Hardware Implementation and Validation of a dc Protection Scheme Based on Local Measurements

Daniel Zintsmaster III, *Student Member, IEEE*, Munim Bin Gani, *Member, IEEE*, Sukumar Brahma, *Fellow, IEEE*, Matthew J. Reno, *Senior Member, IEEE*, Miguel Jimenez-Aparicio, *Member, IEEE* and Javier Hernandez-Alvidrez, *Member, IEEE*

***Abstract*—Direct current (dc) protection hardware must measure, process, and respond to faults in tens of microseconds. This paper validates that modern microcontrollers can execute a dc protection scheme within this strict constraint. The hardware implementation of a novel protection algorithm is designed and evaluated in a real-time, hardware-in-the-loop (HIL) environment, specifically addressing the challenges involved in transitioning from theory to a real hardware prototype.**

***Index Terms*—dc distribution, microgrid, fault, protection.**

I. Introduction

DC microgrids are gaining popularity as an alternative to ac distribution grids due to advantages in efficiency and ease of control when integrating inverter-based generation [1], [2]. However, dc fault currents lack a natural zero-crossing point and rapidly surpass circuit breaker limits over tens of microseconds [3]. Power electronics used within a dc microgrid are susceptible to damage caused by these fault currents [4], [5]. These limits present the need for protection schemes that respond quickly, selectively, and without communication. To the authors' knowledge, this work presents the first microcontroller-based hardware implementation of a single-ended fault location algorithm that samples and executes fault logic at a 1 MHz rate. This sampling rate presents a significant computational burden, requiring measurement and response to microsecond transients before safe limits are exceeded. Developments [7], [8] show that although dc microgrid protection is a relevant topic, the relay response speed constraints are so stringent that any post-processing introduces latency and heavily impacts trip speed. Other methods [9]–[13] operate properly in external hardware, but rely on communication, post-processing, or machine learning methods that impede the rapid, single-ended execution speed necessary for this application. To advance the state of the art in dc feeder protection, this paper implements a local measurement-based low-voltage dc (LVdc) protection scheme on a microcontroller.

This work was supported by Sandia National Laboratories through the U.S. Department of Energy's Office of Energy Efficiency and Renewable Energy(EERE) under the Solar Energy Technologies Office Agreement Number 36533 - subaward to Clemson University and by Clemson University Electric Power Research Association (CUEPRA). Sandia National Laboratories is a multimission laboratory managed and operated by National Technology and Engineering Solutions of Sandia, LLC, a wholly owned subsidiary of Honeywell International Inc., for the U.S. Department of Energy's National Nuclear Security Administration under contract DE-NA0003525.

S. Brahma, M. Gani, and D. Zintsmaster are with the Department of Electrical & Computer Engineering of Clemson University (*sbrahma@clemson.edu, mgani@clemson.edu, dzintsm@clemson.edu*). M. Reno, M. Jimenez-Aparicio, and J. Hernandez-Alvidrez are with Sandia National Laboratories.

979-8-3315-8138-1

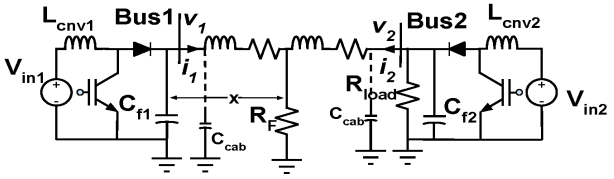

Fig. 1: Two bus dc system fed by converters

The scheme uses a single-ended fault location theory [6] that determines fault location using only three post-fault samples and is immune to fault resistance. Sampling at 1 MHz is necessary to accurately calculate di/dt. The paper demonstrates this relay in a real-time HIL environment, building on preliminary hardware results in [14] by including further filter analysis, addition of a third processing core to the pipelining process, and dissection of latency sources, quantization, and scaling effects at protection boundaries. The results and analyses aim to provide a benchmark for future commercial hardware designs.

II. Theory of Novel dc Protection Scheme

The system is a 200-meter, two-bus dc feeder with 20 kHz switching converters at each end (Fig. 1). Details are provided in [14].

$$x(t_2) = \frac{\begin{vmatrix} v_1(t_1) & i_1(t_1) - I_0 \\ v_1(t_2) & i_1(t_2) - I_0 \end{vmatrix}}{\begin{vmatrix} l \cdot \dfrac{di_1(t)}{dt}\Big|_{t=t_1} + r \cdot i_1(t_1) & i_1(t_1) - I_0 \\ l \cdot \dfrac{di_1(t)}{dt}\Big|_{t=t_2} + r \cdot i_1(t_2) & i_1(t_2) - I_0 \end{vmatrix}} \quad (1)$$

The algorithm uses (1) to determine fault location x from Bus 1. Inputs are three sequential post-fault voltage (v_1) and current (i_1) samples, line parameters (r, l), and pre-fault current (I_0). The equation is closed-form and independent of fault resistance R_F. Under normal conditions, x is negative or $\gg L$. Upon fault, x converges to the location within three samples. For security, the relay requires five consecutive distance results within the zone ($0 < x < L$). This prevents false trips caused by "junk" values that can appear when the pre-fault data has not been completely flushed out of the fault location equation.

In practice, the microcontroller executes a continuous loop calculating x as well as enacting the trip logic. It increments a counter if x is in-zone, tripping only after five sequential counts. The protection scheme relies on accurate x results to perform the relay logic successfully, and accurate x calculation

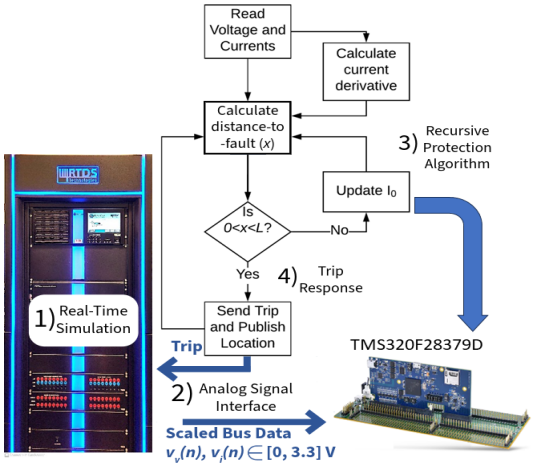

Fig. 2: Overview of HIL implementation

relies on quality inputs, assuming effective noise attenuation to capture the transient [6] before limits are exceeded.

III. Hardware Implementation of Protection Scheme in Real-Time HIL Environment

The hardware is the TI TMS320F28379D microcontroller [15] (DSP board), featuring multiple independent analog-to-digital (ADC) channels and a control law accelerator (CLA) to perform calculations in parallel with the main CPUs. Pipelining processes across dual CPUs and the CLA allows the designed firmware to execute the complete relaying algorithm within the 1 μs interval.

A. HIL Configuration with RTDS

The HIL environment (Fig. 2) validates the DSP performance by connecting it to a real-time simulation, emulating a real analog signal environment. The process involves: 1) Simulating the circuit on the Real Time Digital Simulator (RTDS); 2) Wiring RTDS analog output (AO) signals to hardware; 3) Executing the protection algorithm; 4) Sending the trip signal back to RTDS, completing the loop. The circuit (Fig. 1) is simulated at a 1 μs time step to match the relay's sampling rate. To avoid "time-step overflow" errors, the simulation was divided across multiple RTDS cores, introducing 1 μs time step of inter-core latency between each substep box. The scheme uses only local measurements, ensuring scalability. A software version of the relay was also simulated in one core of the RTDS for direct comparison with the presented hardware implementation.

B. Simultaneous Input Waveform Sampling and Filtering

To simultaneously sample voltage and current, two distinct ADC modules on the DSP were configured to sample in response to the global system clock initiating a "start of conversion" (SOC) flag every microsecond. The identically configured ADC peripherals both complete their sampling, holding, and quantization tasks in the same number of clock cycles before the next SOC flag is set, ensuring time alignment across both channels. Analog samples are quantized at 12-bit resolution, providing $4096 (= 2^{12})$ quantization levels that divide the 3.3V ADC input range into 0.8mV steps. The DSP does possess a 16-bit differential ADC module. However, it requires a floating input common-mode voltage that is explicitly incompatible with the grounding scheme of this experiment configuration [15]. To use the 16-bit mode, additional hardware complexity would be required to shift this common-mode voltage, and was determined to be outside of the scope of this prototype. The 16-bit ADC also requires a greater number of clock cycles to process each sample which impact the algorithm's distribution across the processors. In practical implementations, noise is introduced

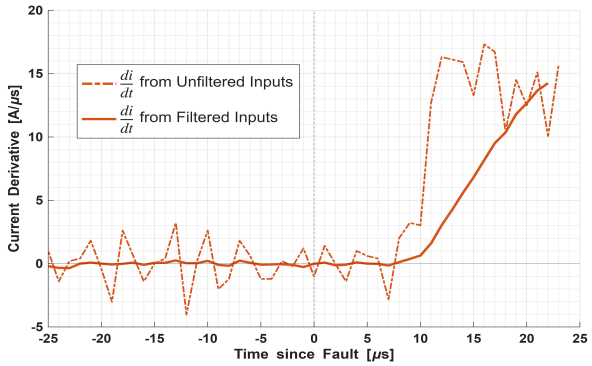

Fig. 3: Digitized values calculated di/dt from unfiltered current input for a 100m fault

from DSP quantization error, analog artifacts, and clock-phase mismatch between RTDS AO and DSP ADC modules. High-frequency noise impacts the raw samples and the calculation of di/dt (Fig. 3). To address the impact of noise, a digital low-pass (LP) filter must be applied to the input data. Applying a standard digital filter to these signals presents a unique challenge, as classical digital filters are designed to operate on periodic signals, not the exponential characteristics of dc fault currents. Therefore, filter tuning required an iterative process to balance noise attenuation with minimum group delay, along with the restriction of whether the microcontroller could perform the required constant coefficient difference equation (3) within a single microsecond operation cycle. Multiple filters were tested and implemented on the software relay to gain an understanding on how an ideal filter would modify the signals input to the relay and then passed to the distance-to-fault equation (1). The result of this process was a 12th-order LP finite impulse response (FIR) filter. Unlike an infinite impulse response (IIR) filter used in previous versions of this hardware relay, the chosen FIR filter does not rely on previously calculated outputs. The general recursive form of an IIR filter is shown in (2), where the filtered output ($\hat{y}[n]$) is dependent on the raw input ($x[n]$), the filter coefficients (α_k, β_k), and the filter poles (p) and zeros (q).

$$\hat{y}[n] = \sum_{k=0}^{q} \beta_k x[n-k] - \sum_{k=1}^{p} \alpha_k \hat{y}[n-k] \tag{2}$$

In contrast, the FIR filter implemented in this work removes the feedback term, as defined by (3). Here, the filtered prediction ($\hat{y}[n]$) depends solely on the weighted sum of current and past inputs ($x[n]$) defined by the coefficients (β_k) and the

filter order (N).

$$\hat{y}[n] = \Sigma_{k=0}^{N} \beta_k x[n-k] \tag{3}$$

Due to this lack of a "lagging average," FIR filters can quickly flush out the pre-fault samples when compared to its IIR counterpart. The reduction in latency is clearly seen in Fig. 4. Applying a linear digital filter to a non-periodic transient signal inherently distorts the resulting time-domain data. This fact is relevant because the accuracy of fault location theory entirely depends on the samples just after fault, which are the measurements that are most impacted by the filter's smearing effect. To the authors' understanding, the inability for simple digital filters to pass exponential signals means that this error is inevitable.

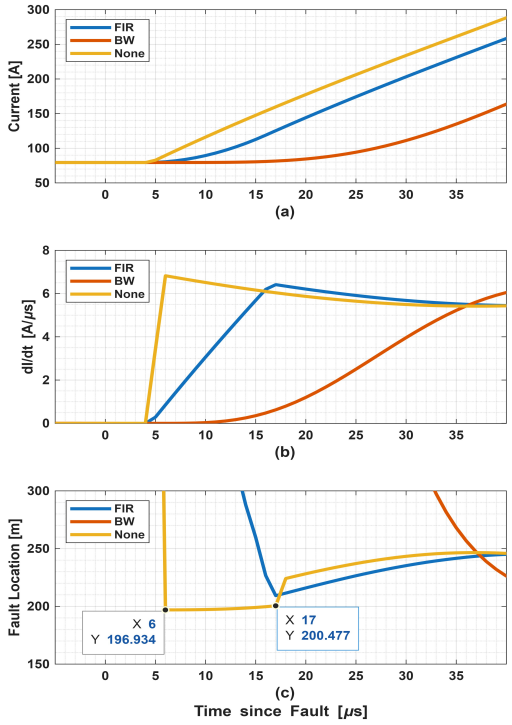

Fig. 4: Impact of filters on the simulated relay results for far end (197m) fault case (a) current, (b) $didt$, (c) distance-to-fault calculation

C. Input Scaling and Calculation

The RTDS AO module outputs analog values over a range of $\pm 10V$ [16], but the DSP's ADC only accepts 0V to 3.3V [17]. This dynamic range mismatch is resolved through analog scaling settings of the RTDS and an external voltage divider. Mapping a large fault current range into the narrow input range introduces a trade-off between the measurable range and the resolution of a signal. The input scaling limit of 800 A was selected to minimize clipping in the near-end fault cases, and quantization noise in the far end fault cases.

D. Multi-Core Pipelined Execution

The protection algorithm operates as a continuous, synchronous loop that begins by obtaining the filtered voltage and current values. Then, the relay approximates di/dt, calculates the distance-to-fault x using (1), and validates whether the result lies within the protected zone ($0 < x < L$). To meet the strict < 1 μs execution constraints, these tasks are pipelined across the parallel processors, as shown in Fig. 5. The CLA handles sampling and filtering, and the main CPUs (CPU1 and CPU2) execute the serial algorithm steps. CPU1 computes the derivative di/dt and updates the pre-fault current (I_0) during normal operation, while CPU2 solves for the fault location (x) and executes the trip logic. This configuration ensures that the individual routines all take less than 1 μs so that the entire process achieves the desired 1 μs throughput.

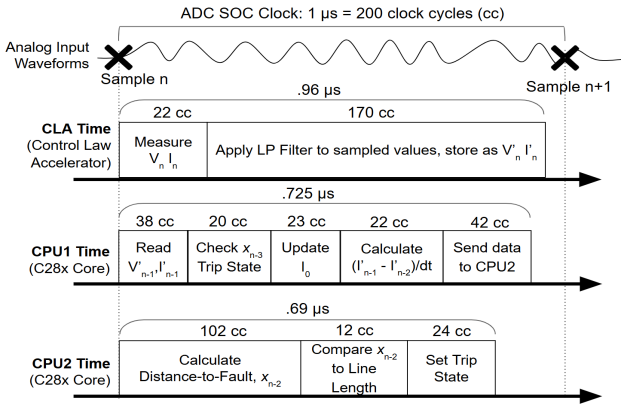

Fig. 5: Hardware task distribution across parallel processors

IV. HIL Latency and Performance

The total latency between the moment of fault initiation and the RTDS receiving a "trip" signal is made up of multiple sources. These delays vary depending on the implementation: the ideal unfiltered simulated relay; the filtered noiseless simulated relay; and the hardware relay. Certain latency sources are common to all of the relays. These sources include the RTDS inter-core latency ($t_{c2c} = 3$ μs), caused by the transfer of data across the control, circuit simulation, and analog interface cores. Inter-core latency is a result of the real-time simulation environment and would not exist in a real deployment scenario. The other consistent latency source is the algorithm's inherent solution ($t_{solve} = 3$–4 μs), plus the time required to accumulate five sequential in-zone fault counts ($t_{count} = 5$ μs). The next delay source is found only in the relay implementations that include the digital FIR filter. This latency is a result of the filter's linear group delay, in which a $\frac{N-1}{2}$ delay is added between the raw current inputs and the filtered current data, where N is the filter order. These delays can be observed by viewing the real-time current waveforms measured by the software and hardware relay with respect to the fault initiation moment (Fig. 6). However, as shown in Fig. 4(a) and (c), the constant group delay in the current signal does not translate to an equivalent constant delay in the fault distance calculation or the final trip decision. The effective filter latency (t_{filt}) was quantified as the difference between the trip times of the filtered simulation (t_{sim}) and the ideal unfiltered simulation (t_{ideal}): $t_{filt} = t_{sim} - t_{ideal}$. Remaining sources of latency were introduced by the pipelined hardware algorithm latency and HIL configuration. Pipelining adds 3 μs of latency to ensure microsecond cycle throughput,

plus an extra cycle for setting the trip output pin high after a fault is detected, totaling $t_{Alg} = 4$ μs. The HIL latency ($t_{HIL} = 5$–7 μs) accounts for the time required to write and read the digital data across devices, and is made up of the DAC/ADC conversion times and clock synchronization of the isolated RTDS and DSP.

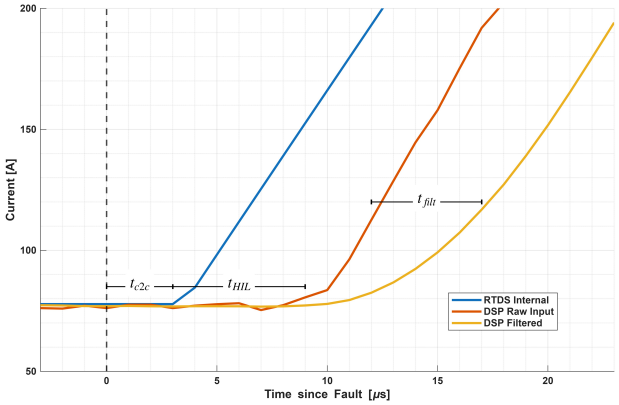

Fig. 6: Real time current waveforms that show the data propagation through the HIL configuration during a 100m fault

V. Results

A. Hardware and Software Relay Comparison

The fault location results obtained by the internal software relay and the external hardware relay were compared within the HIL environment (Fig. 7). The relevant time frame starts immediately before the fault initiation and ends immediately after the external hardware trip signal is received. When identical filtering is applied, the hardware and software relays exhibit similar distance-to-fault results, with a 9 μs latency between the trip signal of each relay, attributable to the HIL interface and hardware pipelining. The hardware relay successfully achieves trip speeds comparable to the software simulation, validating the DSP's ability to execute the complete fault location calculations and protection algorithm within the required 1 μs window. Discrepancies in the final distance-to-fault results are expected, as the hardware relay is subject to the filtering and DAC/ADC process, while the RTDS is using ideal noise-free data. Once the relay logic determines that a fault is within the zone of protection for the required duration, it prioritizes the trip command. As a protection device, the relay ceases calculation to actuate the breaker immediately. Consequently, the "final" distance calculation reported in Fig. 7 and Fig. 8 represents the value at the moment of the trip decision. Delaying the trip to allow for further convergence would compromise protection speed without guaranteeing a significantly more accurate location estimate.

B. Various Fault Distance Scenarios

The hardware relay was tested across a range of fault locations and successfully identified and located all faults within 95% of the protection zone. The relay correctly did not operate for out-of-zone faults, as no distance-to-fault calculations fell within the line length for external faults. For faults occurring within the final 5% of the protection zone (near the 200m boundary),

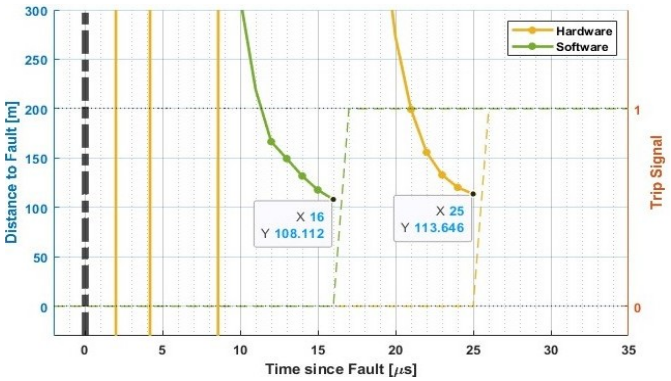

Fig. 7: Fault location and resulting trip signal produced within the HIL configuration for fault at 100m

the relay exhibited inconsistent tripping performance within the 35 μs window. This boundary effect is detailed in Section V-C and is a consequence of the HIL signal chain constraints, rather than a flaw in the firmware design. Fig. 8 illustrates the trajectory of the five sequential distance-to-fault calculations leading to the trip decision. For all successful operations, the relay confirmed the fault and issued a trip signal in less than 35 μs after inception, satisfying the design requirements.

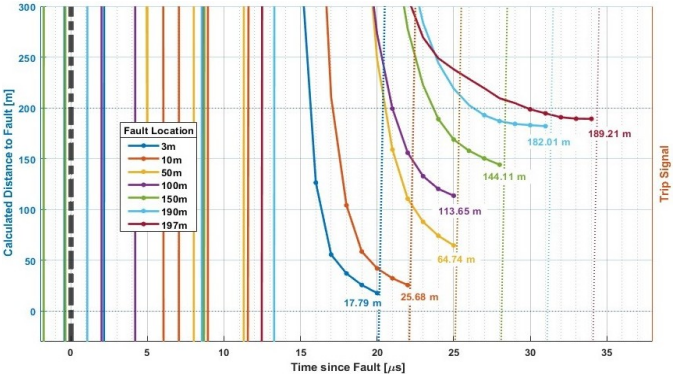

Fig. 8: Distance-to-fault results from hardware relay at trip instance for various faults locations along the test feeder

C. Boundary Performance and Quantization Effects

While the hardware relay exhibits expected performance for a majority of faults (3m to 190m), boundary faults near the 200m zone limit reveal the practical precision limits of the prototype. For a fault at 197m, the calculated distance value x correctly transitions from an undefined value to a result near the boundary. However, as the distance from the bus increases, the magnitude of the current derivative (di/dt) decreases, reducing the denominator of the fault location equation 1 and increasing the result's sensitivity to quantization noise from the 12-bit ADC. The 0.8mV quantization steps are scaled to meet the entire range of current magnitudes that the relay could obtain. When this scale is widened to allow accurate measurement of very high ($>$ 2 kA) current magnitudes in near-end cases, the distance-to-fault result fluctuates across the 200m protection threshold in far-end cases. The current trip logic requires five consecutive counts for security, where the momentary excursions outside the zone reset the counter, delay the trip, and cause the relay to miss the window where

the fault location is most accurate (e.g., the stable period between 6 μs and 17 μs post-fault shown in Fig. 4(c)). This highlights a design trade-off: a higher counter threshold improves security and effectively removes "junk" values, but reduces dependability at the edge of the protection zone under high-noise conditions. Future iterations could mitigate these challenges by improving the ADC resolution, input signal dynamic range, and a system-specific, zone-based trip logic.

VI. Conclusion

This paper presents the design, implementation, and validation of a high-speed dc protection relay capable of 1 MHz sampling and processing. The prototype utilizes a multi-core microcontroller to sample analog voltage and current, apply FIR filtering, and execute a novel single-ended fault location algorithm, all within a strictly enforced 1 μs loop.

A review of current literature and commercial technology suggests that this is one of the first hardware implementations to achieve a complete protection logic cycle within this time constraint. This work bridges the gap between theory and practice, validating that the high sampling rates and derivative-based methods often proposed in simulation are achievable on commercial hardware. Experimental validation in a real-time HIL environment demonstrated the relay's dependability, security, and speed. The system successfully detected and cleared all in-zone faults in less than 35 μs while remaining in normal operation for out-of-zone faults and non-fault conditions.

However, the study also highlights that such an implementation is non-trivial. The results quantify the significant impact of signal processing on transient-based fault location, specifically exposing the limitations of applying classical linear filters to the non-periodic, exponential characteristic of dc faults.

Future work will focus on addressing the boundary performance limitations identified in this study. Objectives include developing specialized filter tuning methods and improving ADC modules to maximize dynamic range and minimize quantization noise. Additionally, further research is required to refine zone-boundary logic and establish system-specific post-trip coordination procedures.

References

[1] H. Kakigano, Y. Miura, and T. Ise, "Low-voltage bipolar-type DC microgrid for super high quality distribution," *IEEE Transactions on Power Electronics*, vol. 25, no. 12, pp. 3066–3075, Aug. 2010.

[2] P. H. Gadde, M. Bin Gani, and J. H. Enslin, "An AC/DC hybrid campus microgrid: Modelling, control and financial analysis," in *2021 IEEE 12th International Symposium on Power Electronics for Distributed Generation Systems (PEDG)*, July 2021, pp. 1–7.

[3] L. Kong, H. Nian, Q. Huang, Z. Zhang, J. Xu, and K. Dai, "Optimization of current breaker and fault current limiter in DC micro-grid based on faulty transient analysis," in *2018 21st International Conference on Electrical Machines and Systems (ICEMS)*, Oct. 2018, pp. 2017–2022.

[4] A. Meghwani, S. C. Srivastava, and S. Chakrabarti, "Local measurement-based technique for estimating fault location in multi-source DC microgrids," *IET Generation, Transmission Distribution*, vol. 12, no. 13, pp. 3305–3313, July 2018.

[5] R. Bhargav, B. R. Bhalja, and C. P. Gupta, "Novel fault detection and localization algorithm for low-voltage DC microgrid," *IEEE Transactions on Industrial Informatics*, vol. 16, no. 7, pp. 4498–4511, July 2020.

[6] M. Bin Gani and S. Brahma, "A closed-form mathematical model and method for fast fault location on a low voltage DC feeder using single-ended measurements," *IEEE Open Access Journal of Power and Energy*, vol. 9, pp. 523–536, Jan. 2022.

[7] E. W. Nahas, D. A. Mansour, H. A. A. el-Ghany, and M. M. Eissa, "Developing a smart power-voltage relay (SPV-relay) with no communication system for DC microgrids," *Electric Power Systems Research*, vol. 187, p. 106432, Oct. 2020.

[8] X. Wei, G. Zou, C. Chen, S. Zhang, C. Zhou, and S. Song, "Single-ended DC line protection scheme based on first peak time of current-limiting reactor voltage," *International Journal of Electrical Power & Energy Systems*, vol. 150, p. 109100, Aug. 2023.

[9] J. Naik, S. Dhar, and P. K. Dash, "Adaptive differential relay coordination for PV DC microgrid using a new kernel based time-frequency transform," *International Journal of Electrical Power & Energy Systems*, vol. 106, pp. 56–67, Mar. 2019.

[10] S. K. Paruthiyil, A. Bidram, M. J. Aparicio, J. Hernandez-Alvidrez, and M. J. Reno, "Hardware implementation of a traveling wave protection device for DC microgrids," in *2023 IEEE Kansas Power and Energy Conference (KPEC)*, Apr. 2023, pp. 1–5.

[11] M. Jimenez-Aparicio, M. J. Reno, and J. Hernandez-Alvidrez, "Fast traveling wave detection and identification method for power distribution systems using the discrete wavelet transform," in *2023 IEEE Power & Energy Society Innovative Smart Grid Technologies Conference (ISGT)*, Jan. 2023, pp. 1–5.

[12] N. Bayati and A. Hajizadeh, "Protection in DC microgrids: a comparative review," *IET Smart Grid*, vol. 1, no. 3, pp. 66–75, Sep. 2018.

[13] C. Srivastava and M. Tripathy, "DC microgrid protection issues and schemes: A critical review," *Renewable and Sustainable Energy Reviews*, vol. 151, Nov. 2021.

[14] D. Zintsmaster, M. Bin Gani, S. Brahma, M. J. Reno, M. Jimenez-Aparicio, and J. Hernandez-Alvidrez, "Hardware implementation and validation of a DC protection scheme based on local measurements," in *2024 IEEE Texas Power and Energy Conference (TPEC)*, Feb. 2024, pp. 1–6.

[15] *TMS320F2837xD Dual-Core Real-Time Microcontroller Datasheet*, Texas Instruments, Dallas, TX, USA, 2023, sPRS880P.

[16] *RTDS Control Components Manual*, [Online]. Available: https://www.rtds.com, RTDS Technologies, 2019, version 20191125.

[17] *TMS320F2837xD Dual-Core Microcontrollers Technical Reference Manual*, Texas Instruments, Dallas, TX, USA, 2019, sPRUHM8I.