Toroid Signal Processing Container with Redis Integration

Elliot Bowers
Cabrillo College
Advisors: Michelle A. Ibrahim and Jose R. Berlioz Rivera
PUB-25-0511-AD-STUDENT

July 23, 2025

Abstract

This report presents the research, design, and implementation of a Toroid Signal Processing Container with Redis Integration, conducted during my internship assignment. The project focused on developing a modular signal processing system capable of performing baseline correction, droop compensation, and real-time data communication for beam diagnostic signals. Using Redis as a message broker and configuration manager, the system ensures modularity, scalability, and efficient inter-process communication. Signal correction algorithms, such as Asymmetric Least Squares (ALS) baseline smoothing and a numerical droop correction approach, were implemented and tested using Linac beam data. The results confirm that this system improves signal fidelity and supports real-time analysis requirements, offering valuable contributions to the field of beam instrumentation and data acquisition systems.

This manuscript has been authored by FermiForward Discovery Group, LLC under Contract No. 89243024CSC000002 with the U.S. Department of Energy, Office of Science, Office of High Energy Physics.

Contents

1 Introduction

Precise beam current measurements are fundamental in accelerator physics experiments and industrial applications. The toroid current transformer, a non-invasive beam diagnostic device, generates raw voltage signals proportional to the beam current. However, these signals often suffer from baseline drift, low-frequency droop, and high-frequency noise, complicating accurate measurement and analysis. Originally, these issues were corrected directly on the FPGA. However, a server-side solution, such as this project, allows access to more complex Python digital signal processing (DSP) libraries.

The Toroid Signal Processing Container project addresses these challenges by implementing a software-based signal correction pipeline that can be easily integrated into larger diagnostic systems. A critical feature of the project is the use of Redis (Remote Dictionary Server) as a high-performance in-memory data store for handling inter-process communication and configuration management. The project objectives included:

- Developing algorithms for baseline correction and droop compensation suitable for real-time processing.
- Establishing a Redis-based architecture for modular signal processing.
- Validating the system's performance through real Linac data.

This report documents the methodology, system architecture, algorithm design, and testing results, structured according to the American Institute of Physics (AIP) style guidelines.

2 Project Objectives

The primary objectives of the Toroid Signal Processing Container with Redis Integration project were:

- Baseline Detection and Correction: Implement an algorithm that can identify and correct for the baseline of a toroid signal under varying noise conditions.
- Droop Correction: Develop a method to compensate for droop effects resulting from AC coupling or inherent system response.
- Modular Communication: Utilize Redis for reliable, scalable, and low-latency communication between different processing modules.
- Performance Validation: Quantitatively assess the system's accuracy and reliability using controlled experiments.

The project also aimed to ensure flexibility in deployment, allowing integration with existing data acquisition systems and potential adaptation to various diagnostic contexts. Providing configurable parameters for signal correction ensured that the system could be tailored to different signal conditions and acquisition hardware.

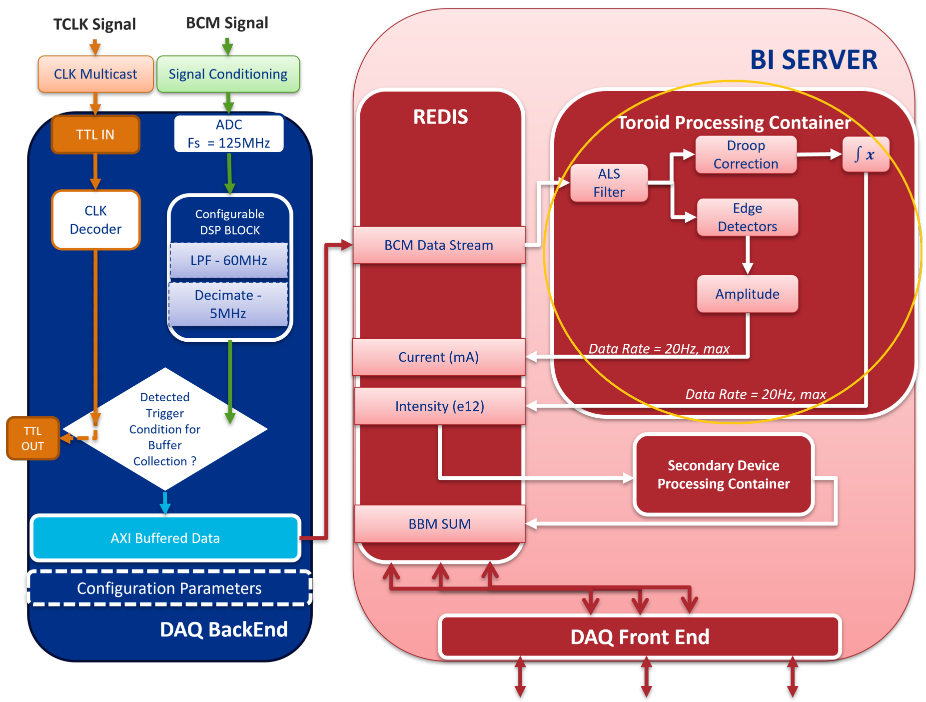

Figure 1: Full Toroid Processing Pipeline with this project circled in yellow.

3 Methodology

3.1 Signal Processing Workflow

The signal processing workflow was designed with a modular approach, where each stage functions as an independent process that communicates via Redis. The architecture comprises:

- Data Acquisition Module: Interfaces with data sources, providing real-time raw toroid signal data.
- Baseline Correction Module: Processes incoming signals to remove baseline drift using Asymmetric Least Squares smoothing.
- Droop Correction Module: Applies a numerical droop correction algorithm based on convolution and integration methods.
- Redis Integration Layer: Manages data transfer and configuration using Redis streams and key-value storage.

This design facilitates independent development, testing, and deployment of each module, promoting a flexible and maintainable system architecture. Several of the signal processing functions utilize functions contained in the Python SciPy library.[1]

3.2 Baseline Correction Using Asymmetric Least Squares (ALS)

Baseline drift is a common issue in signal processing, often due to environmental factors, electronic offsets, or slow system response. The Asymmetric Least Squares (ALS) smoothing algorithm, introduced by Eilers (2003), offers a robust method for baseline estimation.[2]

The ALS algorithm minimizes the following cost function:

$$S = \sum_{i=1}^{n} w_i (y_i - z_i)^2 + \lambda \sum_{i=2}^{n} (z_i - z_{i-1})^2$$

Where:

- y_i is the observed signal,
- z_i is the estimated baseline,
- w_i is a weight factor (adjusted asymmetrically).
- λ is the smoothing parameter controlling the balance between fidelity and smoothness.

By iteratively adjusting the weights w_i based on the residuals $y_i - z_i$, the ALS algorithm effectively suppresses peaks while fitting a smooth baseline. This algorithm was implemented with configurable parameters, allowing tuning based on the characteristics of the signal being processed.

3.3 Droop Correction Algorithm

Signal droop typically results from capacitive coupling or high-pass filter characteristics in signal acquisition systems. A droop effect manifests as a gradual decrease in signal amplitude over time, distorting true pulse shapes.

The implemented droop correction method models the distortion as a convolution with an exponential decay kernel, expressed as:

$$H(t) = \frac{1}{\tau} e^{\frac{-t}{\tau}}$$

Where τ is the time constant associated with the droop effect. The correction involves a numerical integration process to deconvolve the droop kernel from the signal, restoring the true amplitude and shape.[3]

Careful tuning of the τ parameter was essential to balance correction strength against noise amplification. The algorithm was tested across a range of synthetic signals with varying droop characteristics to ensure its general applicability.

3.4 Redis Communication Architecture

Redis is used as a middle layer between the front end and ACNET. The architecture uses:

- Streams: For real-time data streaming between modules.
- Key-Value Storage: For sharing configuration parameters and system states.

Messages are JSON-encoded for interoperability, with defined schemas for signal data and control commands. This setup enables dynamic configuration changes and system scalability. Redis's widespread support in multiple programming languages and robust community support further justified its selection for this project.[4]

4 Results

The plots below show the raw toroid data in blue, and the post-processing data in purple (after applying the mode filter, ALS, and droop correction functions).

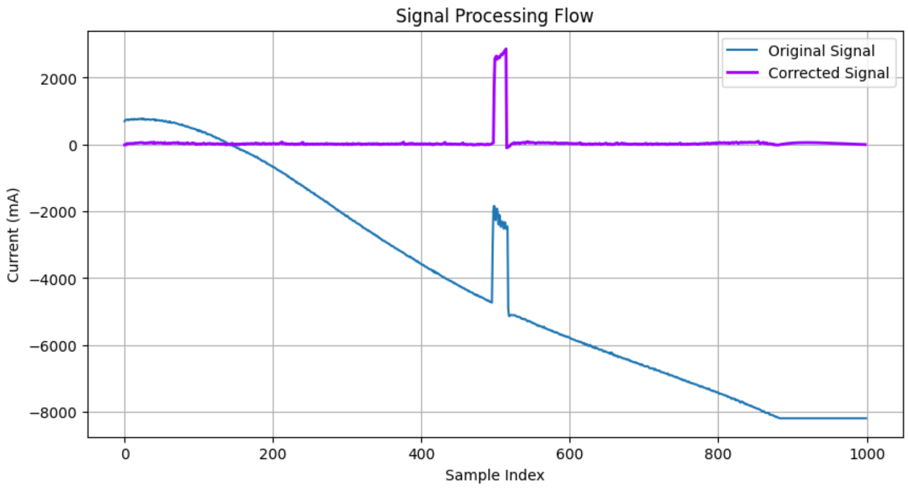

Figure 2: Test Signal

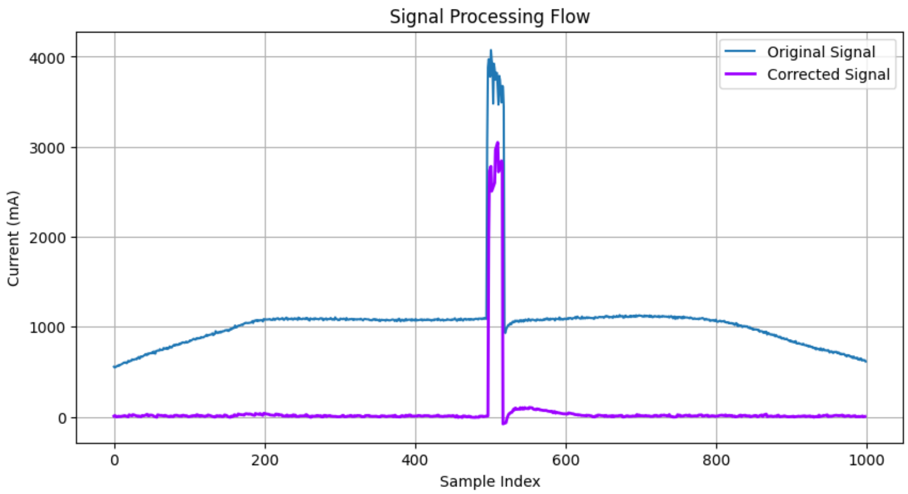

Figure 3: Toroid L:D4TOR

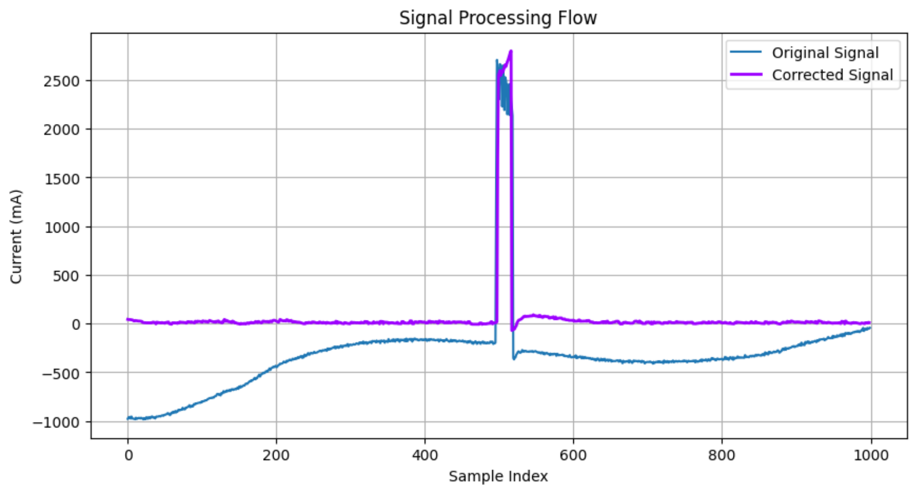

Figure 4: Toroid L:D5TOR

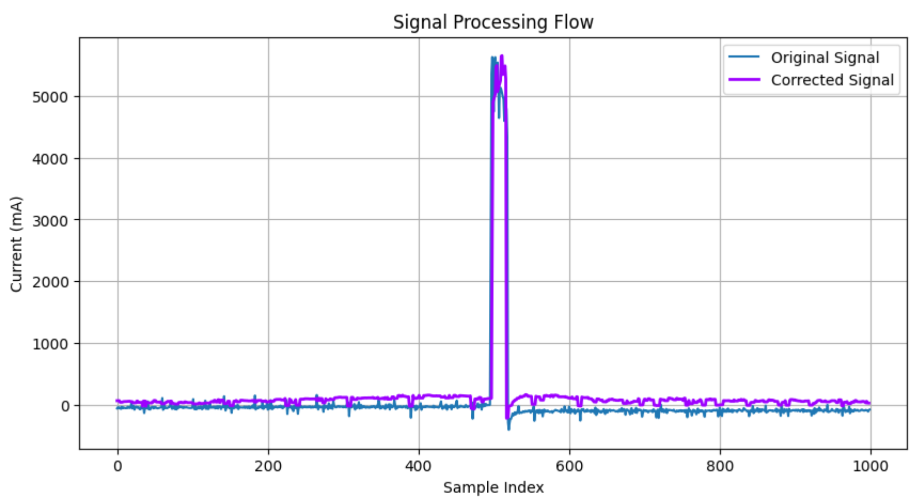

Figure 5: Toroid L:D7TOR

4.1 Baseline Correction Performance

Baseline correction was evaluated using old and current raw data from the FPGA. Visual inspection of the corrected signals confirmed that the algorithm effectively maintained pulse integrity while eliminating low-frequency drift.

Additionally, the flexibility of the ALS implementation allowed adaptation to signals with varying baseline characteristics, demonstrating the system's robustness.

4.2 Droop Correction Efficacy

Using simulated droop-distorted pulses, the droop correction algorithm successfully restored peak amplitudes. The method effectively compensated for a range of τ values, with minimal impact on high-frequency noise when parameters were properly tuned.

4.3 Data Throughput and Redis Performance

Benchmarks on a local network setup demonstrated that Redis could handle message rates exceeding 500 messages per second with minimal latency. The system maintained stable operation under continuous data flow conditions, affirming its suitability for real-time applications.

The modular communication setup allowed seamless integration of additional modules without significant performance degradation. However, Redis's single-threaded nature imposed some limitations on scalability, suggesting that future improvements might involve Redis clustering or alternative messaging frameworks for higher throughput applications.

The flexibility of the design also means that future extensions, such as additional filtering stages or more complex analysis algorithms, can be implemented with minimal disruption to the existing system.

4.4 Algorithm Sensitivity and Adaptation

The ALS algorithm's sensitivity to smoothing parameters highlights the need for adaptive tuning based on signal characteristics. Incorporating real-time analysis to adjust parameters dynamically could enhance robustness.

Similarly, droop correction requires careful balancing to avoid introducing noise artifacts. Further research into adaptive filter techniques may yield improved results.

An adaptive control module that monitors signal quality metrics and adjusts processing parameters accordingly could be a significant enhancement for future iterations of this system.

4.5 System Integration and Use Cases

The system's modularity and reliance on Redis for communication make it suitable for a variety of applications beyond beam diagnostics. Potential use cases include:

- Industrial sensor networks requiring real-time data correction and monitoring.

- Scientific experiments involving high-speed data acquisition and preprocessing.
- Embedded systems where lightweight, efficient communication between processing units is critical.

Deploying this system on actual beam diagnostic hardware remains a critical next step for validating its performance under operational conditions. Integration with control systems and data acquisition frameworks used in particle accelerators will provide valuable insights into practical deployment challenges and opportunities.

4.6 Future Work

Potential future directions include:

- Adaptive Parameter Tuning: Implementing feedback mechanisms for dynamic adjustment of correction parameters, such as p in ALS (in case the toroid output is inverted). AI/ML models can be used to dynamically deploy the future tuning optimizations of the parameters.
- High-Throughput Messaging Alternatives: Exploring ZeroMQ or gRPC for scenarios requiring higher data rates.
- Automated Signal Quality Monitoring: Developing algorithms to monitor signal quality metrics in real-time and trigger alerts or adjustments as needed.

These enhancements will not only improve system performance but also broaden its applicability to a wider range of real-time signal processing applications.

5 Conclusion

The Toroid Signal Processing Container with Redis Integration successfully met its objectives, providing a robust framework for real-time signal correction and modular processing. The combination of ALS baseline correction, numerical droop compensation, and Redis-based communication resulted in a system capable of enhancing signal fidelity and supporting distributed processing architectures.

This project replaces the data processing formerly done by the FPGA, and by doing it on the server, it can use more powerful correction mechanisms. Additionally, it is more user-friendly to configure which Redis key is being used, which callback function to use, and to add new callback functions to the program.

The insights gained during this project, particularly in balancing algorithm sensitivity with real-time processing requirements and managing inter-process communication, will inform future work in both academic research and industrial applications.

6 Acknowledgements

I'd like to thank my mentors Michelle Ibrahim and Jose Berlioz Rivera for giving me the opportunity to work on this project, and for guiding me through the process. I've enjoyed my time working with the Accelerator Division's Instrumentation Department so far, and I'd be thrilled to return in the future.

- This manuscript has been authored by Fermi Research Alliance, LLC under Contract No. DE-AC02-07CH11359 with the U.S. Department of Energy, Office of Science, Office of High Energy Physics.

- This work was supported in part by the U.S. Department of Energy, Office of Science, Office of Workforce Development for Teachers and Scientists (WDTS) under the Community College Internship (CCI)

7 References

1. SciPy Library Documentation, https://docs.scipy.org/doc/scipy/.

2. P. H. C. Eilers, "A perfect smoother," Anal. Chem., vol. 75, no. 14, pp. 3631–3636, 2003.

3. D. Salomon, Data Compression: The Complete Reference, Springer, 4th ed., 2007.

4. Redis Documentation, https://redis.io/docs/.