Design and Implementation of a LabVIEW based Computer Control for EPR Instrumentation

Diploma Thesis Defense

Matthias Kolja Miehl

October 22, 2010 at 2:30 pm

Milwaukee, Wisconsin, USA
Medical College of Wisconsin
MACC Fund Research Center
Room L3075

FACH HOCHSCHULE LÜBECK
University of Applied Sciences

MSOE

MEDICAL COLLEGE OF WISCONSIN

- Control and acquisition framework
- Easily change hardware
- All experiments in one application

- Control and acquisition framework
- Easily change hardware
- All experiments in one application

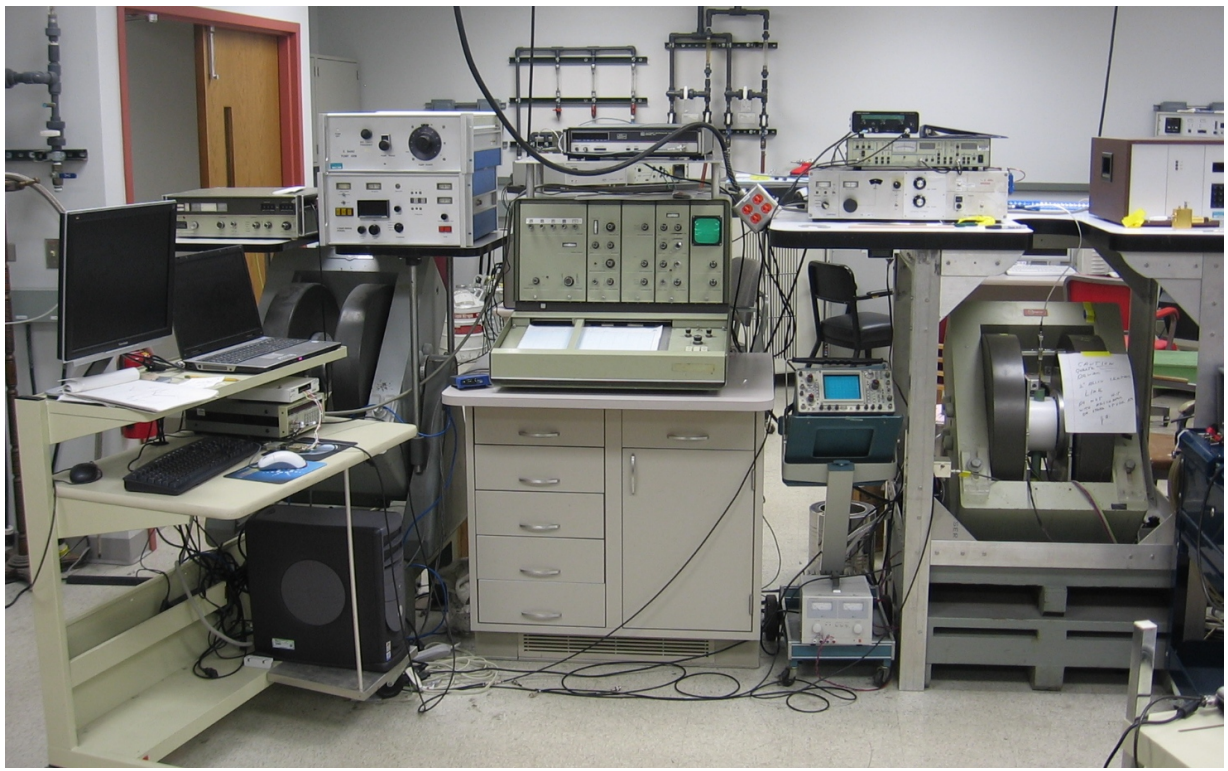


Figure: L-Band spectrometer at MCW's Biophysics Research Department

Contents

Contents

Need

MCW Research:

- New experiments
- Different spectrometers
- Changing Hardware

Need

MCW Research:

- New experiments
- Different spectrometers
- Changing Hardware

Situation

Use of separate programs for

- each experiment,
- each spectrometer, and
- were rewritten for new instruments.

Need

MCW Research:

- New experiments
- Different spectrometers
- Changing Hardware

Situation

Use of separate programs for

- each experiment,
- each spectrometer, and
- were rewritten for new instruments.

Problem

- Labor and time intensive tasks
- Reduction of actual research time

Motivation

Increase the actual research time by reducing redundancy.

- Experiment implementation
- Changing hardware

Motivation

Increase the actual research time by reducing redundancy.

- Experiment implementation
- Changing hardware

Goal

Versatile framework:

- All experiments
- Every spectrometer

Existing Approaches

- WinEPR
- EWWin
- SpecMan4EPR

Existing Approaches

- WinEPR
- EWWin
- SpecMan4EPR

MCW's Requirements

- Exchange of instruments
- Storing experiment settings
- Easy extension by customer

Contents

Key Requirements

- All experiments in one application
- Instruments easily exchangeable
- Maintainability and extendability
- 32 and 64-bit driver support

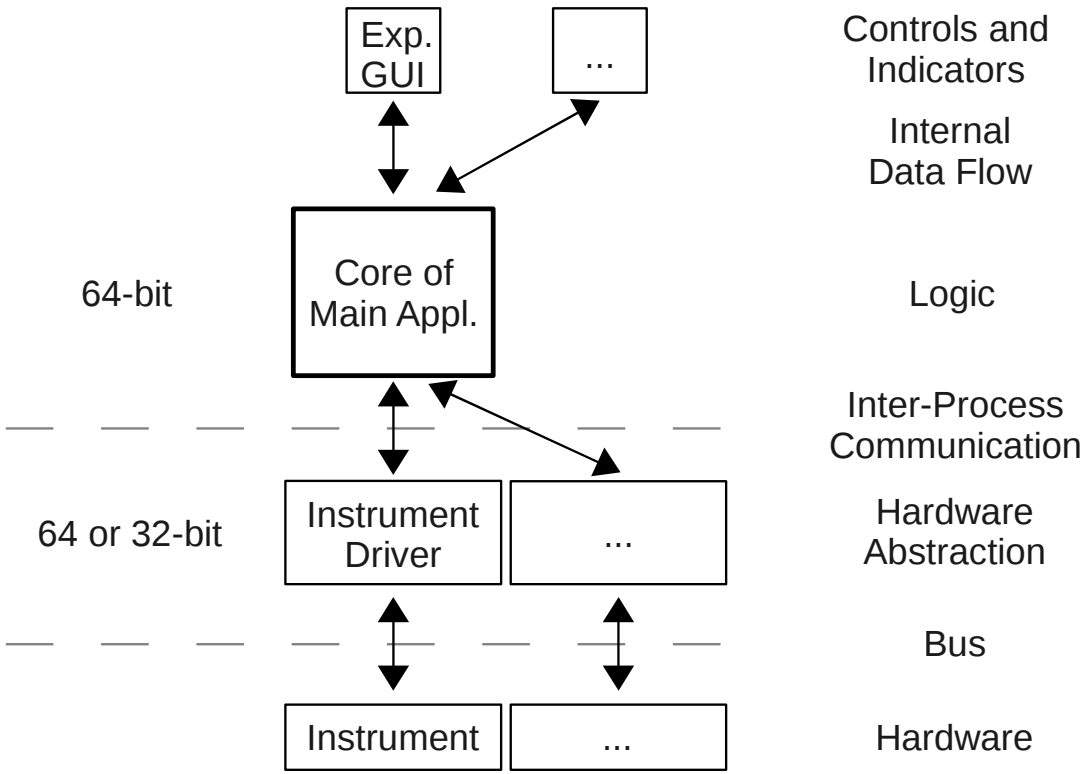


Figure: Overall application structure (concept)

Contents

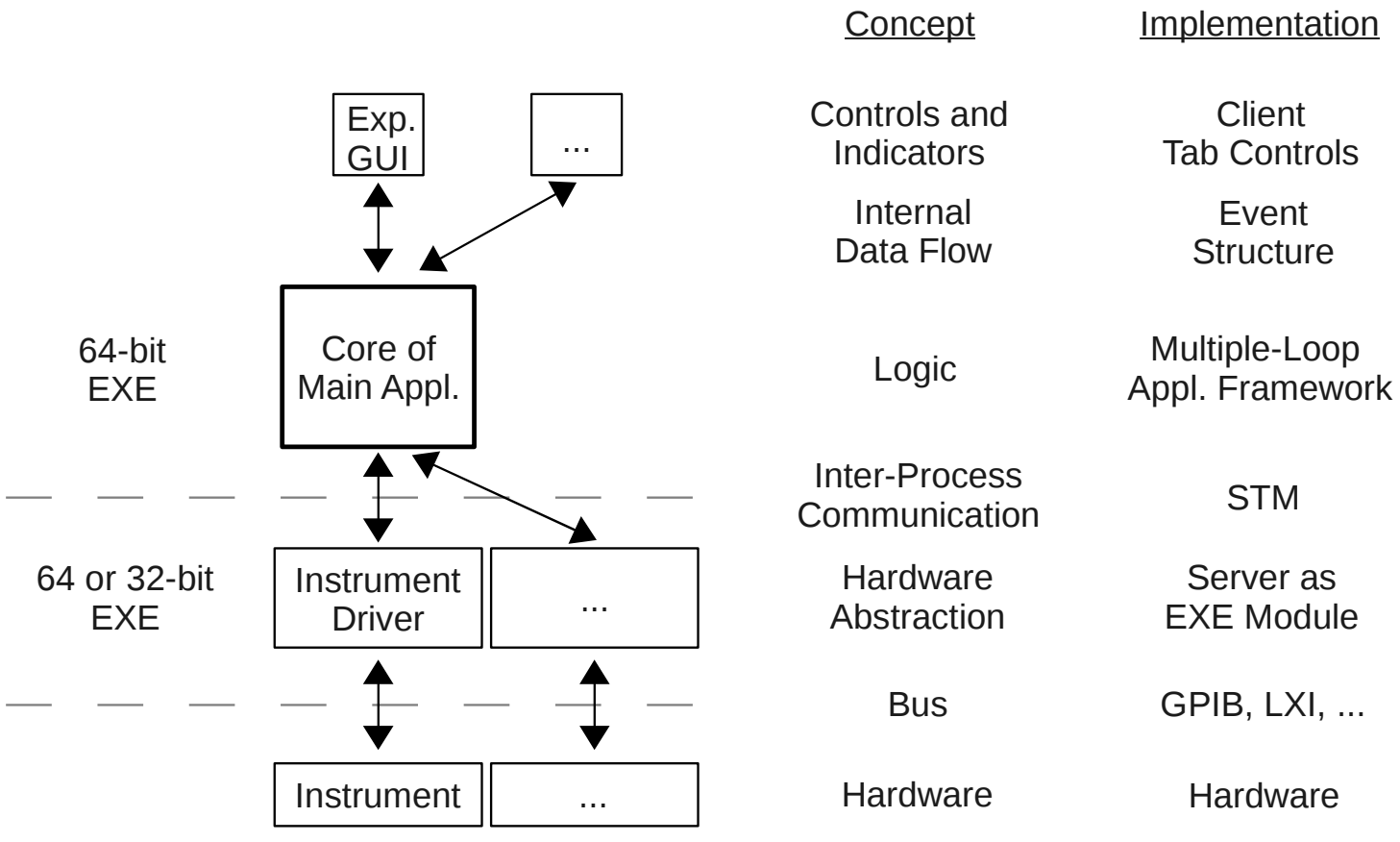


Figure: Overall application structure (implementation)

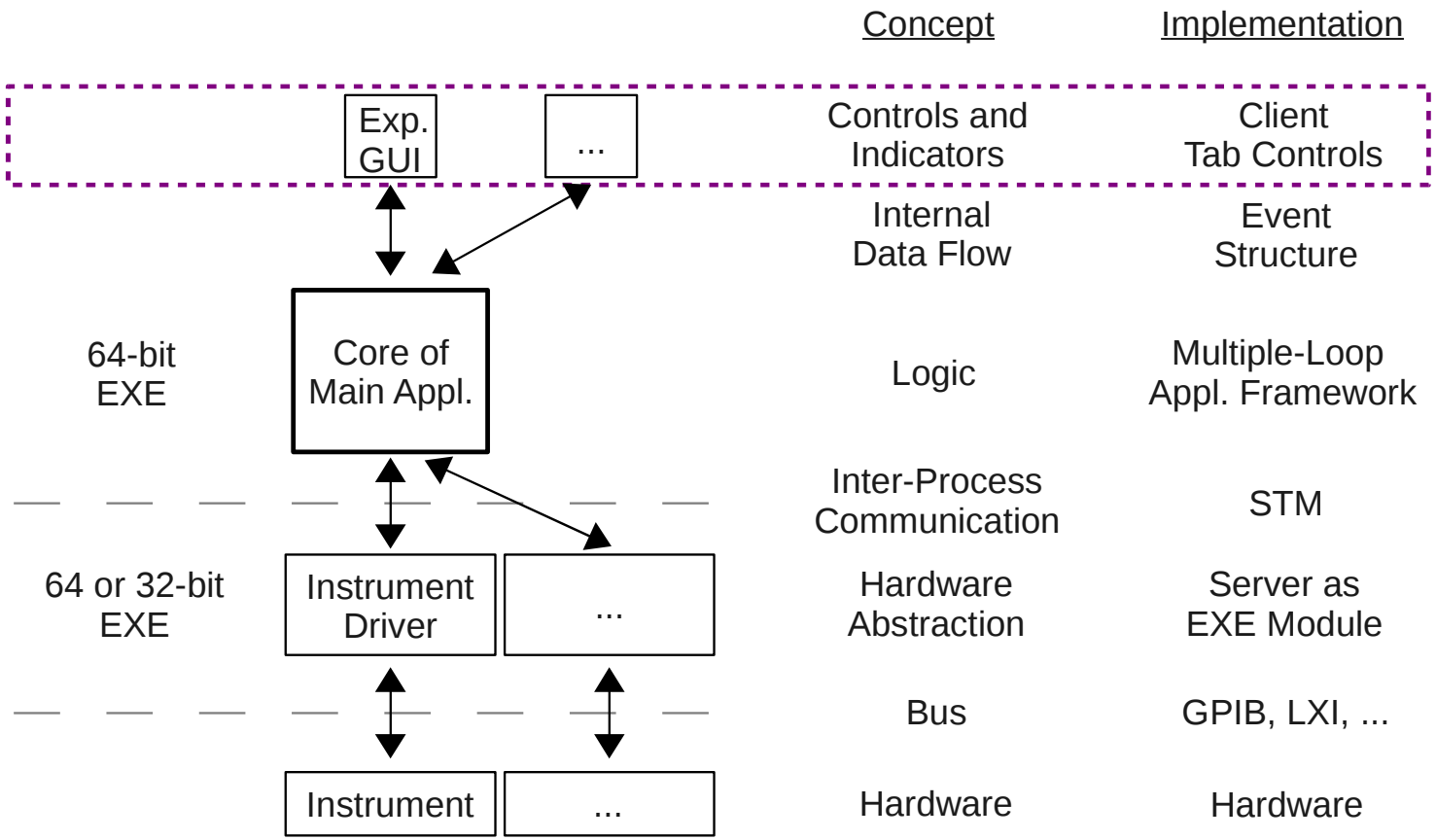


Figure: Overall application structure (implementation)

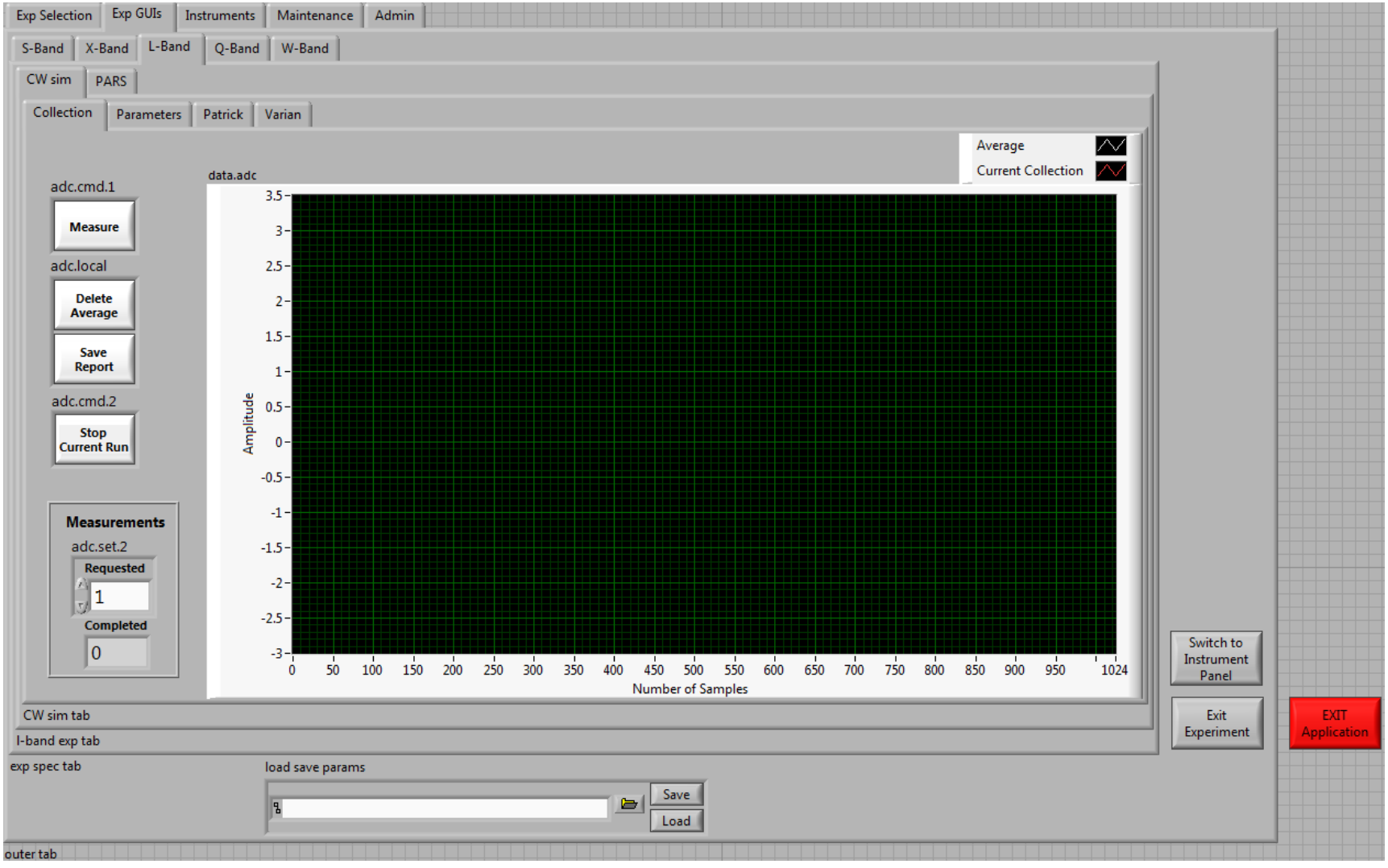


Figure: Tab control structure of the application

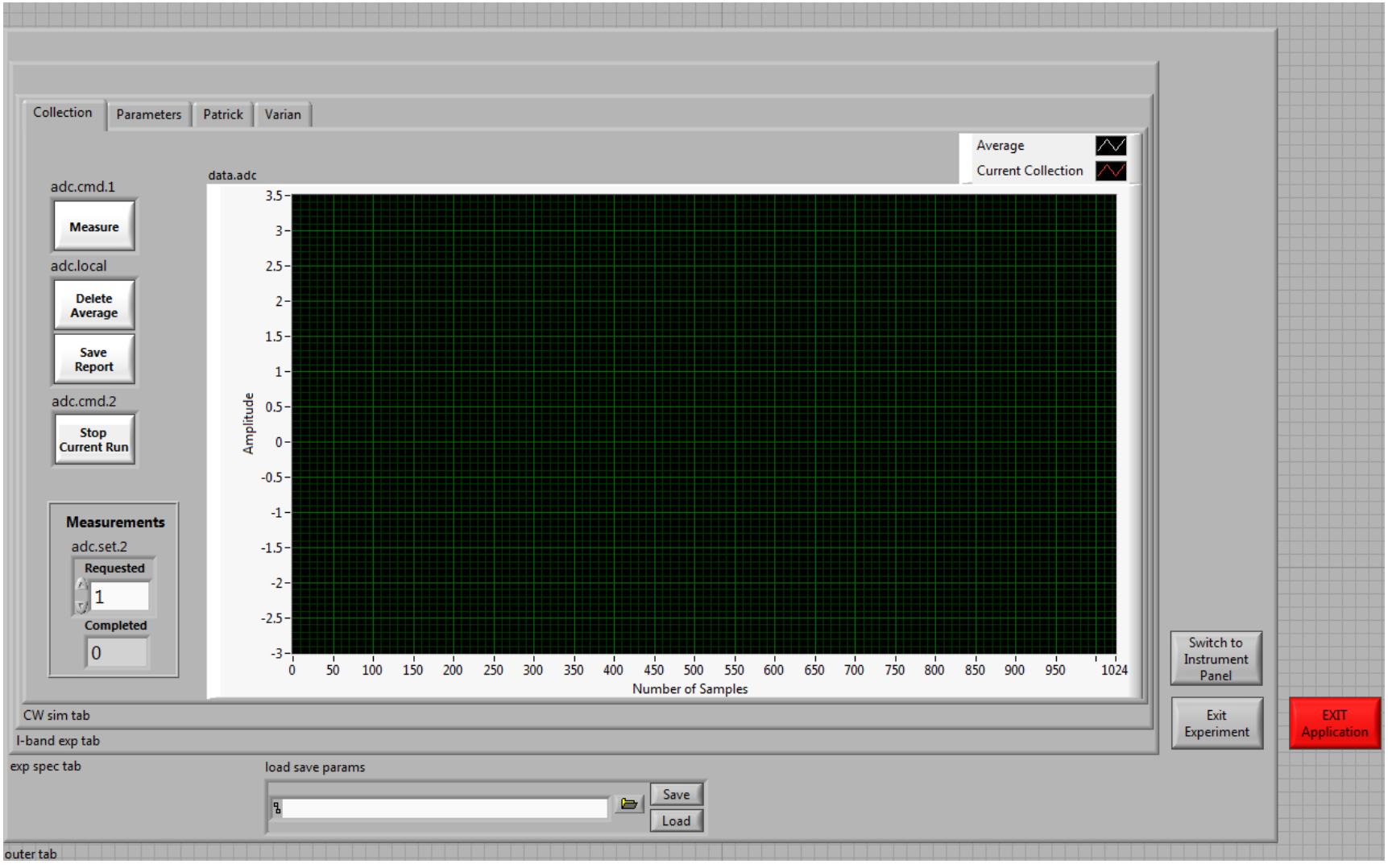


Figure: Tab control structure as the operator sees it

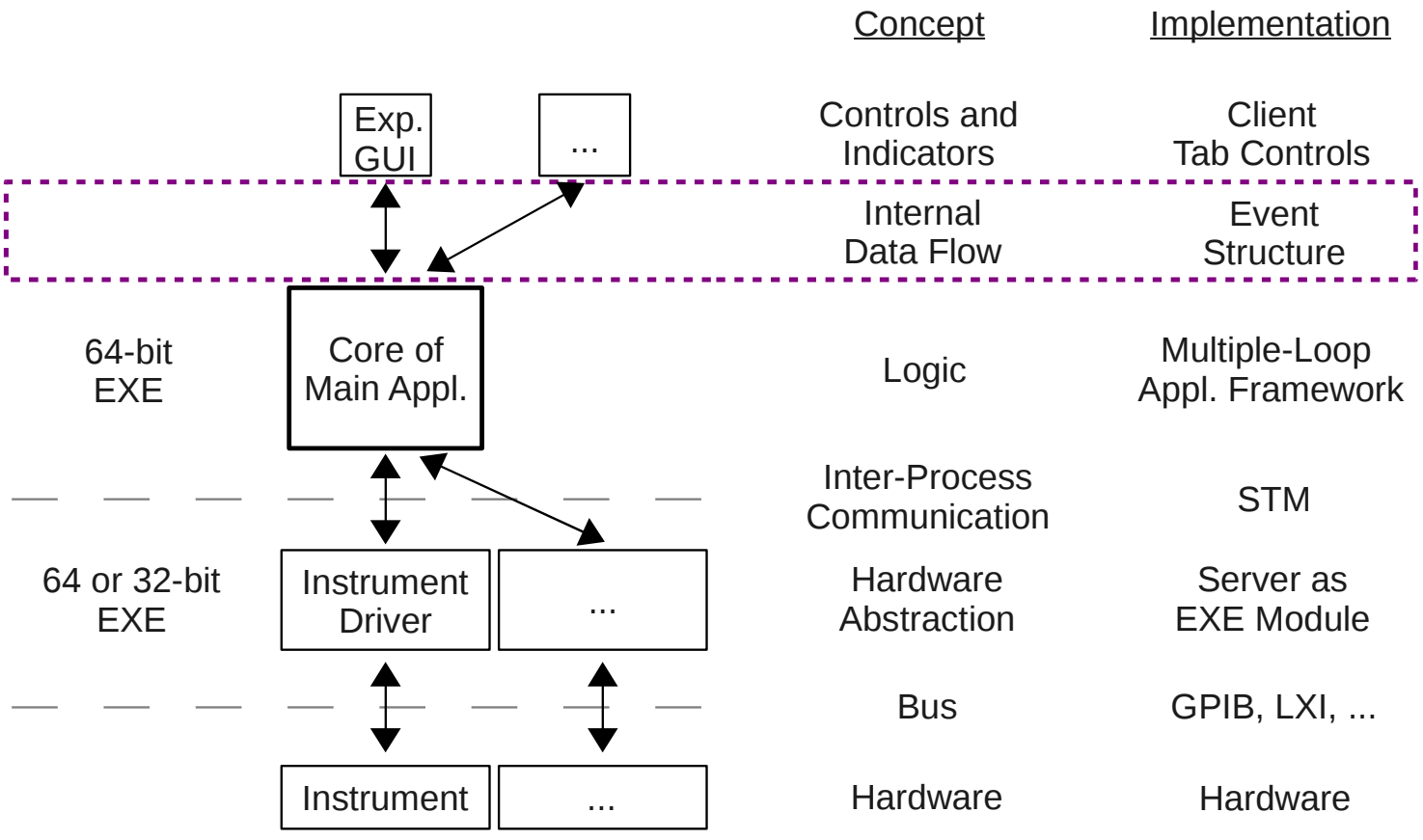


Figure: Overall application structure (implementation)

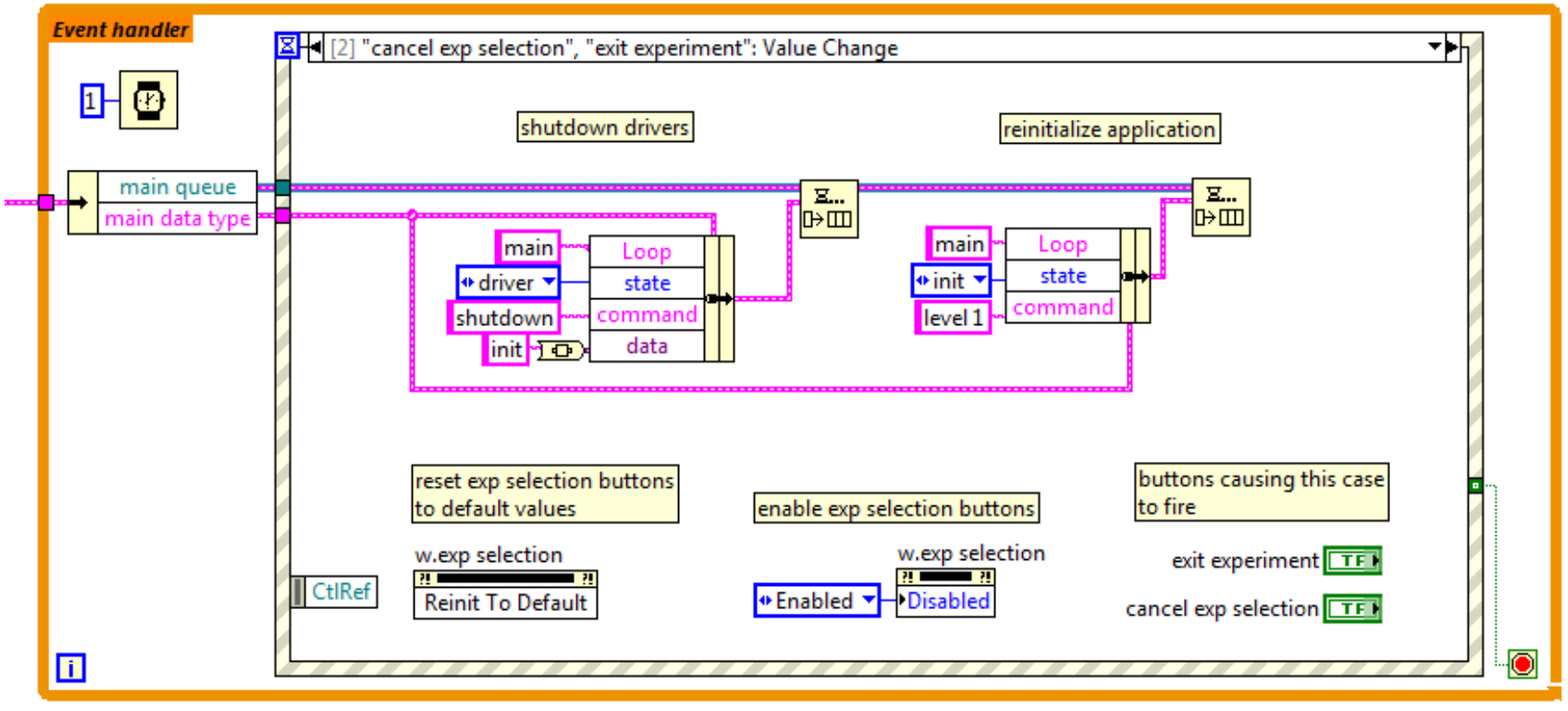


Figure: Main program event handler loop

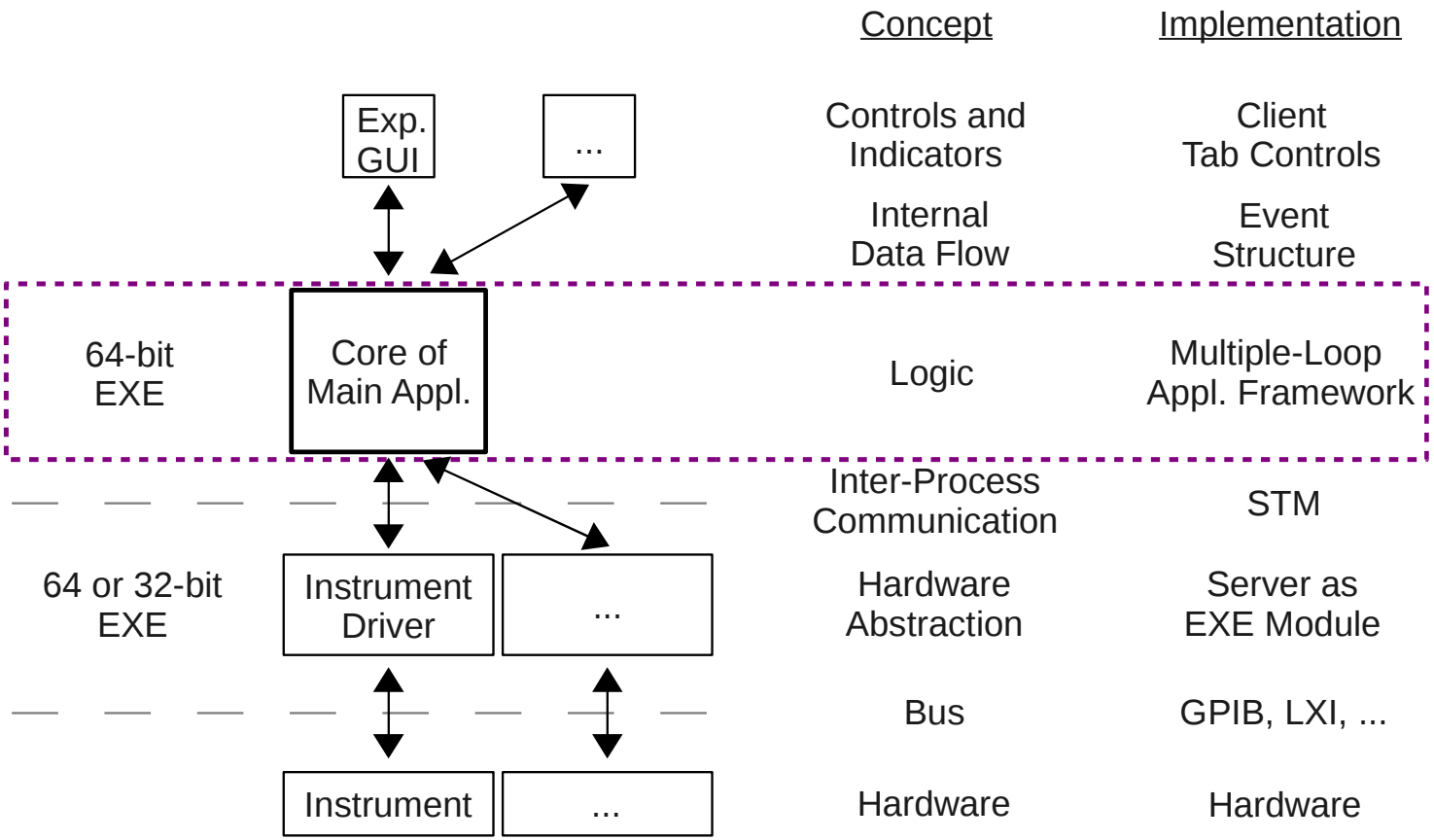


Figure: Overall application structure (implementation)

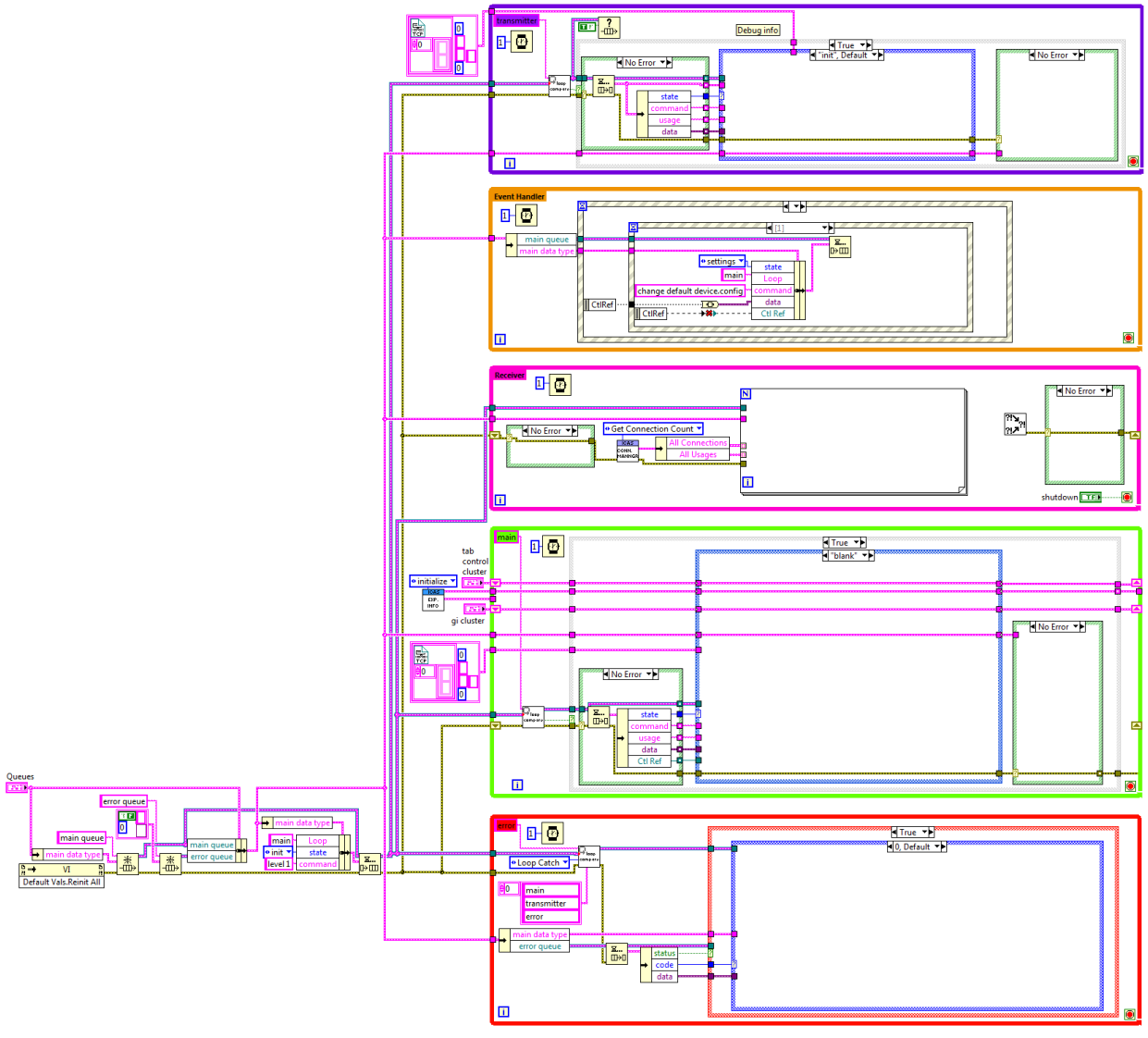


Figure: Implementation of the main program's parallel loop structure

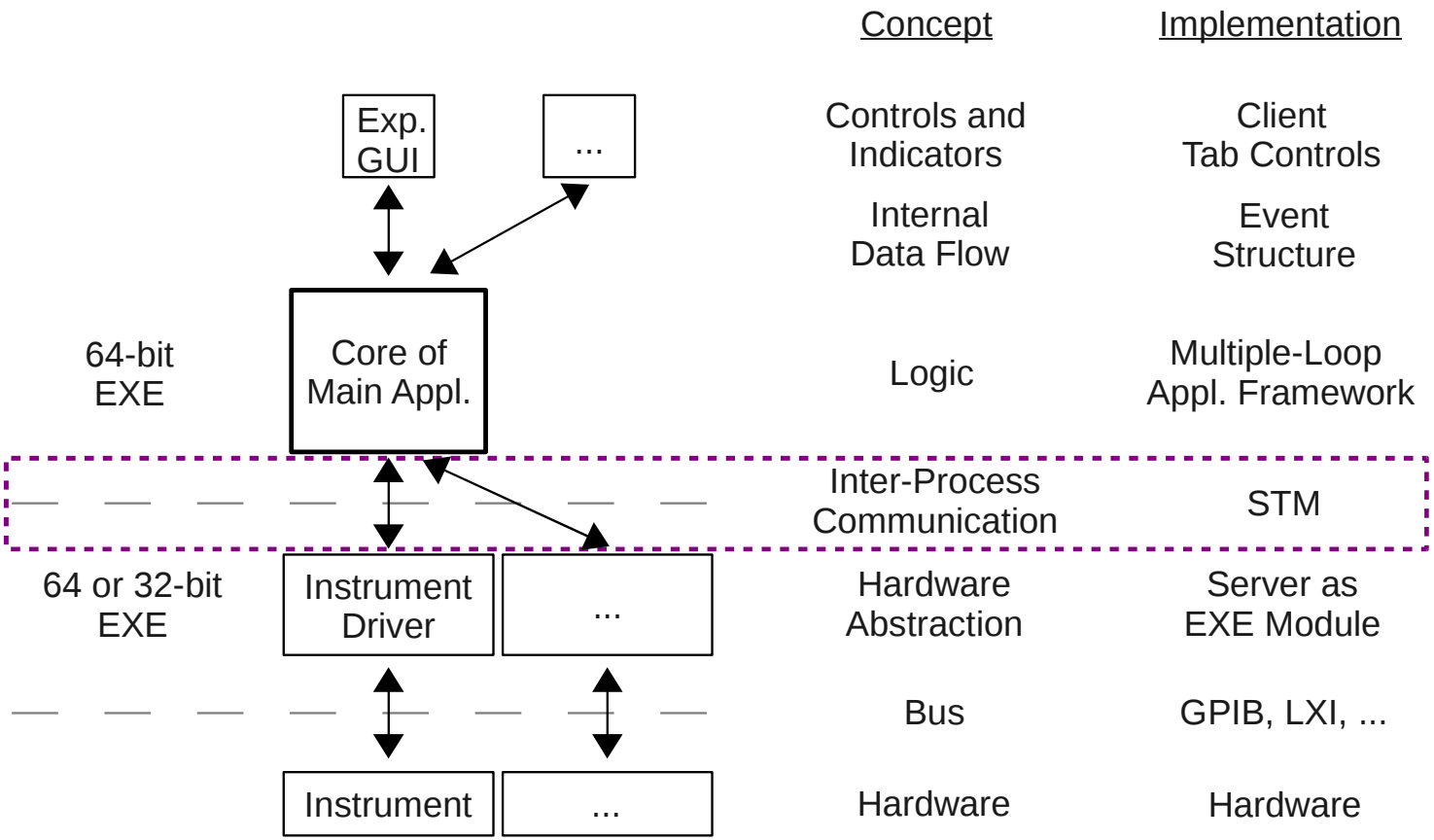


Figure: Overall application structure (implementation)

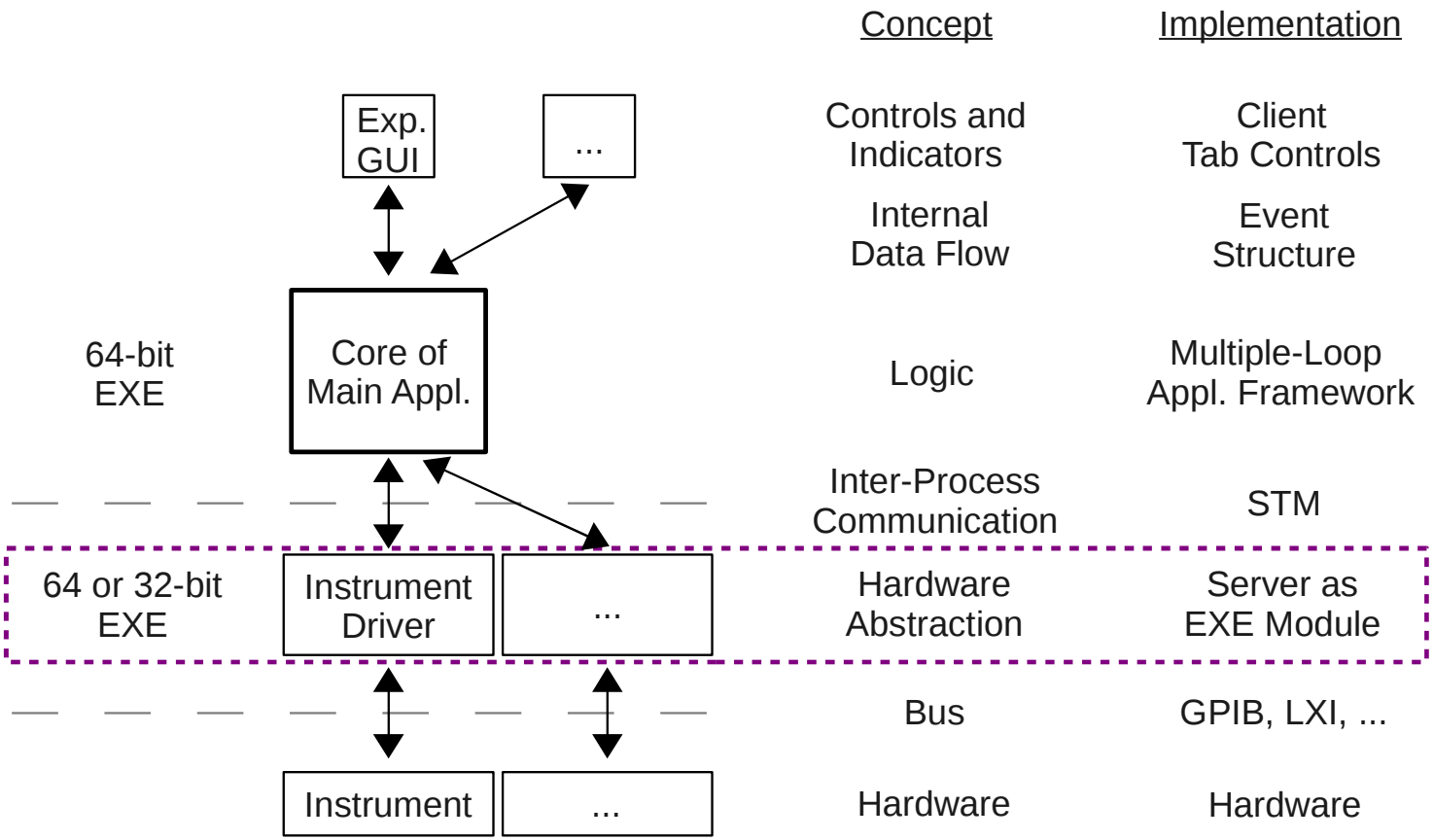


Figure: Overall application structure (implementation)

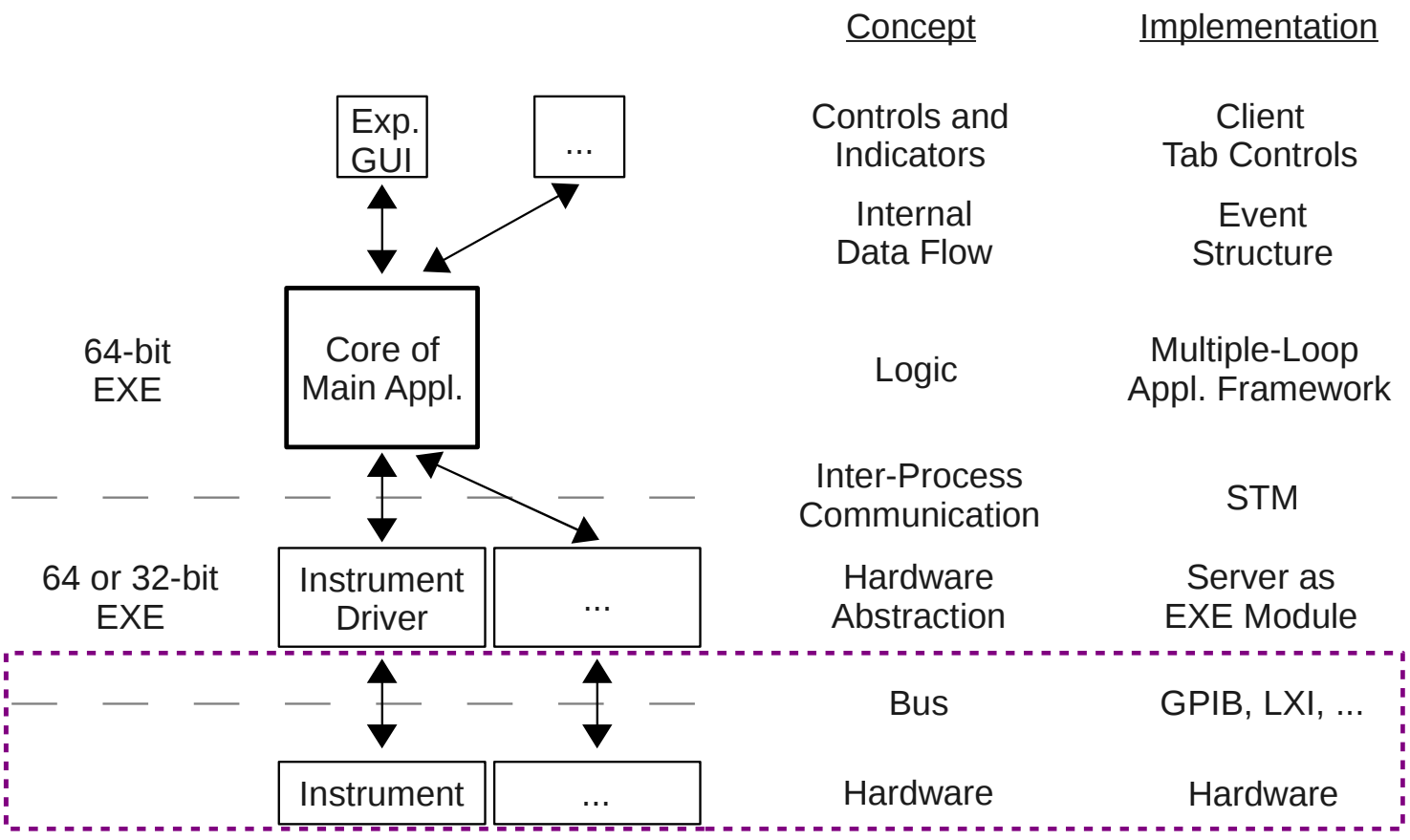
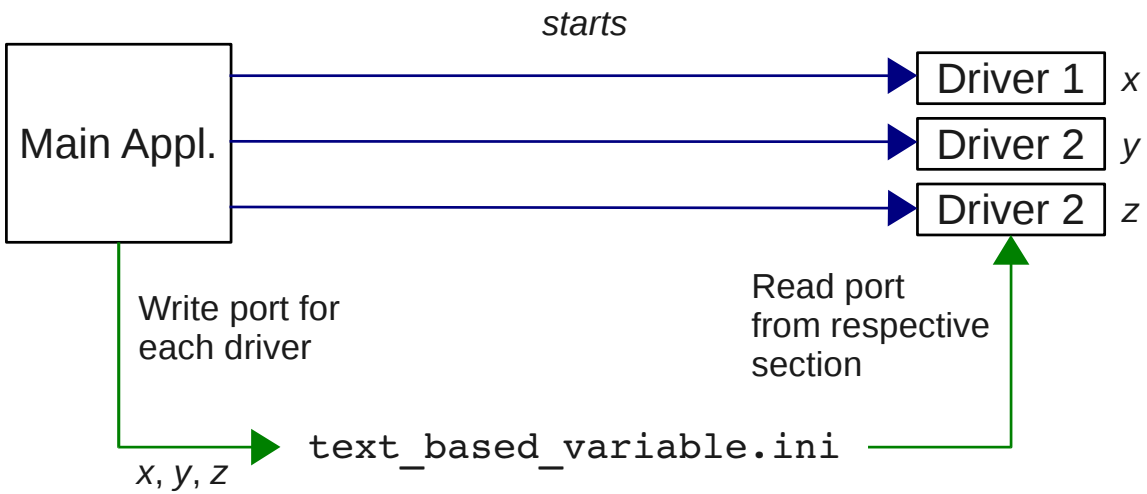
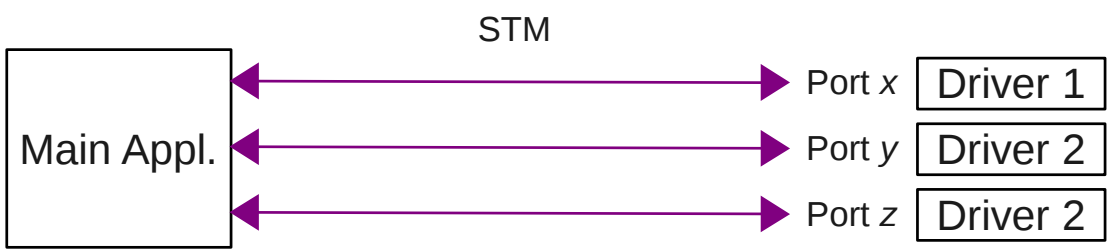


Figure: Overall application structure (implementation)

Startup of Drivers



Communication



Advantages:

- Command-based

Advantages:

- Command-based
- Hides transport layer details

Advantages:

- Command-based
- Hides transport layer details
- Minimizes network traffic

Advantages:

- Command-based
- Hides transport layer details
- Minimizes network traffic
- Small overhead

Advantages:

- Command-based
- Hides transport layer details
- Minimizes network traffic
- Small overhead
- High throughput

Variant Data

Variant Data

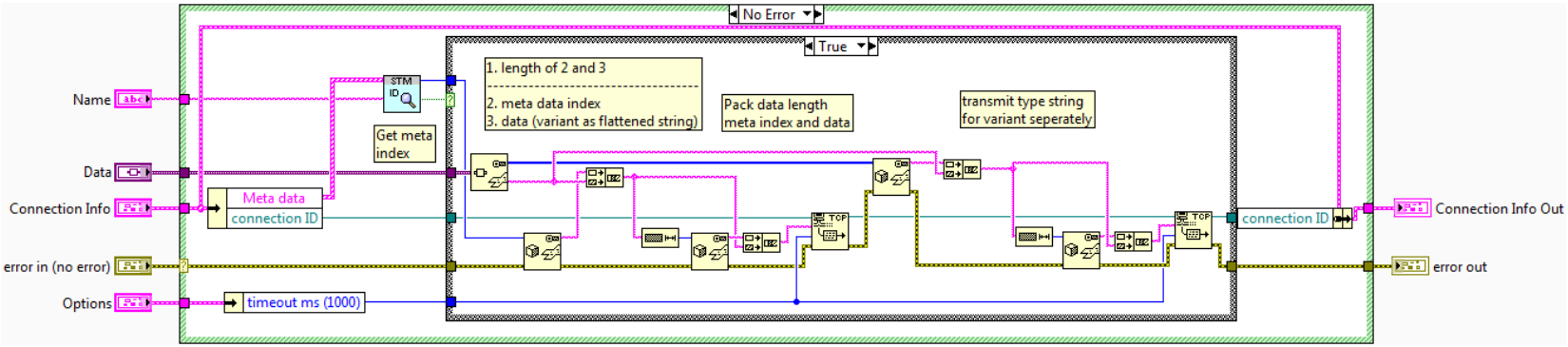


Figure: Code of modified STM *write* function block

Variant Data

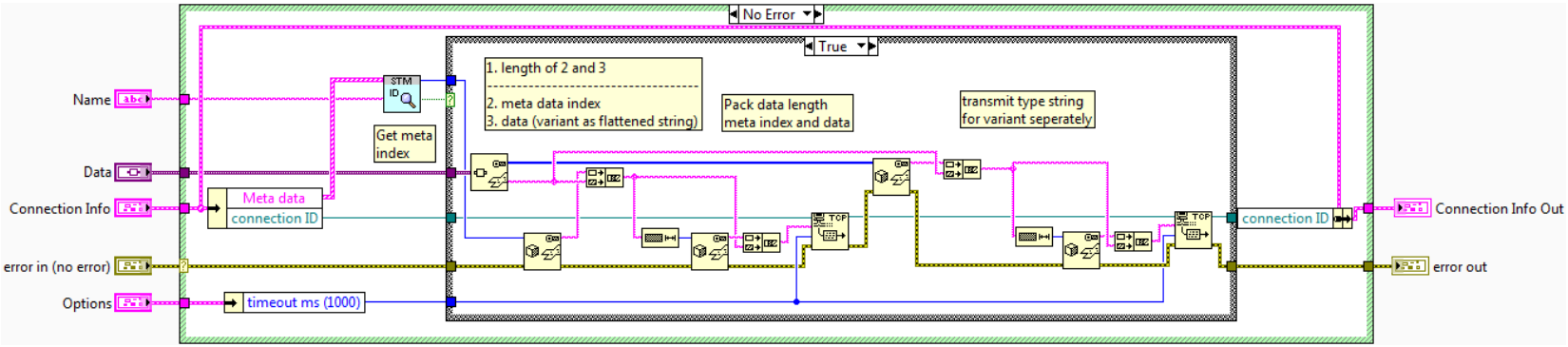


Figure: Code of modified STM *write* function block

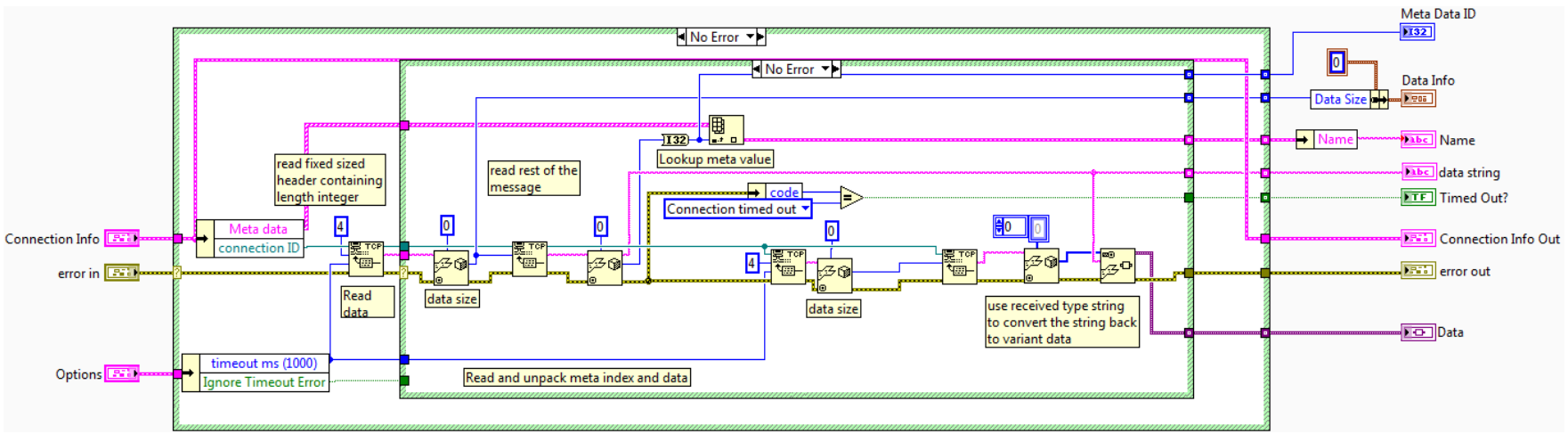


Figure: Code of modified STM *read* function block

Variant Data

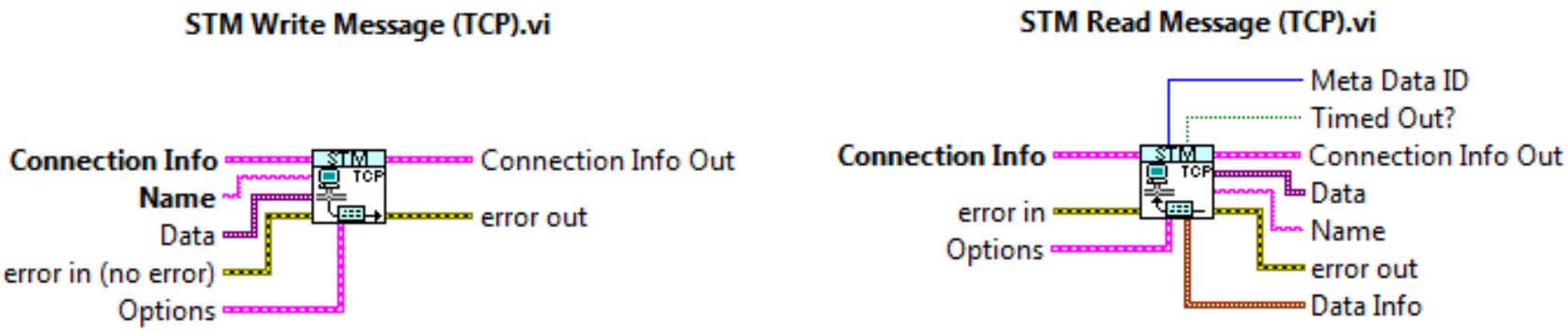


Figure: Modified STM function blocks for writing and reading

Command-Based Communication Framework

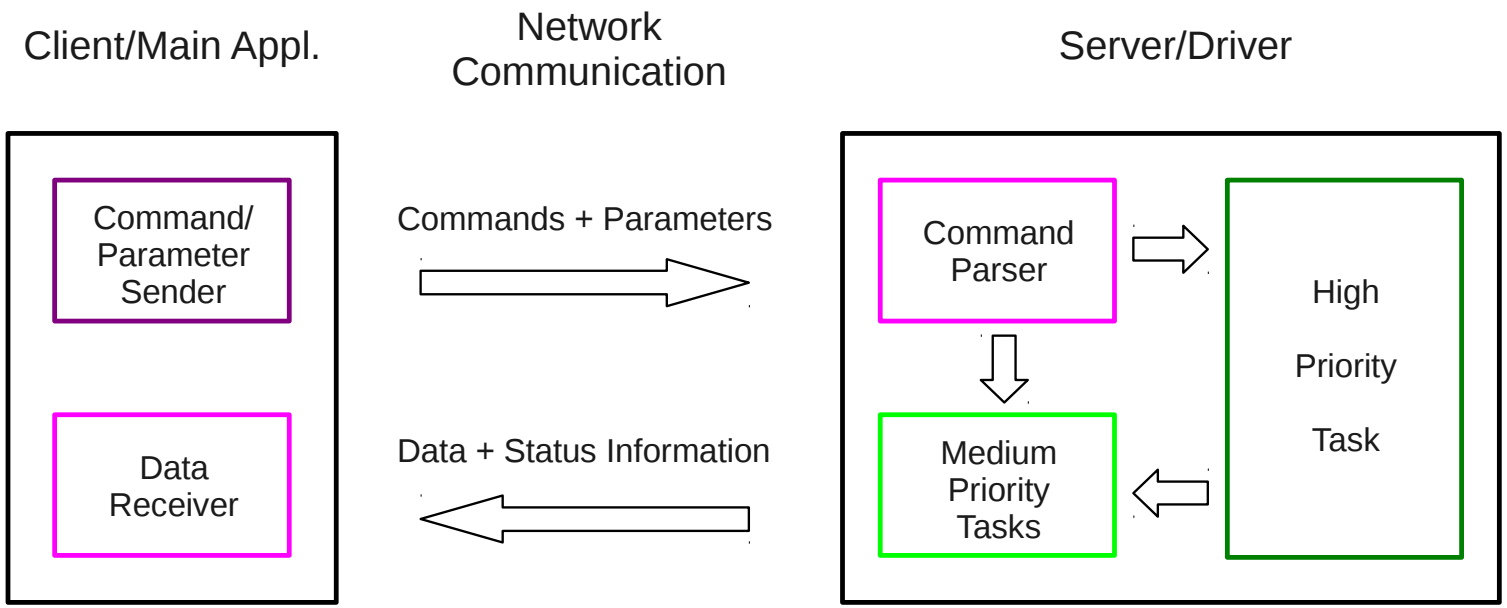


Figure: STM's command-based communication framework

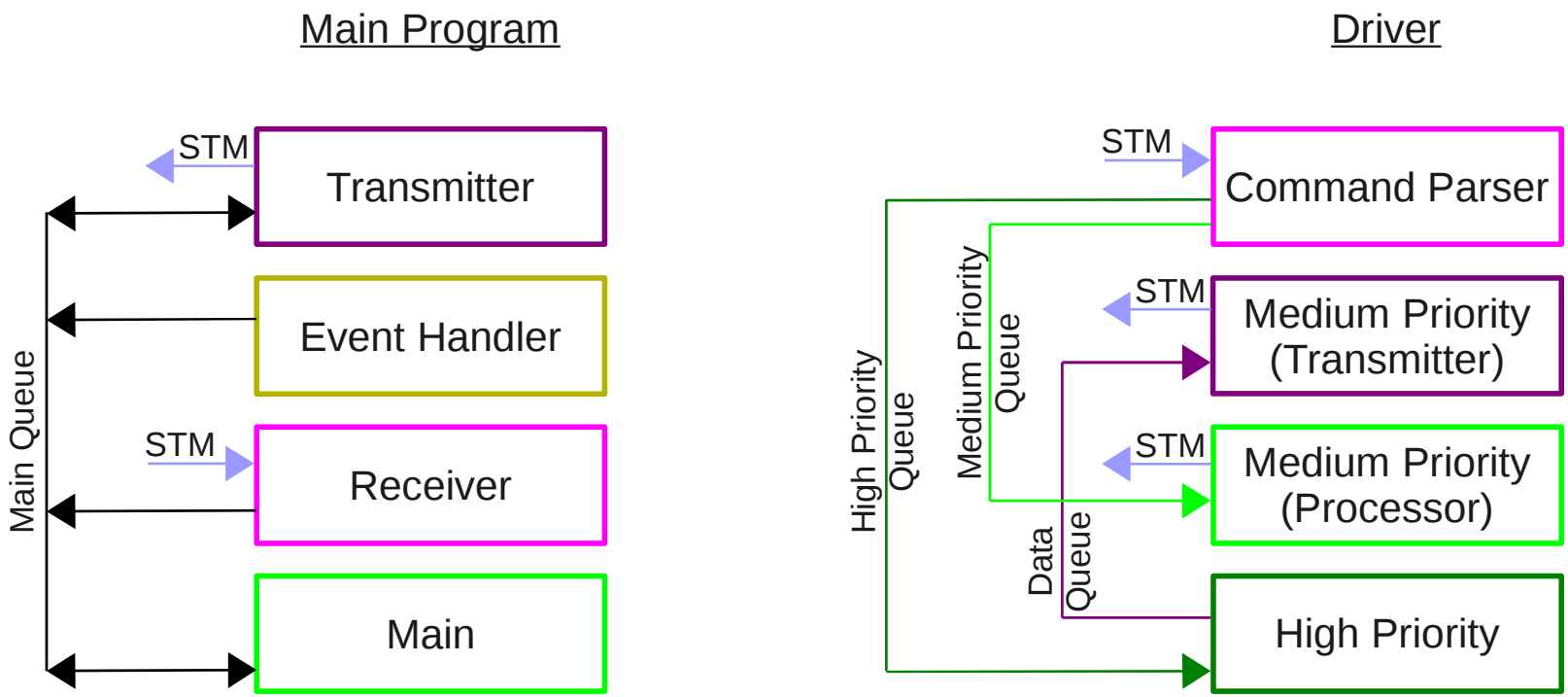


Figure: Parallel loop structure of main program and drivers

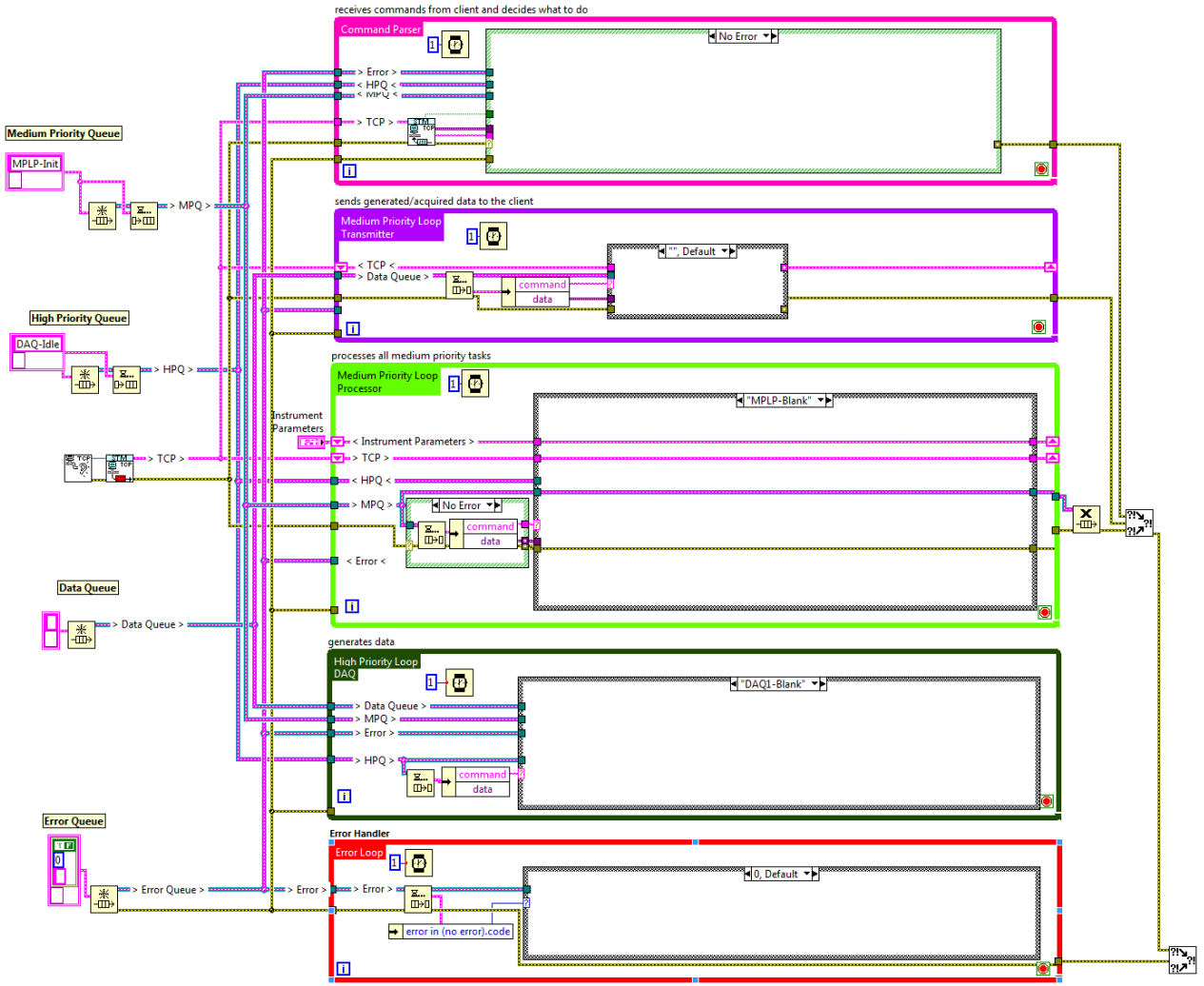


Figure: Actual loop structure of driver

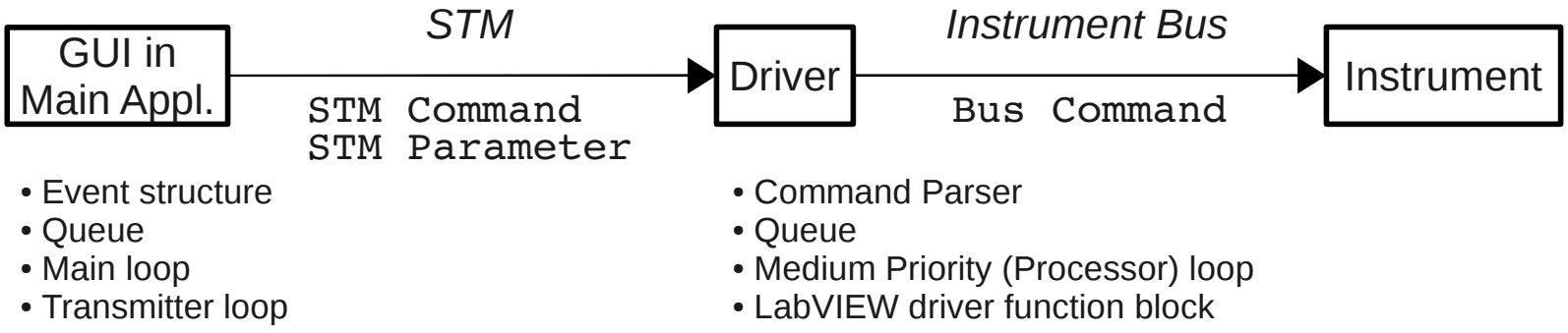


Figure: Resulting information flow on *instrument control* change

Goal: Relation between drivers and instruments

Goal: Relation between drivers and instruments

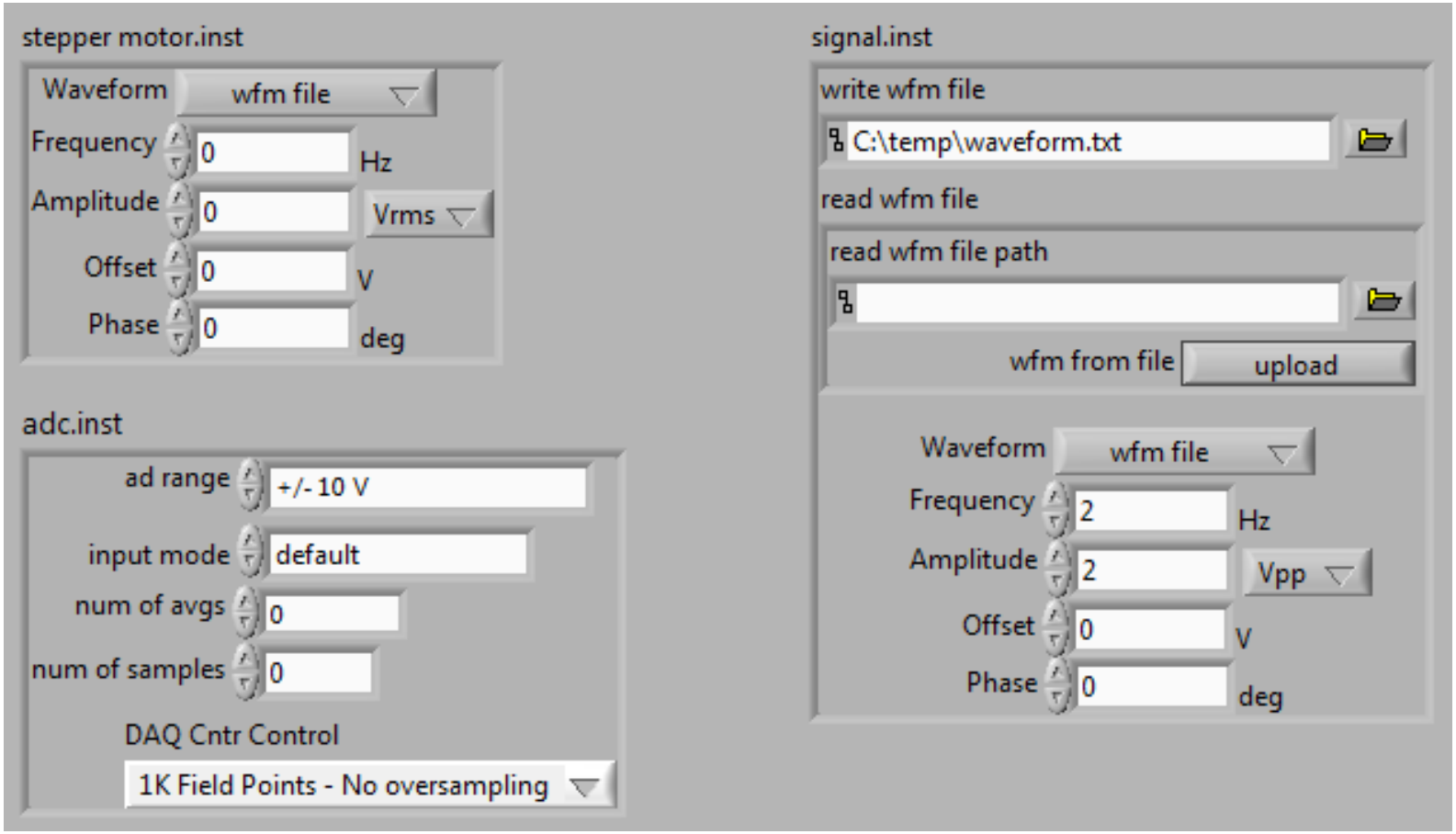


Figure: Instrument control clusters on instrument panel

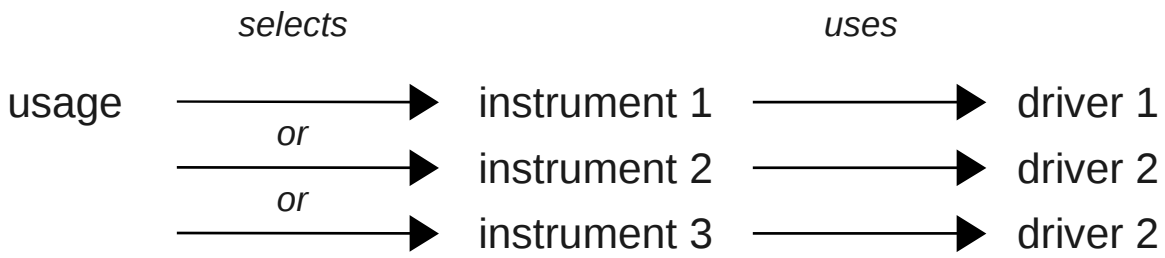


Figure: Usage concept

L-Band.CW sim.ini

```
; assignment of instrument to usage
; -------------------------------
; usage = instrument name
[config]
stepper motor = "DS345(1)"
signal = "DS345(2)"
adc = "NI PCI-6024E"
```

L-Band.CW sim.ini

```
; EXE names
; ---------
; instrument name = driver name
[driver]
DS345(1) = "srds345"
DS345(2) = "srds345"
N8241A = "n8241a"
N8241A Option 330 = "n8241a"
NI PCI-6024E = "ni pci-6024e"
```

L-Band.CW sim.ini

```
; instrument resource names
; ------------------------
; instrument name = resource name
[resource]
DS345(1) = "17"
DS345(2) = "18"
N8241A = "TCPIP0::169.254.1.20::inst0::instr"
N8241A Option 330 = "TCPIP0::169.254.1.22::inst0::instr"
NI PCI-6024E = "Dev1"
```

L-Band.CW sim.ini

```
; default parameters
; ------------------
; parameter_DataType = value

[signal]
frequency_Digital = 1.000000
amplitude_Digital = 3.000000
phase_Digital = 0.000000
offset_Digital = 0.000000
amplitude unit_Ring = 1.000000
waveform_Ring = 1.000000
write wfm file_Path = "/C/temp/waveform.txt"

[...]
```

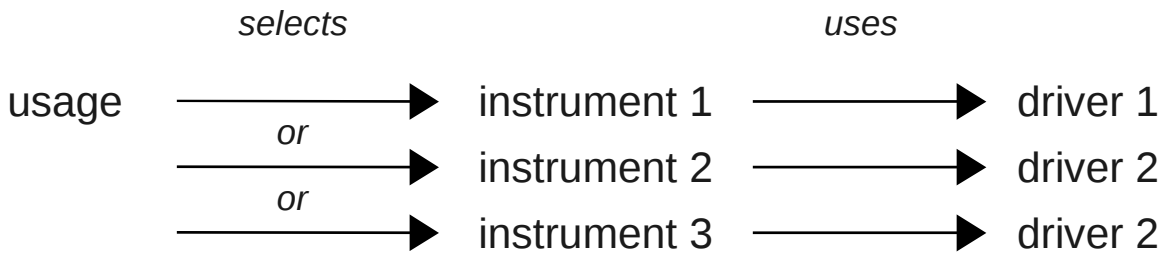


Figure: Usage concept

Figure: Usage implementation

Goal: Select different instruments for a usage

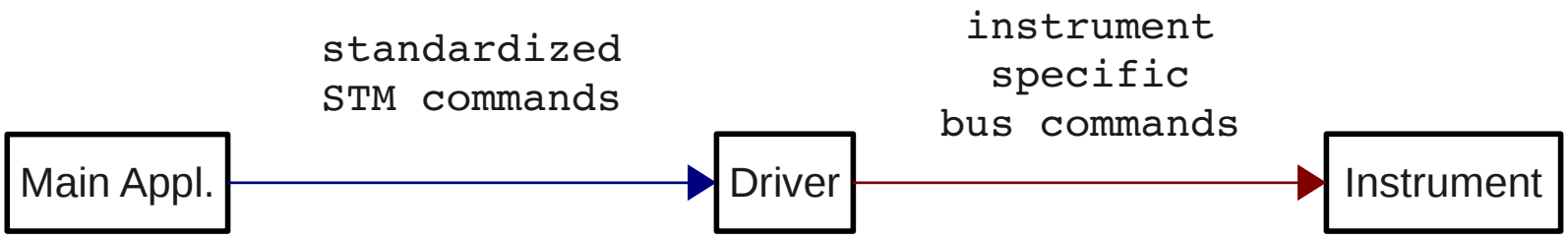


Figure: Driver as hardware abstraction layer

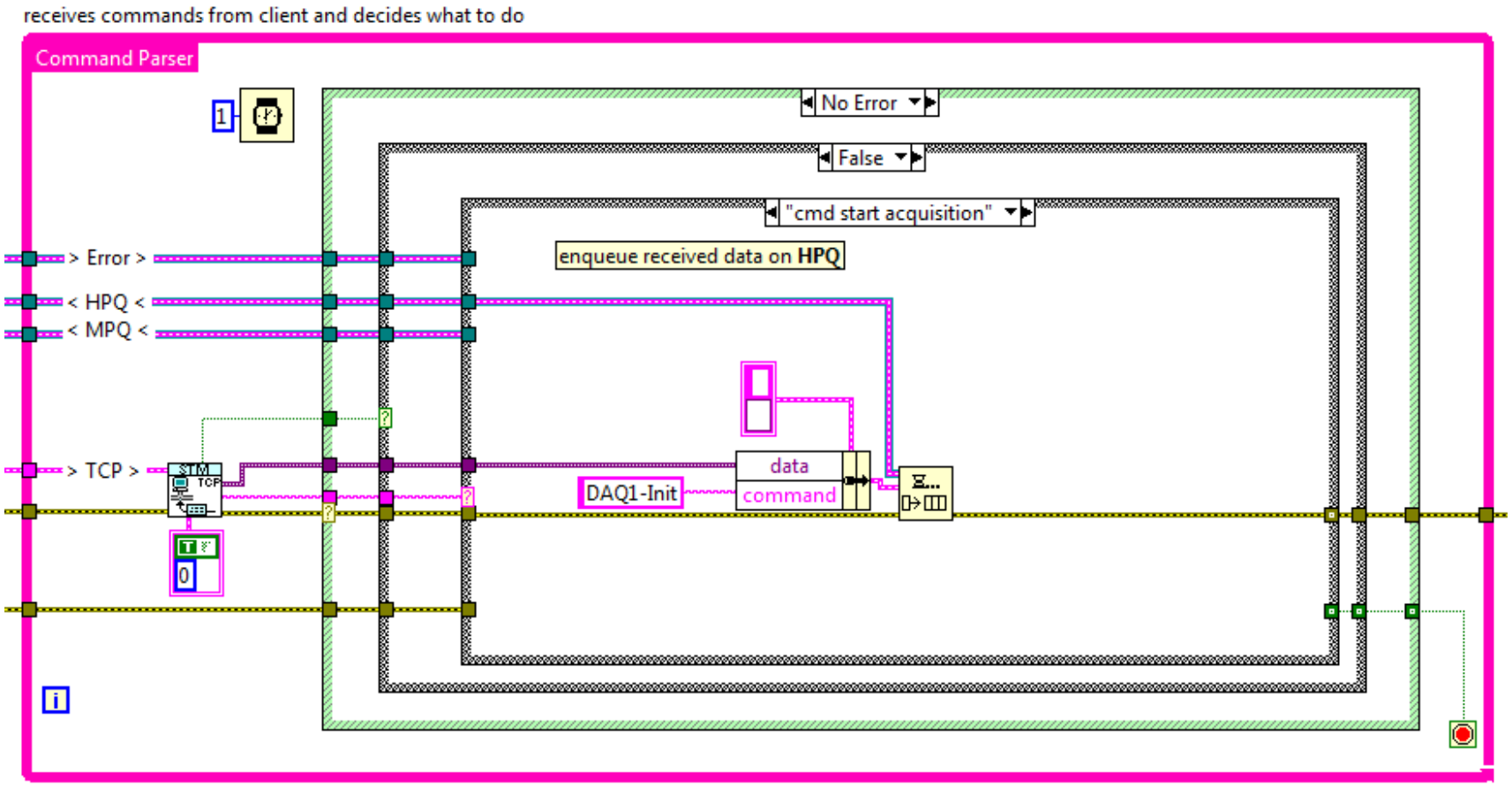


Figure: Driver's *command parser* loop

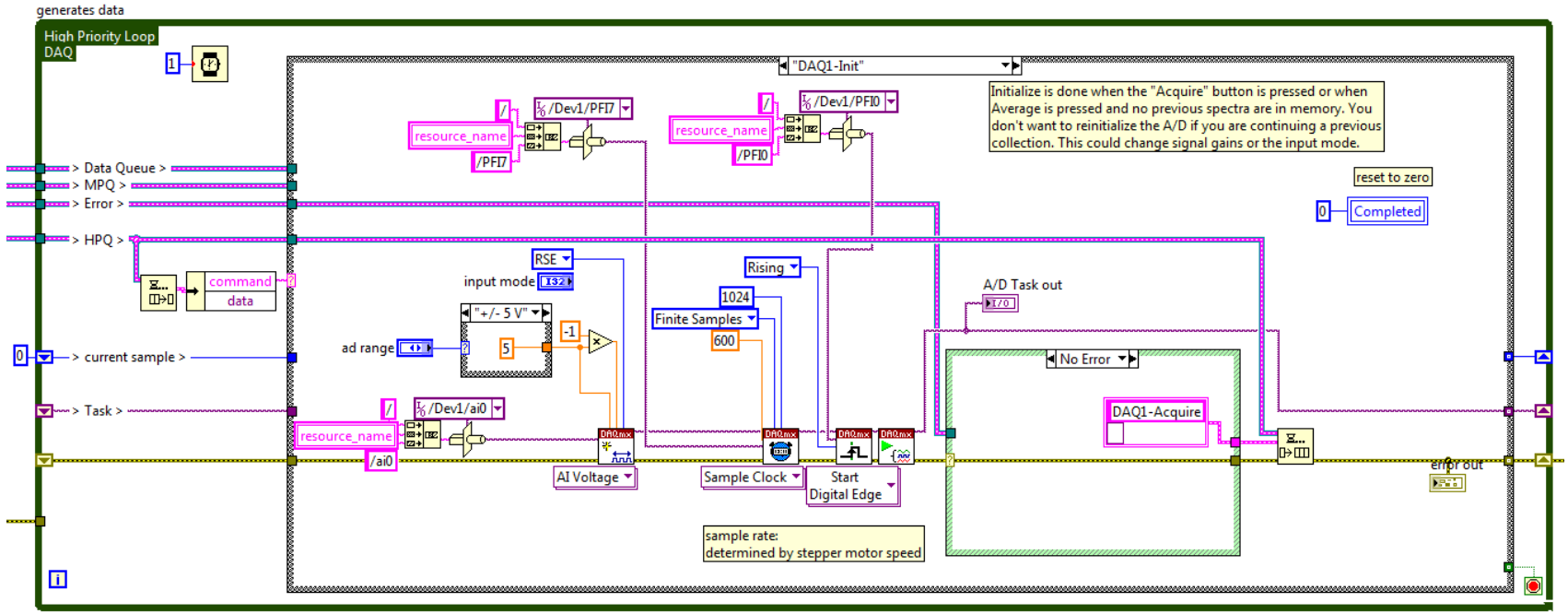


Figure: Driver's *high priority* loop

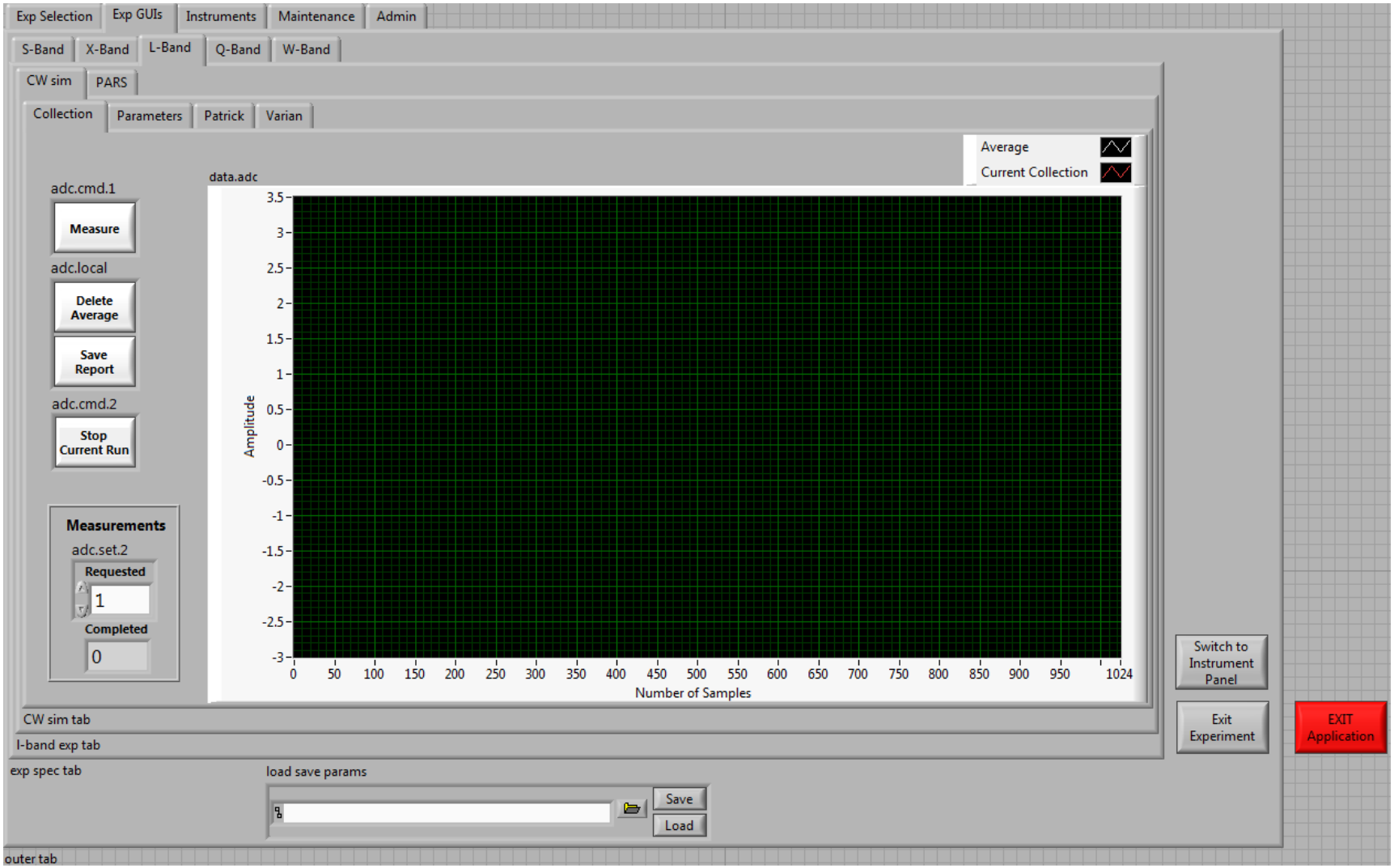


Figure: CW experiment GUI *Collection* page

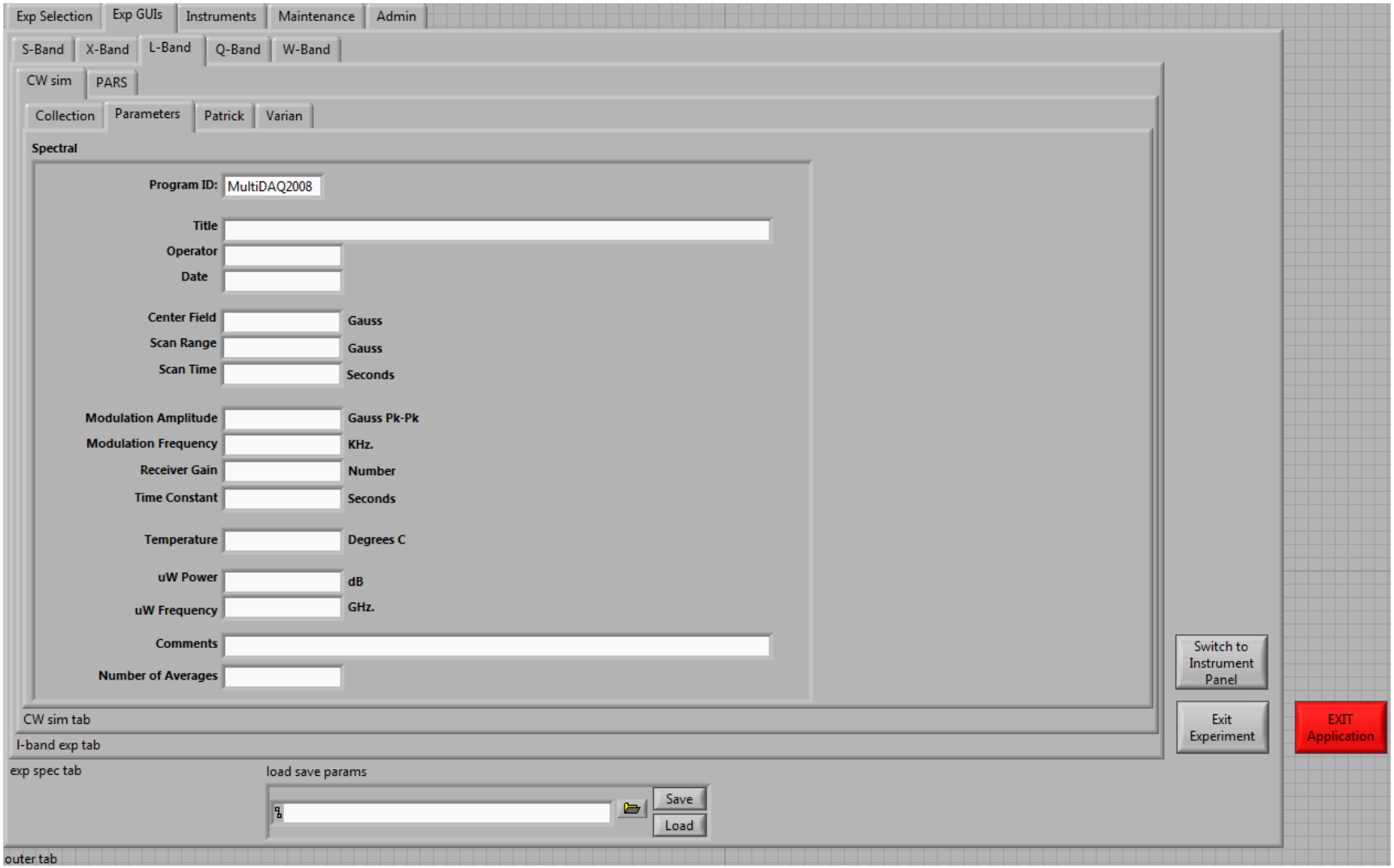


Figure: CW experiment GUI *Parameters* page

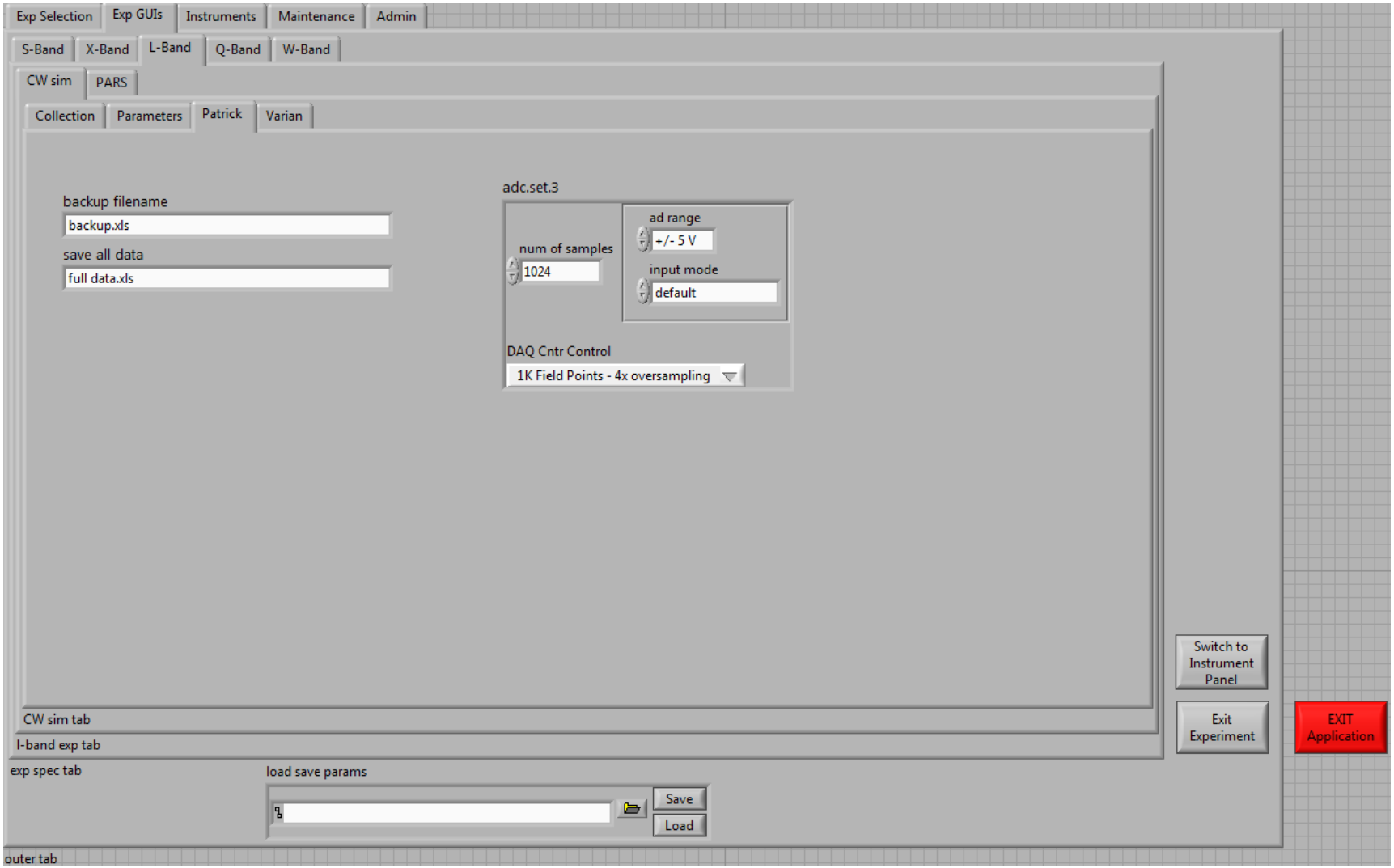


Figure: CW experiment GUI *Patrick* page

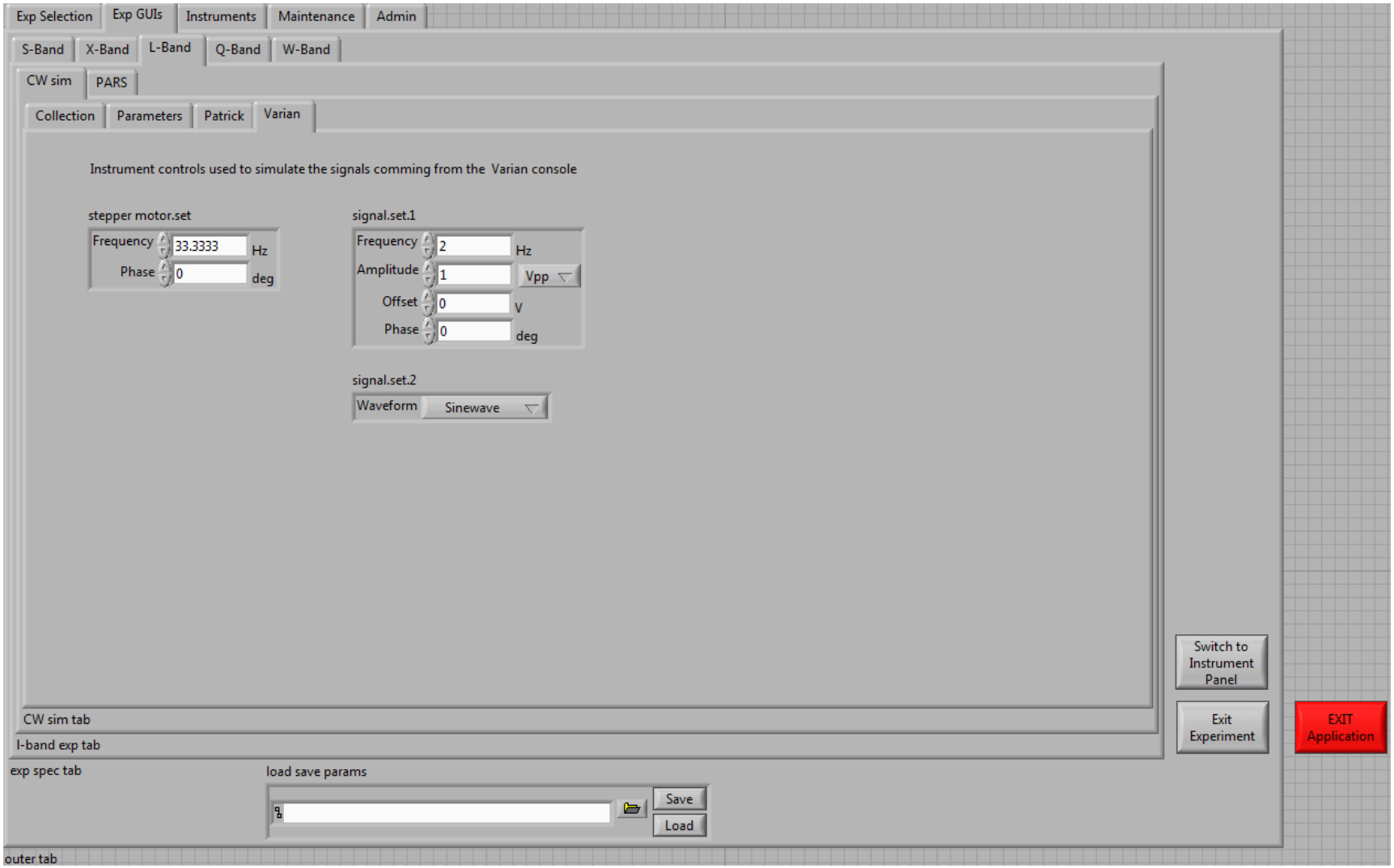


Figure: CW experiment GUI *Varian* page

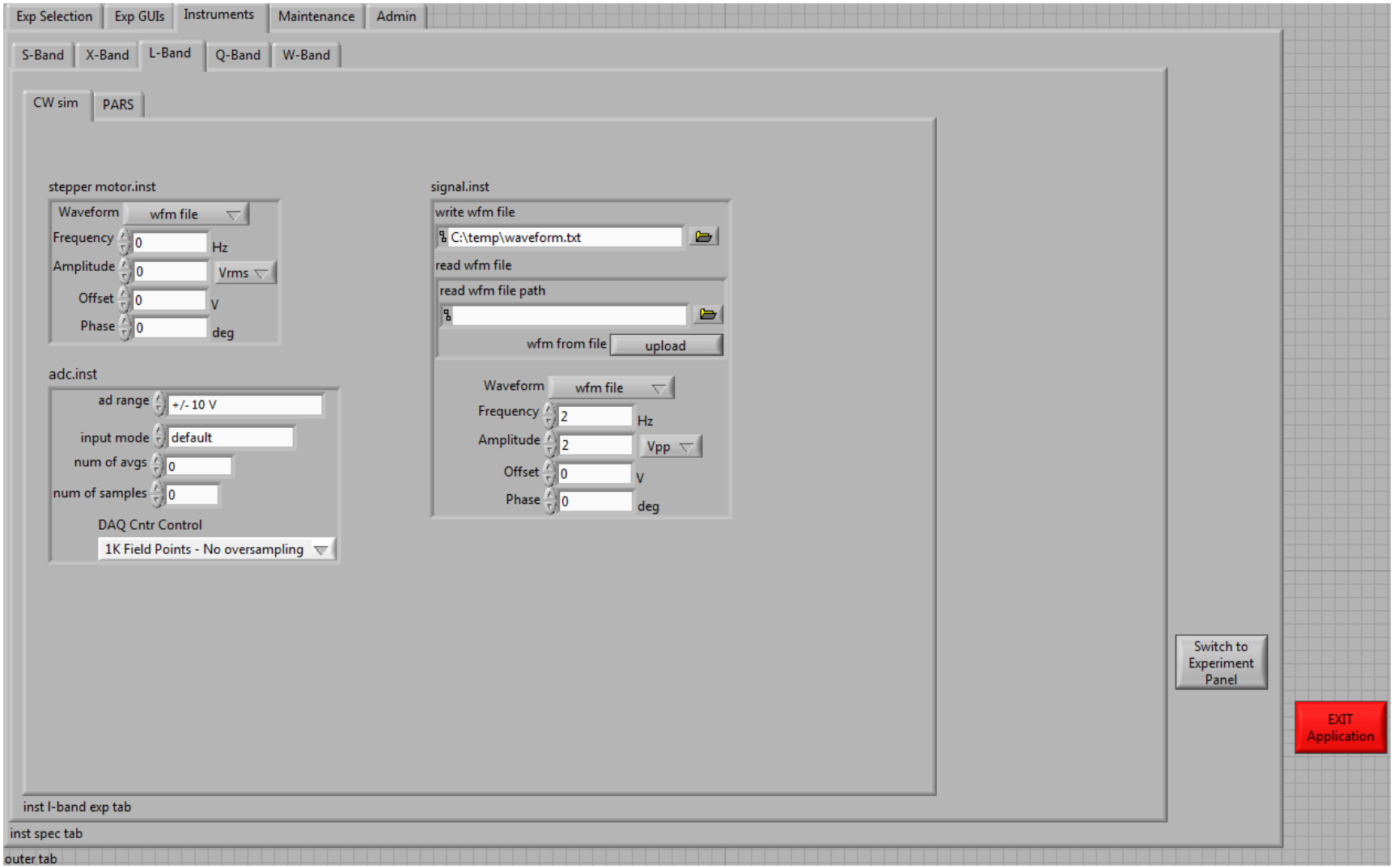


Figure: CW experiment GUI *Instrument* panel

Contents

Measurement Setup

Sample	:	Spin label
Microwave frequency	:	1.908 GHz (L-Band)
Microwave input power	:	120 µW
Center field	:	675 G
Sweep	:	105 G, *i.e.* 622.5 G .. 727.5 G
Number of averages	:	5
Acquisition time	:	$\approx$ 5:30 min

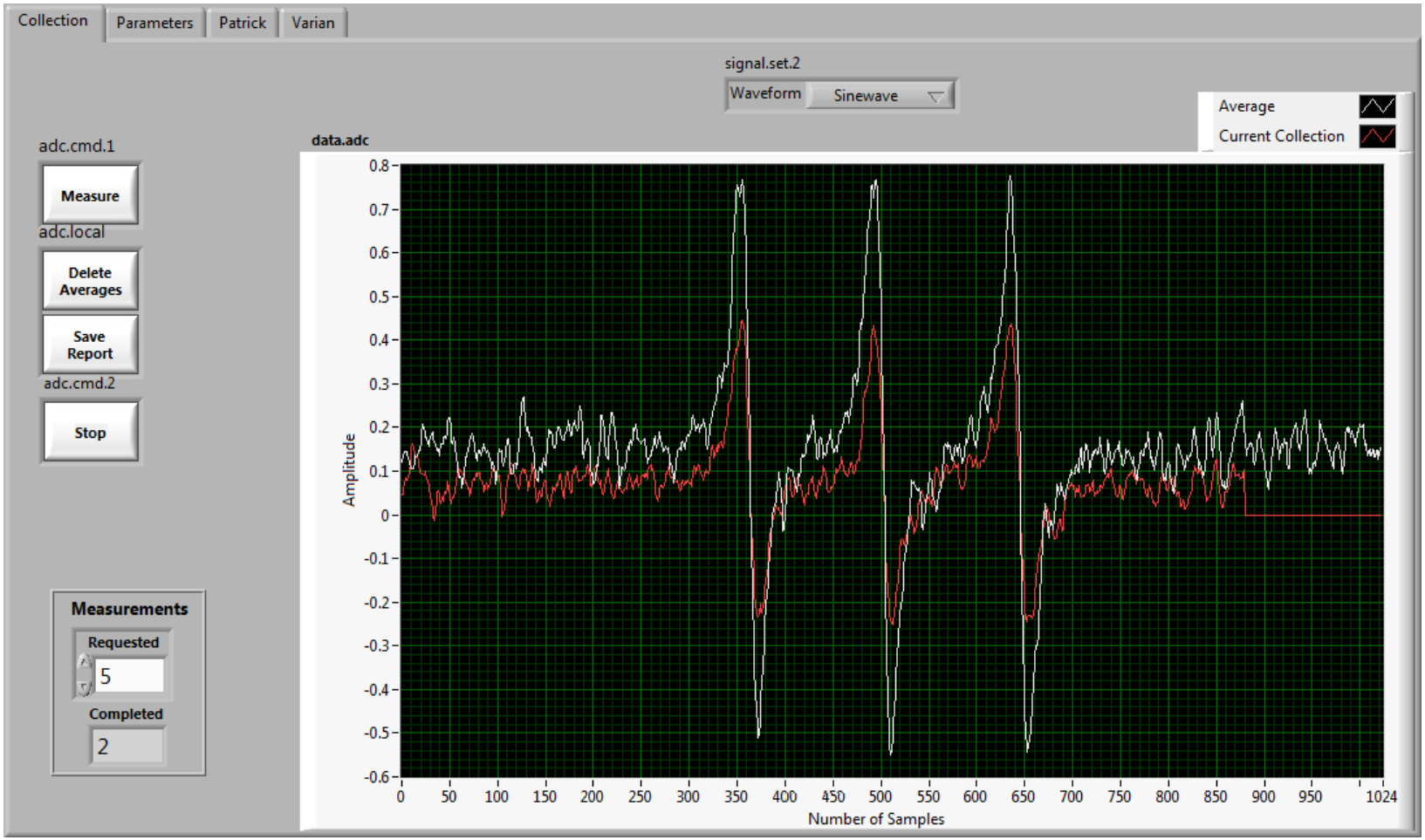


Figure: Collection page after 2nd collection

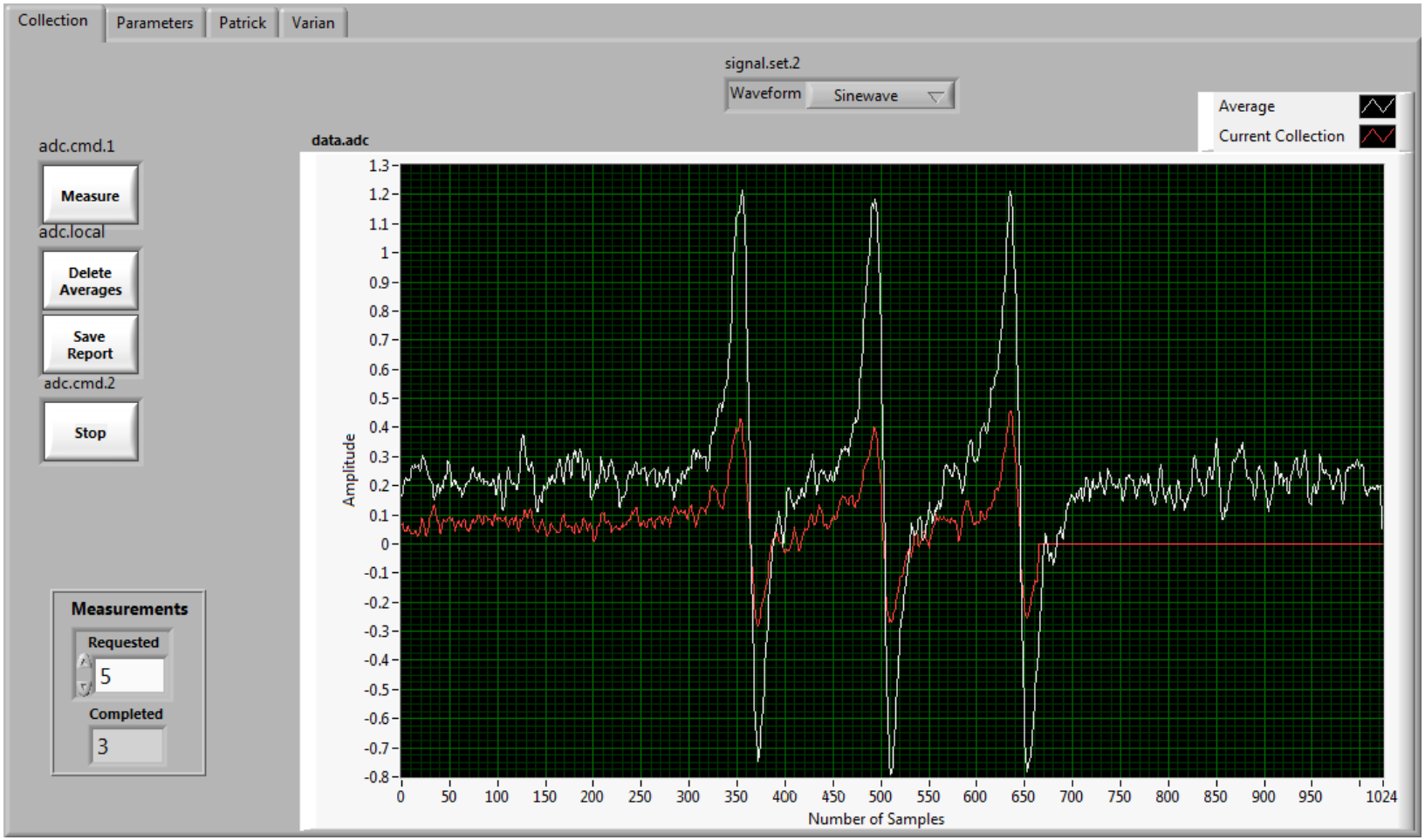


Figure: Collection page after 3rd collection

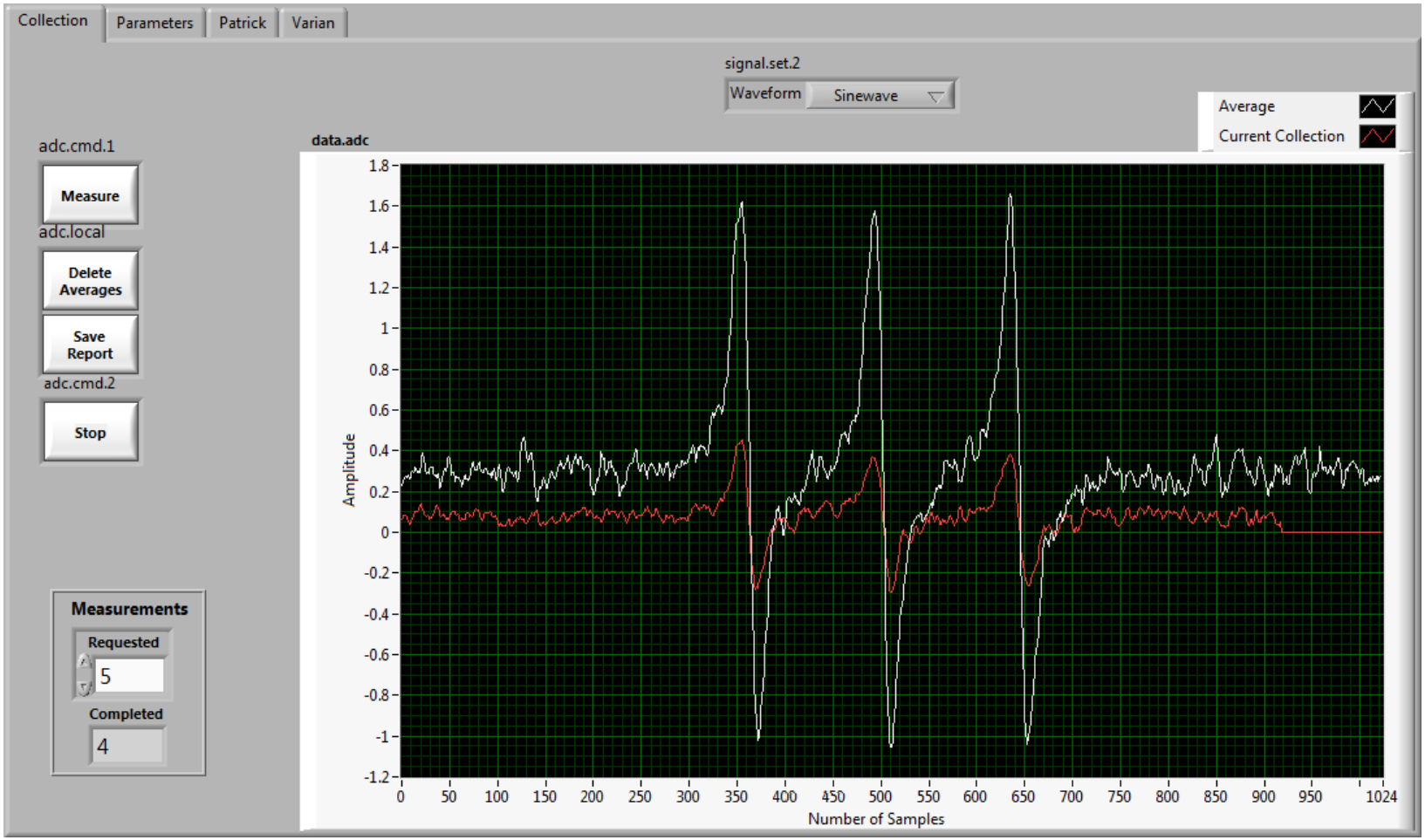


Figure: Collection page after 4th collection

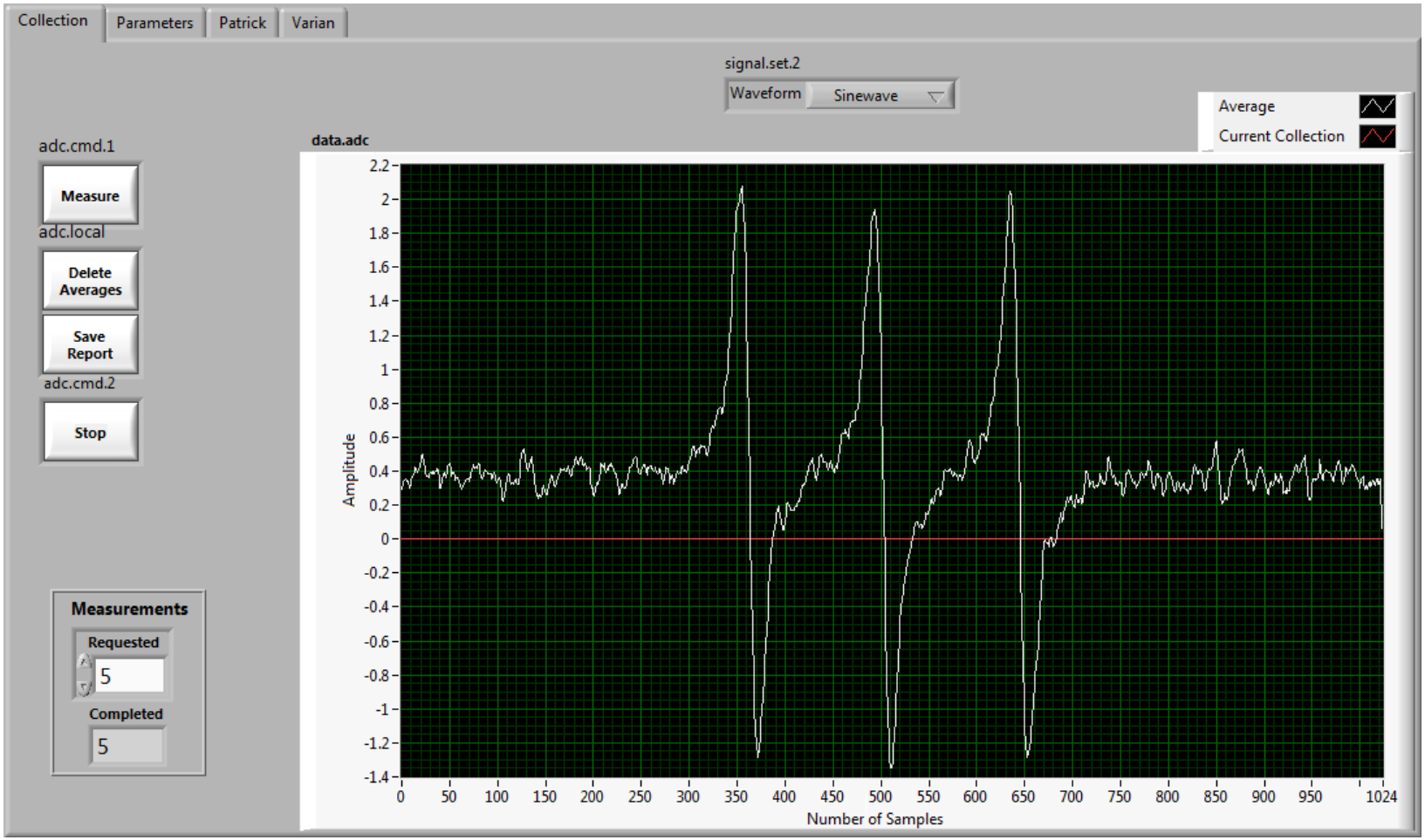


Figure: Collection page after 5th collection

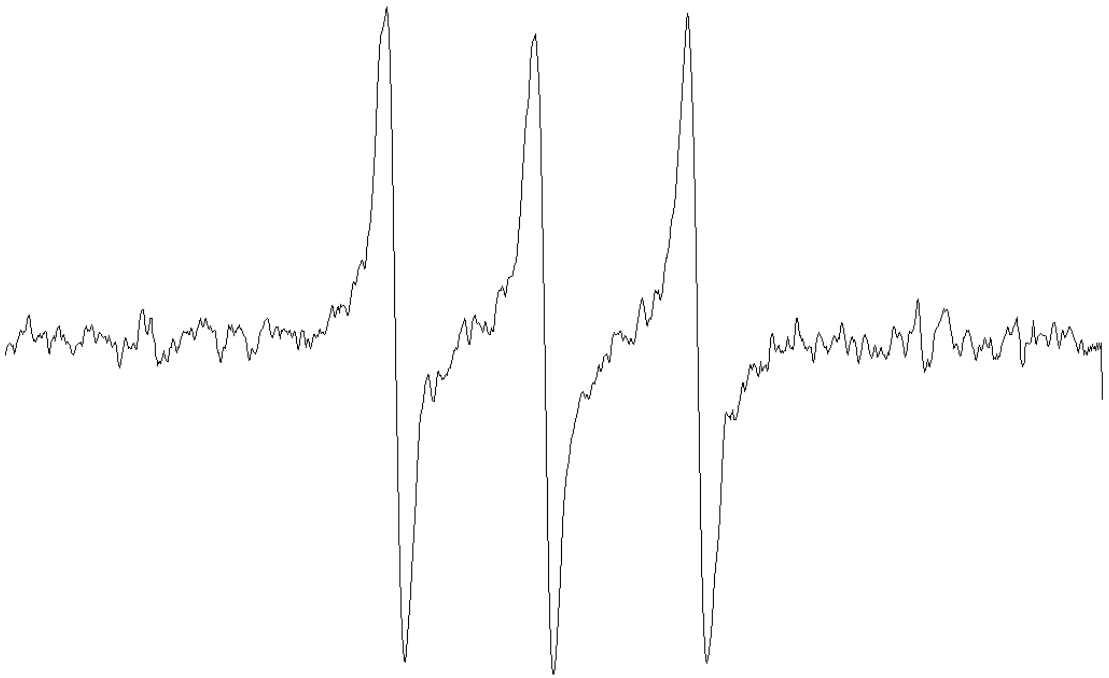


Figure: Plot of *full data.xls*

Contents

Key Features

- Can run on all spectrometers
- Can implement all experiments
- Faster experiment implementation
- Improved modularity and flexibility
- 32 and 64-bit driver capability
- Multi-core processors and multi-threading capable

Issues

- Improve thread allocation
- Decrease time it takes to display acquired data
- Emphasize the common basis of experiments

Discoveries

- Shared variables work unreliable for some applications
- Module based concept for large projects
- STM with case structure to avoid polling for new messages

Future Improvements

- Adaptive Signal Averaging Technique
- Technical Data Management Streaming (TDMS)
- Implement more drivers

Any Questions?

Many thanks for your attention!