Module Interface Specification for PolyHarmonics

Nolan Driessen

August 26, 2015

Contents

1 Introduction

The following document details the Module Interface Specifications for the implemented modules in PolyHarmonics. It is intended to ease navigation through the program for design and maintenance purposes.
Complementary documents include the System Requirement Specifications and Module Guide.

2 Module Hierarchy

The following table is taken directly from the Module Guide document for this project.

Table 1: Module Hierarchy

Level 1	Level 2	Level 3
Hardware-Hiding Module		
Behaviour-Hiding Module	Input Module	Input Finding Module Input Data Module
	Transformed Signal Data Module Plot Data Module Control Module Output Module Filtered Data Module	
Software Decision Module	Signal Transforming Module Filtering Module	

3 MIS of Control Module

3.1 Module Name: main

3.2 Uses

3.2.1 Imported Data Types

3.2.2 Imported Access Programs

Uses Input Finding Module **Imports** parameterSearch
Uses Signal Transform Module **Imports** transform
Uses Plot Data Module **Imports** plot
Uses Output Module **Imports** output
Uses Filtering Module **Imports** filter

3.3 Interface Syntax

3.3.1 Exported Data Types

3.3.2 Exported Access Programs

Name	In	Out	Exceptions
main	–	–	Files not found

3.4 Interface Semantics

3.4.1 State Variables

3.4.2 Environment Variables

path : Location of the directory to be searched for TDMS files.
win: 2D sequence of pixels displayed on screen

3.4.3 Assumption

3.4.4 Invariant

None

3.4.5 Access Program Semantics

Main consists of a sequence of calls to other modules access programs. The sequence is as follows: InputFinding : parameterSearch, Signal Transform Module : transform, Plot Data Module : plot, Output Module : output and Filtering Module : filter. Main will call each of these access programs for a specific set of tests which it will identify through the use of the "path" environment variable. Additionally Main will print messages by modifying the "win" environment variable to communicate any errors or warnings as necessary.

4 MIS of Input Finding Module

4.1 Module Name: InputFinding

4.2 Uses

4.2.1 Imported Data Types

4.2.2 Imported Access Programs

Uses Input Data Module **Imports** Set_Start_Freq
Uses Input Data Module **Imports** Set_Stop_Freq
Uses Input Data Module **Imports** Set_Step_Freq
Uses Input Data Module **Imports** Set_TDMS_Time
Uses Input Data Module **Imports** Set_TDMS_Data

4.3 Interface Syntax

4.3.1 Exported Data Types

4.3.2 Exported Access Programs

Name	In	Out	Exceptions
parameterSearch	string	InputData	Start_Freq out of range Stop_Freq out of range Step_Freq out of range

4.4 Interface Semantics

4.4.1 State Variables

4.4.2 Environment Variables

TDMSFile : A set of files to be read.

4.4.3 Assumption

4.4.4 Invariant

4.4.5 Access Program Semantics

parameterSearch will collect the input for the system in order for InputData to organize it. For each file within the "TDMSFile" environment variable it will read and store the information into a list for the InputData module to use, along with a start stop and step frequency for the set. parameterSearch will use the access programs provided by InputData in order to populate the state of InputData.

[Explicitly mention that the state of input format will be changed. These module names do not match what is in your MG. Is InputFormat the Input Data module? I like the name Input Data better, since InputFormat implies input format hiding, which seems to beh the job of Input Finding. I see that you use both the names Input Data module and InputFormat. Why have two names? —SS]

[Changed the name of each module to match, additionally wrote about changing the state in terms of state variables. Can be more specific if it would be better, ie) uses Set_Start_Freq to assign a value to the starting frequency but I believe that is apparent from the imported access programs. —ND]

5 MIS of Input Data Module

5.1 Module Name: InputData

5.2 Uses

5.2.1 Imported Data Types

5.2.2 Imported Access Programs

5.3 Interface Syntax

5.3.1 Exported Data Types

5.3.2 Exported Access Programs

Name	In	Out	Exceptions
Set_Start_Freq	int	–	–
Set_Stop_Freq	int	–	–
Set_Step_Freq	int	–	–
Set_TDMS_Time	list of lists	–	–
Set_TDMS_Data	list of lists	–	–
Get_Start_Freq	–	int	–
Get_Stop_Freq	–	int	–
Get_Step_Freq	–	int	–
Get_TDMS_Time	–	list of lists	–
Get_TDMS_Data	–	list of lists	–

5.4 Interface Semantics

5.4.1 State Variables

Start_Freq : int
Stop_Freq : int
Step_Freq : int
TDMS_Time : list of lists
TDMS_Data : list of lists

5.4.2 Environment Variables

5.4.3 Assumption

None of the Get access programs will be called before their respective Set function.

5.4.4 Invariant

5.4.5 Access Program Semantics

Set: **Input:** Each Set function will take a parameter as input according to which function was called.
Transition: Each set function will modify the state of the corresponding parameter.

Get: **Input:** Each Get function requires no input
Output: Each Get function will return the value of the corresponding parameter.

6 MIS of Transformed Signal Data Module

6.1 Module Name: TransformedSignalData

6.2 Uses

6.2.1 Imported Data Types

6.2.2 Imported Access Programs

6.3 Interface Syntax

6.3.1 Exported Data Types

6.3.2 Exported Access Programs

Name	In	Out	Exceptions
Set_Original_Transformed_Data	list of lists	–	–
Set_Filtered_Transformed_Data	list of lists	–	–
Set_Original_Frequency_Data	list of lists	–	–
Set_Filtered_Frequency_Data	list of lists	–	–
Get_Original_Transformed_Data	–	list of lists	–
Get_Filtered_Transformed_Data	–	list of lists	–
Get_Original_Frequency_Data	–	list of lists	–
Get_Filtered_Frequency_Data	–	list of lists	–

6.4 Interface Semantics

6.4.1 State Variables

Original_Transformed_Data : list of lists
Filtered_Transformed_Data : list of lists
Original_Frequency_Data : list of lists
Filtered_Frequency_Data : list of lists

6.4.2 Environment Variables

6.4.3 Assumption

None of the Get access programs will be called before their respective Set function.

6.4.4 Invariant

6.4.5 Access Program Semantics

Set: **Input:** Each Set function will take a parameter as input according to which function was called.
Transition: Each Set function will modify the state of the corresponding parameter.

Get: **Input:** Each Get function requires no input.
Output: Each Get function will return the value of the corresponding parameter.

7 MIS of Signal Transform Module

7.1 Module Name: SignalTransform

7.2 Uses

7.2.1 Imported Data Types

7.2.2 Imported Access Programs

Uses Input Data Module **Imports** Get_TDMS_Time
Uses Input Data Module **Imports** Get_TDMS_Data
Uses Filtered Data Module **Imports** Get_Detailed_Coefficients
Uses Transformed Signal Data Module **Imports** Set_Original_Transformed_Data
Uses Transformed Signal Data Module **Imports** Set_Filtered_Transformed_Data
Uses Transformed Signal Data Module **Imports** Set_Original_Frequency_Data
Uses Transformed Signal Data Module **Imports** Set_Filtered_Frequency_Data

7.3 Interface Syntax

7.3.1 Exported Data Types

7.3.2 Exported Access Programs

Name	In	Out	Exceptions
transform	class	class	–

7.4 Interface Semantics

7.4.1 State Variables

7.4.2 Environment Variables

7.4.3 Assumption

7.4.4 Invariant

7.4.5 Access Program Semantics

Input:
The SignalTransform Module requires the TDMS files formatted into the proper data structure by the Input Data Module as well as the filtered data from the Filtered Data Module.
Transition:
The Transforming Module will modify the state of the Transformed Signal Data Module through its access programs and return the fully populated TransformedSignalData class.

8 MIS of Plot Data Module

8.1 Module Name: PlotData

8.2 Uses

8.2.1 Imported Data Types

8.2.2 Imported Access Programs

Uses Input Data Module **Imports** Get_Start_Freq
Uses Input Data Module **Imports** Get_Stop_Freq
Uses Input Data Module **Imports** Get_Step_Freq
Uses Input Data Module **Imports** Get_TDMS_Time
Uses Input Data Module **Imports** Get_TDMS_Data
Uses Transformed Signal Data Module **Imports** Get_Original_Transformed_Data
Uses Transformed Signal Data Module **Imports** Get_Original_Frequency_Data

8.3 Interface Syntax

8.3.1 Exported Data Types

8.3.2 Exported Access Programs

Name	In	Out	Exceptions
plot	list of lists	–	–

8.4 Interface Semantics

8.4.1 State Variables

8.4.2 Environment Variables

plots : A set of files that will be modified to each have a plot of an input signal and its transformed frequency.

8.4.3 Assumption

8.4.4 Invariant

8.4.5 Access Program Semantics

Input:
plot will accept a series of parameters from its imported access programs.
Transition:
plot will use the input given to it in order to modify each element of the "plots" environment variable.

9 MIS of Output Module

9.1 Module Name: Output

9.2 Uses

9.2.1 Imported Data Types

9.2.2 Imported Access Programs

Uses Input Data Module **Imports** Get_Start_Freq
Uses Input Data Module **Imports** Get_Stop_Freq
Uses Input Data Module **Imports** Get_Step_Freq
Uses Input Data Module **Imports** Get_TDMS_Data
Uses Transformed Signal Data Module **Imports** Get_Original_Transformed_Data
Uses Transformed Signal Data Module **Imports** Get_Filtered_Transformed_Data
Uses Transformed Signal Data Module **Imports** Get_Original_Frequency_Data
Uses Transformed Signal Data Module **Imports** Get_Filtered_Frequency_Data

9.3 Interface Syntax

9.3.1 Exported Data Types

9.3.2 Exported Access Programs

Name	In	Out	Exceptions
output	int,list of lists	–	–

9.4 Interface Semantics

9.4.1 State Variables

9.4.2 Environment Variables

output_txt : A series of text files that Output will create.

9.4.3 Assumption

9.4.4 Invariant

9.4.5 Access Program Semantics

Input:
The Output Module requires data from the Input Data module and the Transformed Signal Data module which it will receive through their access programs.
Transition:
The Output Module modifies its "output_txt" environment variable by producing three text files.

10 MIS of Filtering Module

10.1 Module Name: Filtering

10.2 Uses

10.2.1 Imported Data Types

10.2.2 Imported Access Programs

Uses Input Data Module **Imports** Get_TDMS_Data
Uses Filtered Data Module **Imports** Set_Approximate_Coefficients
Uses Filtered Data Module **Imports** Set_Detailed_Coefficients

10.3 Interface Syntax

10.3.1 Exported Data Types

10.3.2 Exported Access Programs

Name	In	Out	Exceptions
filter	–	–	–

10.4 Interface Semantics

10.4.1 State Variables

10.4.2 Environment Variables

10.4.3 Assumption

10.4.4 Invariant

10.4.5 Access Program Semantics

Input:
The Filtering Module requires the data stored within the the Input Data Module which it will acquire through its access programs.
Transition:
The Filtering Module will populate the state of the Filtered Data Module through use of its access programs.

11 MIS of Filtered Data Module

11.1 Module Name: FilteredData

11.2 Uses

11.2.1 Imported Data Types

11.2.2 Imported Access Programs

11.3 Interface Syntax

11.3.1 Exported Data Types

11.3.2 Exported Access Programs

Name	In	Out	Exceptions
Set_Approximate_Coefficients	list of lists	–	–
Set_Detailed_Coefficients	list of lists	–	–
Get_Approximate_Coefficients	–	list of lists	–
Get_Detailed_Coefficients	–	list of lists	–

11.4 Interface Semantics

11.4.1 State Variables

Approximate_Coefficients : list of lists
Detailed_Coefficients : lists of lists

11.4.2 Environment Variables

11.4.3 Assumption

None of the Get access programs will be called before their respective Set function.

11.4.4 Invariant

11.4.5 Access Program Semantics

Set: **Input:** Each Set function will take a parameter as input according to which function was called.
Transition: Each Set function will modify the state of the corresponding parameter.

Get: **Input:** Each Get function requires no input.
Output: Each Get function will return the value of the corresponding parameter.

A Exceptions

Table 2: Possible Exceptions

Message ID	**Error Message**
Files not found	error: The given directory contains no files suitable for analysis
Start_Freq out of range	error: Start Frequency must be > 0
Start_Freq out of range	error: Stop Frequency must be > 0
Start_Freq out of range	error: Step Frequency must be > 0