functionality is validated through comprehensive virtual prototyping, to demonstrate a practical and cost-efficient approach for pre-hardware evaluation of programmable reference systems in voltage regulation and protection applications.**

***Index Terms*—Arduino, voltage stabilizer, Proteus, simulation,**

I. Introduction

A stable alternating-current (AC) voltage supply is a fundamental requirement for the reliable and efficient operation of modern electrical and electronic systems. Even moderate deviations from nominal voltage may trigger malfunction, accelerate equipment aging, or lead to permanent damage in voltage-sensitive devices [1]. As power-electronic converters and digitally controlled loads become more common, their tolerance to voltage disturbances continues to shrink, which makes power-quality preservation a critical concern in residential, commercial, and industrial infrastructures [2]. Supply networks frequently experience voltage fluctuations caused by grid instability, sudden load variation, and distribution feeder losses. These scenarios result in sags, swells, transient distortions, or sustained undervoltage or overvoltage faults, each capable of degrading performance, shortening equipment lifetime, and compromising operational safety [3]. To address these challenges, voltage stabilizers, more broadly automatic voltage regulators (AVRs), maintain supply standards by continuously supervising input voltage and enabling a regulated level compatible with the safe operation of connected loads.

Historically, conventional stabilizers have employed tap-changing transformers and autotransformers to correct input deviations [4]–[6]. These devices alter the effective transformer turns ratio through mechanical switching of taps under varying supply conditions. Although widely used, their operational characteristics impose restrictions. Mechanical switching delays reduce the ability to address rapid voltage changes. Contact wear increases maintenance overhead. Discrete tap steps limit regulation resolution. In addition, the absence of real-time diagnostic and protective logic limits their suitability for modern environments where sensitive electronics dominate the load mix [3], [7]. Industrial installations operating continuously are especially vulnerable, as a single poorly corrected disturbance can result in irreversible equipment failure or costly downtime.

Advances in microcontrollers and power-electronics control architecture have led to stabilizers that respond faster and accommodate programmable voltage thresholds. Microcontroller-based solutions perform measurement, decision-making, user feedback, and fault isolation through compact digital circuitry. A PIC-based stabilizer documented in [8] adjusts a multi-tap transformer when the supply crosses safe limits and disconnects downstream equipment when voltage exceeds allowable tolerances. An Arduino-driven implementation investigated in [6] demonstrated that relay-operated tap adjustment achieves improved voltage consistency across multiple regulation steps even with varying supply conditions. Additional work in [9] validates that microcontroller supervision prevents overload conditions by isolating the load until system voltage returns to operational thresholds. These examples confirm the practical value of embedded controllers in stabilizer performance enhancement and protection functionality.

Digital supervision further enables fault detection, real-time display capabilities, and user-adjustable operation. However, control algorithms and implementation architectures vary greatly. Some designs employ comparator techniques to activate relays when thresholds are breached, while others integrate pulse-triggered semiconductor switching for finer response. More advanced methods seek continuous output-voltage correction through real-time control computation executed on the microcontroller. To support these developments, simulation environments now serve as key verification tools. Work described in [10] shows simulation of an AVR under alternative control strategies, demonstrating measurable improvements in dynamic response and deviation minimization through digitally tuned control logic. By enabling safe virtual testing of extreme or rare fault scenarios, simulation-first methodologies reduce design risk while allowing efficient iteration toward robust hardware implementation.

Although the literature includes significant progress in both hardware-controlled and simulation-verified voltage regulation systems, comprehensive integration remains limited. Many studies prioritize regulation accuracy and often overlook coordinated isolation and recovery when severe disturbances occur. Systems focusing primarily on protection seldom include dynamic correction functions or real-time user monitoring. A majority of reported prototypes depend on immediate hardware setup, which restricts scalability, increases development time, and introduces potential for component damage during early testing. Research of designs that unify sensing, digital logic, step regulation, user feedback, and automatic shutdown in a modular and simulation-verified framework remain relatively scarce.

This work addresses these limitations by developing a microcontroller-based voltage monitoring and protection system validated entirely in a virtual prototyping environment. The system detects and processes the AC input voltage through an analog-to-digital interface and evaluates the conditions against programmable reference thresholds. Relay actuation regulates the output voltage within acceptable bounds. If the supply crosses the lower or upper safety limits, the controller initiates an immediate shutdown sequence that isolates the load to avoid further stress. Real-time voltage values are presented via a display interface to ensure operational transparency. The simulation-first architecture provides a controlled platform for assessment under diverse supply scenarios without exposing hardware to premature risk. The approach supports cost-efficient development, rapid modification, and ensures functional reliability before physical deployment. Ultimately, the proposed design offers a practical pathway toward adaptive and customizable voltage stabilizers capable of addressing modern power-quality challenges.

II. Methodology

The development of the proposed programmable AVS follows a *simulation-first methodology*, in which the complete control, protection, and monitoring workflow is designed, implemented, and validated within the *Proteus* simulation environment. This approach allows early-stage verification of signal acquisition, conditioning, analog-to-digital conversion, control logic, and relay-based voltage regulation prior to physical prototyping. As a result, design inconsistencies and logic errors can be identified and corrected before hardware realization, reducing development cost and risk.

The primary objective of the proposed system is to develop a programmable reference voltage stabilizer capable of maintaining safe operating limits during both under-voltage and over-voltage conditions while simultaneously providing real-time monitoring and user feedback. The overall methodology includes system architecture formulation, AC signal sensing and conditioning, digital processing using a microcontroller, and relay-driven voltage regulation.

A. System Architecture

In research and development, the architecture of the system defines the structural organization of the system, which includes its functional blocks, hardware and software components, data flow, and interactions among subsystems. For the proposed AVS, the architecture is divided into following layers:

1) **AC input and reference setting:** A configurable AC source provides the input supply, while user-defined reference settings determine the target operating window.
2) **Signal acquisition and conditioning:** Rectification, filtering, and scaling convert the AC input into a measurable DC-level signal suitable for digital processing.
3) **Analog-to-digital conversion:** The conditioned voltage is converted into digital codes through ADC conversion to enable discrete-level decision making.
4) **Microcontroller decision logic:** The Arduino interprets the digital codes, applies threshold logic, and selects the appropriate relay state.
5) **Actuation and voltage regulation:** A relay-driver stage energizes one relay from an array of 8 relays to regulate the output voltage and enforce protection under abnormal conditions.
6) **Monitoring and user feedback:** A 16×2 LCD and a 4-digit seven-segment display provide real-time status, voltage levels, and relay activation information.

TABLE I: Major Hardware Components and Their Functions

Component	Function
Arduino Uno (ATmega328P)	Controls voltage sensing, display operation, and relay switching logic
AC power source	Provides conditioned AC input and regulated DC output
16×2 LCD	Displays input voltage, output voltage, and active relay number
8-relay array	Ensures safe operation during over-voltage and under-voltage conditions

III. Signal Flow

The signal flow diagram shown in Fig. 1 illustrates the operational sequence of the programmable voltage stabilizer. The AC input voltage is first applied to the sensing and conditioning stage, where it is scaled, rectified, and filtered. The conditioned voltage is then supplied to the analog-to-digital conversion stage.

A. AC measurement interface

A typical AC voltage source element configured for AC circuit simulations in LTspice is used. In this work, that element is listed under the voltage source category and specifies the characteristics of the source. As illustrated in Fig. 2, this forms a high-voltage AC measurement front end. Its purpose

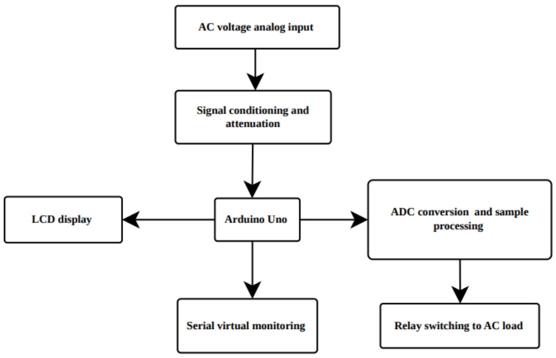

Fig. 1: Signal flow for AVS

is to take the 230VAC mains, attenuate and scale it, apply a DC offset, provide protection, and ultimately output a 0–5V waveform suitable for an Arduino ADC. Because the Arduino cannot handle negative voltages or inputs above 5V, the circuit reshapes the AC signal into a safe, centered form. On the left, the input is a 230VAC, 60Hz mains supply, which is reduced by a transformer to 6VAC RMS. This is the safe starting point for signal conditioning. After the transformer, the AC is still ±8.5V, which is too high and swings negative. In this circuit, R2, R3, and R4 set the signal amplitude, introduce a DC offset, and restrict the current flowing into the ADC input. The two BAT54 Schottky protection diodes clamp the input voltage between 0 and +5 V. If a transient attempts to drive the ADC pin beyond this safe window, the diodes switch on and shield the microcontroller. C1 attenuates high-frequency noise coming from the transformer, while C2 stabilizes the bias line so the 2.5 V reference remains steady. This reduces jitter in the ADC measurements and provides the Arduino with a clean and safe AC input waveform.

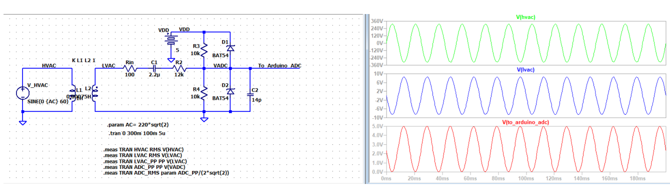

Fig. 2: AC measurement interface to Arduino

label=a.

1) The peak input voltage,

$$V_{pk} = V_{\text{RMS}} \times \sqrt{2} \tag{1}$$

$$V_{pk} = 230 \text{ VAC} \times \sqrt{2} = 325.26 \text{ V}$$

2) The transformer secondary RMS ,

$$\frac{V_p}{V_s} = \frac{N_p}{N_s} = \sqrt{\frac{L_p}{L_s}} \tag{2}$$

$$\text{V}_s = V_p\sqrt{\frac{L_s}{L_p}}$$

$$V_p = 230 \text{ V}, L_p = 1\text{H}, L_s \quad = 0.7\text{mH}, V_s = 5.99 \text{ V}, \tag{3}$$

3) The LVAC peak-to-peak voltage ,

$$V_{LVAC,pp} = 2\sqrt{2} \times V_s \tag{4}$$

$$V_{\text{LVAC,pp}} \approx 16.97 \text{ V} \tag{5}$$

4) The peak to peak ADC voltage ,

$$V_{\text{ADC,pp}} \approx 4.99 \text{ V} \tag{6}$$

5) The RMS value of ADC voltage is,

$$V_{\text{ADC,RMS}} = \frac{V_{\text{ADC,pp}}}{2\sqrt{2}} \approx 1.753 \text{ V} \tag{7}$$

So the circuit converts the mains into a small AC riding on a DC offset, and at the ADC pin that AC component has an RMS of about 1.75V.

6) The overall scaling factor ratio,

$$k = \frac{V_{\text{HVAC,RMS}}}{V_{\text{ADC,RMS}}} \tag{8}$$

$$k \approx \frac{229.90}{1.7527} \approx 131.3 \tag{9}$$

$$V_{\text{HVAC,RMS}} \approx 131.3 \cdot V_{\text{ADC,RMS}} \tag{10}$$

Thus the mains RMS is about 131 times the RMS of the AC component at the ADC pin.

B. ADC Conversion and Data Processing

An Arduino UNO, for instance, has a 10-bit analog-to-digital converter that can handle multiple channels. It takes any voltage from zero up to its operating voltage (5 volts) and converts it into a number between 0 and 1023. This means each step in voltage it can detect is about 0.0049 volts, or 4.9 millivolts.

This device has a 10-bit analog-to-digital converter, which means it can represent analog signals with digital values from 0 to 1023. This range of possible values is known as its resolution, and it tells you how many distinct levels the converter can output for the given analog input range.

Digital Output value calculation for this application,

ADC Resolution:

$$\text{Resolution} = \frac{V_{ref}}{(2^n) - 1} \tag{11}$$

Digital Output:

$$\text{Digital Output} = \frac{V_{in}}{\text{Resolution}} \tag{12}$$

Where, V_{ref} is the reference voltage representing the highest analog voltage that the ADC can measure.

For simplicity, assume $V_{ref} = 5$ V:

- When $V_{in} = 0$ V, the digital output $= 0$
- When $V_{in} = 5$ V, the digital output $= 1023$ (for a 10-bit ADC)

The voltage for a given digital code can be approximated using the following formula, where V_{ref} is the reference voltage.

$$V = \frac{\text{Decimal Value}}{(2^n) - 1} \times V_{ref} \tag{13}$$

The Arduino Uno microcontroller runs the program stored in its flash memory. It applies logic, calculations, or conditions to the sampled data.

Now for coding the sampling points are taken over several cycles and the point should be considered as max ADC count: Nmax min ADC count: Nmin

To convert counts to peak-to-peak voltage, the equation becomes

$$V_{\text{ADC,pp}} = \left(\frac{N_{\max} - N \min}{1023} \right) \cdot V_{\text{ref}} \tag{14}$$

The peak-to-peak value of a sine wave can be transformed into its RMS value.

$V_{\text{ADC,RMS}} = \frac{V_{\text{ADC,pp}}}{2\sqrt{2}}$

After combining, the equation becomes ,

$$V_{ADC,RMS} = \frac{1}{2\sqrt{2}} \left(\frac{N_{max} - N_{min}}{1023} \right) V_{ref}$$

The final expression for converting mains RMS voltage to ADC counts is:

Now, by substituting $V_{ADC,RMS}$ into the scaling factor, we obtain:

$$V_{HVAC,RMS} \approx 131.3 \cdot V_{ADC,RMS}$$

$$V_{HVAC,RMS} \approx 131.3 \cdot \frac{1}{2\sqrt{2}} \left(\frac{N_{max} - N_{min}}{1023} \right) V_{ref}$$

If $V_{ref} = 5$ V:

$$V_{HVAC,RMS} \approx 131.3 \cdot \frac{5}{2\sqrt{2} \cdot 1023} \cdot (N_{max} - N_{min})$$

It is possible to calculate the constant in advance:

$$C = \frac{131.3 \cdot 5}{2\sqrt{2} \cdot 1023} \approx 0.226$$

So a very practical formula is:

$$V_{HVAC,RMS} \approx 0.226 \cdot (N_{max} - N_{min})$$

IV. Proteus Simulation Setup

This Proteus simulation represents a complete 8-channel relay control system built around an Arduino, integrating power conversion, signal isolation, transistor-based relay driving, and a user interface. A step-down transformer with rectification and filtering components generates the DC supply for the logic circuitry, while the Arduino manages all control logic and interacts with the user through a keypad and an LCD display. Each of the Arduino's eight digital outputs feeds an optocoupler, providing electrical isolation between the low-voltage microcontroller side and the higher-current relay driver stage. The optocouplers activate BC547 transistors that switch the relay coils, with flyback diodes protecting the transistors from inductive voltage spikes. The relays themselves are capable of switching AC loads, making the system suitable for home automation or multi-device control. Overall, the circuit demonstrates a safe and modular design that separates user input, logic control, and high-voltage switching while ensuring reliable operation through proper isolation and protection components.

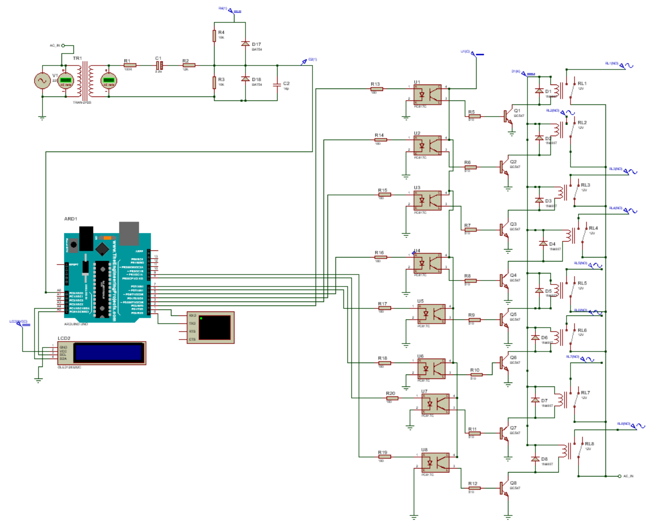

Fig. 3: AVR set up in Proteus professional 8

A. Source Code Debugging

Proteus supports source code debugging through its source code control system. This system's primary role is to edit and assemble source code while ensuring that any changes are applied promptly. It includes a source code (source program) editor, an assembler, a debug data extractor (DDX), and a loader. The DDX retrieves debugging information from the file produced by the assembler and passes it to the loader. The general procedure for source code debugging is as follows: create a source code file, load it into the system, choose a microcontroller and an assembler, load the target code generated by assembling the source code into the selected microcontroller, and then start the simulation to perform source code debugging. After assembly, HEX or other debugging files can be generated.

The program implements an 8-relay automatic voltage stabilizer using an Arduino Uno, an I2C LCD, and AC voltage sensing signal conditioner. It continuously measures the incoming AC voltage by sampling the analog input 200 times, removing the DC offset, and calculating the RMS value using a calibration factor. Based on this measured voltage, the code selects one of eight relay "taps," each associated with a predefined voltage threshold. A small hysteresis margin prevents rapid switching when the voltage fluctuates near a threshold. Only one relay is allowed to be active at a time, and a mandatory delay between switching ensures safe operation and protects the relays from rapid toggling. The LCD displays the measured voltage and relay activity, while the serial monitor outputs the same information for debugging. Overall, the code provides a complete control loop that reads AC voltage, determines the appropriate correction tap, and safely switches relays to stabilize the output voltage.

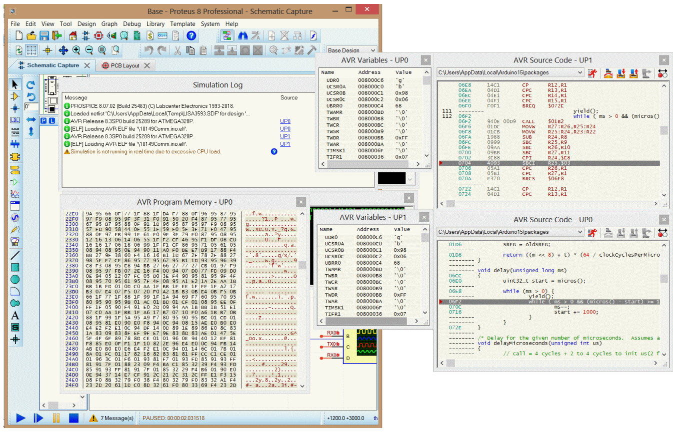

Fig. 4: Debug control panel

V. Conclusion

This work presented a simulation-driven design and validation of a microcontroller-based programmable AVS using the Proteus environment. The proposed system integrates AC voltage sensing and conditioning, analog-to-digital conversion, embedded decision logic, and relay-based actuation within a unified architecture. By employing a simulation-first methodology, the complete control and protection workflow was verified prior to hardware implementation. This enables accurate assessment of voltage regulation behavior, relay switching logic, and protection thresholds under under-voltage and over-voltage conditions. Real-time monitoring through display interfaces further enhances operational transparency and supports effective debugging and verification. The results demonstrate that virtual prototyping provides a reliable and cost-efficient pathway for developing programmable voltage stabilization and protection systems.

Future work will focus on physical hardware implementation and experimental validation under real-world grid disturbances, which include voltage sags, swells, and transient events. Additional enhancements may include finer-granularity regulation using adaptive thresholding, integration of closed-loop control strategies, and replacement of relay-based switching with solid-state devices to improve response speed and durability. The proposed architecture can also be extended to support data logging, remote monitoring, and intelligent control schemes, to make it suitable for modern power-quality applications in residential, commercial, and industrial environments.

Acknowledgment

References

[1] M. H. J. Bollen, *Understanding Power Quality Problems: Voltage Sags and Interruptions*. IEEE Press, 2000.

[2] P. Kundur, *Power System Stability and Control*. McGraw-Hill, 1994.

[3] N. G. Hingorani and L. Gyugyi, *Understanding FACTS: Concepts and Technology of Flexible AC Transmission Systems*. Wiley-IEEE Press, 1995.

[4] D.-H. Jang and G.-H. Choe, "Step-up/down ac voltage regulator using transformer with tap changer and pwm ac chopper," *IEEE Transactions on Industrial Electronics*, vol. 45, no. 6, pp. 905–911, 1998.

[5] J. V. Lopez, S. M. Garcia, J. C. C. Rodriguez, R. V. Garcia, S. M. Fernandez, and C. C. Olay, "Electronic tap-changing stabilizers for medium-voltage lines optimum balanced circuit," *IEEE Transactions on Power Delivery*, vol. 27, no. 4, pp. 1909–1918, 2012.

[6] C. Chaudhary, A. Kshetri, A. Shrestha, A. Bista, and D. Bista, "Design and fabrication of automatic voltage stabilizer using relay-based tap changing of transformer," in *Proc. Current Research in Hydropower Technologies 2023 (CRHT-XI), Journal of Physics: Conference Series 2629*, 2023.

[7] B. K. Bose, *Power Electronics and Motor Drives: Advances and Trends*. Academic Press, 2006.

[8] G. Ofualagba and C. Igbinoba, "Implementation of a microcontroller based 5 kva automatic voltage stabilizer," *International Journal of Latest Research in Engineering and Technology*, pp. 01–08, 2017.

[9] M. Y. Alnaham and M. M. Suliman, "Microcontroller-based system for voltage monitoring, protection and recovery using proteus vsm software," *International Journal of Computer Applications*, vol. 118, no. 3, 2015.

[10] A. Yusre, S. Samuji, N. S. nd N. Saidin, and N. Azhuan, "Modeling and simulation of avr systems using matlab/simulink: Performance comparison of pid and ann controllers," *Journal of Power and Energy Engineering*, vol. 13, pp. 315–325, 2025.