

ОТЧЕТ ПО РАЗРАБОТКЕ АВТОМАТИЗИРОВАННОЙ
СИСТЕМЫ СБОРА И ОБРАБОТКИ ДАННЫХ ЭКСПЕРИМЕНТА
НА БАЗЕ ПАКЕТА LABVIEW

Студент: Черняев Сергей Аркадьевич БИТ151
Научный руководитель: д.ф.-м.н., профессор
Арутюнов К.Ю., зав. лаб., ИФП им. П.Л. Капицы
РАН

Содержание

1 Поставленная задача и выбор среды разработки	3
2 Экспериментальная установка	4
3 Приборы и образец	5
3.1 Источник тока	5
3.2 Мультиметр	5
3.3 Фазочувствительный детектор	6
3.4 DAQ	6
3.5 Предусилители тока и напряжения	8
3.6 Образец	8
4 Программа сбора и обработки экспериментальных данных	8
4.1 Фронтальная панель	8
4.1.1 Комментарии и папка	10
4.1.2 Переключатель между начальными настройками ПО и Lock-In	10
4.1.3 Показания Lock-In	12
4.1.4 График измерений относительно времени TIME GRAPH	13
4.1.5 График показателей приборов относительно других показателей MULTIPLE GRAPHS	13
4.1.6 Масштабы	14
4.1.7 Коэффициенты подстраиваемости	14
4.1.8 Индикаторы измерений	15
4.1.9 Панель админа	15
4.2 Блок-Схема	16
4.2.1 Инициализация приборов	16
4.2.2 Определение начального и конечного тока и их проверка	17
4.2.3 Расчет шага	18
4.2.4 Создание папки и файла измерений	19
4.2.5 Временная задержка и настройка параметров	22
4.2.6 Предизмерительные проверки	23
4.2.7 Основной измерительный цикл	23
5 Список использованных источников	35
6 Благодарности	35

1 Поставленная задача и выбор среды разработки

Для любого эксперимента нужно иметь соответствующее программное обеспечение (ПО) для настройки и контроля измерительных приборов в течение эксперимента, сбора и анализа полученных данных и записи их в отдельный файл для дальнейшей обработки и визуализации в таких программах как Excel, gnuplot, Origin и тд.

Данное ПО можно реализовать с помощью различных языков программирования (Python, C++, и тд), но перед тем как определить подходящий язык и среду разработки, надо определить объем задачи и её особенности (Количество приборов, наличие или отсутствие графического интерфейса, поддержка связи с приборами и тд). Задача, которая была поставлена мне моим научным руководителем подразумевала следующие важные аспекты, которые надо было учесть:

1. Наличие более двух измерительных приборов и возможное увеличение их количества в будущем
2. Обязательное наличие графического интерфейса
3. Кросплатформенность разработанного ПО, т.е. возможность его запуска не только на Windows машинах, но и на UNIX и LINUX подобных системах.
4. Представление результатов в графическом виде в режиме в реального времени

Данные задачи были расширены различными уточнениями, но уже на основе данного небольшого списка аспектов наш выбор остановился на LabVIEW.

Во первых, данный продукт был специально разработан компанией National Instruments для решения технических задач тестирования, измерения и управления с быстрым доступом к оборудованию и результатам обработки. Являясь полноценной средой разработкой с компиляцией "на лету" он сочетает в себе как простую разработку графического интерфейса без подключения отдельных библиотек, так и понятный графический язык программирования "G". Созданные в LabVIEW программы могут быть запущены под любой операционной системой без скачивания всего пакета, а имея всего лишь соответствующий Run-Time Engine.

Во вторых, имелся в наличии достаточно обширный менеджер пакетов, что значительно упрощало поиск библиотек приборов. Можно было не искать список поддерживаемых SPI команд для каждого из них.

В третьих, при установке LabVIEW ставится отдельное ПО MAX (Measurement & Automation Explore), которое позволяет настроить все подключенные устройства (их порты, скорость обмена пакетами, названия и тд).

Из изложенных выше пунктов видно, что LabVIEW отлично подходит для нашей разработки и мы выбрали LabVIEW 2015 SP1 для написания программы. Дальше будут показаны его особенности и некоторые недостатки, которые обнаружились при создании ПО.

2 Экспериментальная установка

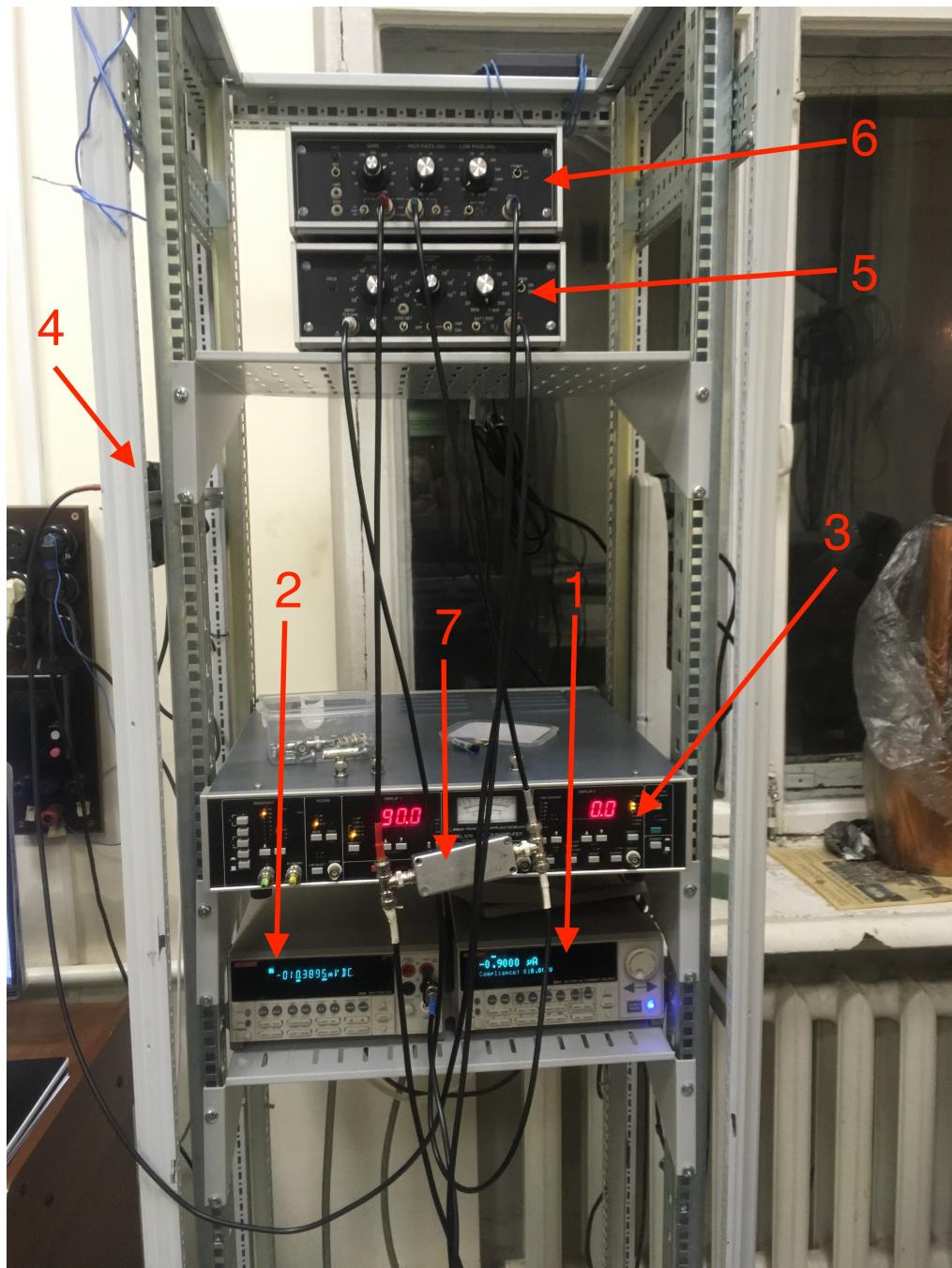


Рис. 1: Экспериментальная установка на которой тестировались приборы и разработанное ПО

1. 6221 Keithley Precision DC and AC+DC Low Noise Current Source
2. 2000 Keithley Precision Multimeter
3. 5110 EG&G Princeton Applied Research Lock-In Amplifier
4. KUSB-3116 Keithley 16-Bit USB DAQ Module
5. 1201 DL Instruments Low Noise Voltage Preamplifier
6. 1211 DL Instruments Current Preamplifier
7. Образец

Далее будет представлены увеличенные картинки приборов и небольшая информация об их применении и поиске соответствующих библиотек для их работы.

3 Приборы и образец

3.1 Источник тока



Рис. 2: 6221 Keithley Precision DC and AC+DC Low Noise Current Source

С него подавался постоянный ток на образец для дальнейшего анализа напряжения на образце. Подключается к компьютеру по GPIB, что позволяет одновременно отправлять и принимать пакеты с прибора. Поиск библиотек для источника тока оказался быстрым из-за большой популярности использования среди научного сообщества.

3.2 Мультиметр



Рис. 3: 2000 Keithley Precision Multimeter

Данный прибор использовался для измерения напряжения приходящего с предусилителя напряжения. Также подключается по GPIB и имеет программную поддержку со стороны National Instruments, что облегчало поиск соответствующих библиотек.

3.3 Фазочувствительный детектор



Рис. 4: 5110 EG&G Princeton Applied Research Lock-In Amplifier

Данный прибор позволяет выделить сигнал определенной частоты (задаваемая опорным сигналом) среди некоторого диапазона. Тоже подключался с помощью GPIB, но имел некоторые неприятности при настройки.

Во первых, официальный сайт National Instruments не имел официальных библиотек, поэтому пришлось воспользоваться сайтом производителя. К сожалению не производила их обновления в течении долго времени, что приводило к ошибке соединения с прибором, а конкретнее в блоке открытия порта соединения. Анализ при помощи MAX показал, что Lock-In прекрасно отзывается на CPI команды, что означало, что баг находился в самом блоке открытия порта соединения. Дальнейший парсинг показал, что проблема была в функции создания сессии с прибором "VISA open". Из-за того, что компания не следила за обновлениями, данная функция больше не поддерживалась современными версиями LabVIEW. Ошибка была решена заменой данной функции на новую с аналогичным названием и прибор стабильно заработал.

Во вторых, после каждой сессии прибор всегда сбрасывает порт GPIB с 11 на 12, что является небольшим неудобством, заключающимся в том, что при каждом включении прибора в сеть приходится переводить его на 11 порт вручную на фронтальной панели Lock-In. Порт 11 был выбран для всех приборов подключаемых по GPIB из системных соображений.

3.4 DAQ



Рис. 5: KUSB-3116 Keithley 16-Bit USB DAQ Module

Данный прибор имеет 16 аналоговых входов и 4 аналоговых выхода, а также несколько цифровых входа и выхода. В нашем ПО были задействованы 5 аналоговых входов (AD Ch0 - AD Ch4). DAQ позволяет собирать информацию с большого количества каналов одновременно, что делает его незаменимым для экспериментов с большим количеством образцов. К величайшему сожалению данный прибор оказался самым "капризным" из всех. Эпопея с его подключением достойна отдельного отчета, поэтому выделию только основные проблемы.

Во первых, так как компания Tektronix приобрела компанию Keithley Instruments и прекратила поддержку данного прибора, то изначально библиотеки имелись только на диске, который шел с ним в комплекте. Диск имел exe файл, который записывал в корень LabVIEW системные файлы для работы с прибором. Это являлось огромным минусом, потому что установочная программа требовала наличие LabVIEW 7.1, а ПО разрабатывалось на LabVIEW 2015. Даже если бы старая версия стояла на рабочем ПК, принудительный перенос библиотек в соответствующие папки LabVIEW 2015 не привел бы к положительным результатам, так как сами библиотеки были созданы с помощью LabVIEW 7.1, чьи файлы прекратили поддержку уже в LabVIEW 2010. Поиск оптимального решения этой, на первый взгляд кажущейся неразрешимой, проблемы начался с анализа мою платы прибора. Очень часто производители экспериментального оборудования используют одинаковые платы, меняя только логотип компании. Данный подход оказался плодотворным и привел к оригинальному создателю данных устройств Measurement Computing, которое производит аналогичные устройства (Рис. 6).



Рис. 6: DT9834 Series High-Speed Synchronous USB Device

Можно заметить поразительное сходство двух приборов, что навело на мысль использовать их библиотеки. Это оказалось успешным, хотя и подразумевало запуска аналогичного exe файла, который устанавливал в корень LabVIEW системные файлы прибора. К счастью требовалась LabVIEW 8.1, которая стояла на другом ПК, и созданные в ней файлы имели поддержку в LabVIEW 2015. Данный exe также ставил отладочные программы и ПО для тестирования каналов прибора. Казалось, что на этом мучения должны были закончиться, но это прибор все равно отказывался работать.

Второй глобальной проблемой оказалась установка драйверов для опознавания как LabVIEW, так и самим ПК нашего DAQ. Обычно данная проблема решается с помощью MAX, который упрощает данный процесс, но к сожалению он не справился с этой задачей, так как он никак не обнаруживал прибор, было решено ставить драйвера вручную. Те которые были на диске оказались нерабочими, что было неудивительно, но их установка была до обнаружения аналогичного прибора от Measurement Computing. Драйвера для DT9834 оказались подходящими и прибор заработал.

3.5 Предусилители тока и напряжения



Рис. 7: 1211 DL Instruments Current Preamplifier и 1201 DL Instruments Low Noise Voltage Preamplifier

Данные устройства используются для усиления сигнала приходящего с образца, так как предполагается данные сигналы будут достаточно малыми для анализа без усилителей. Усиленный сигнал идет на DAQ и мультиметр.

3.6 Образец

Для тестирования ПО и приборов в качестве образца выступал резистор в 1кОм.

4 Программа сбора и обработки экспериментальных данных

Теперь рассмотрим саму программу, а точнее её графический интерфейс и алгоритм. В последующих пунктах будет представлено не только описание, но и критика и возможные улучшения ПО.

В процессе разработки появилось 9 Pre-Alpha, 6 Alpha и 3 Beta версий ПО. На данный момент происходит тестирование Beta 1.2 и именно она разобрана в данном отчете.

4.1 Фронтальная панель

Во время разработки в LabVIEW одновременно редактируется блок-схема и фронтальная панель (графический интерфейс). Рассмотрим сначала её и отдельные составляющие подробнее, чтобы упростить объяснение блок-схемы.

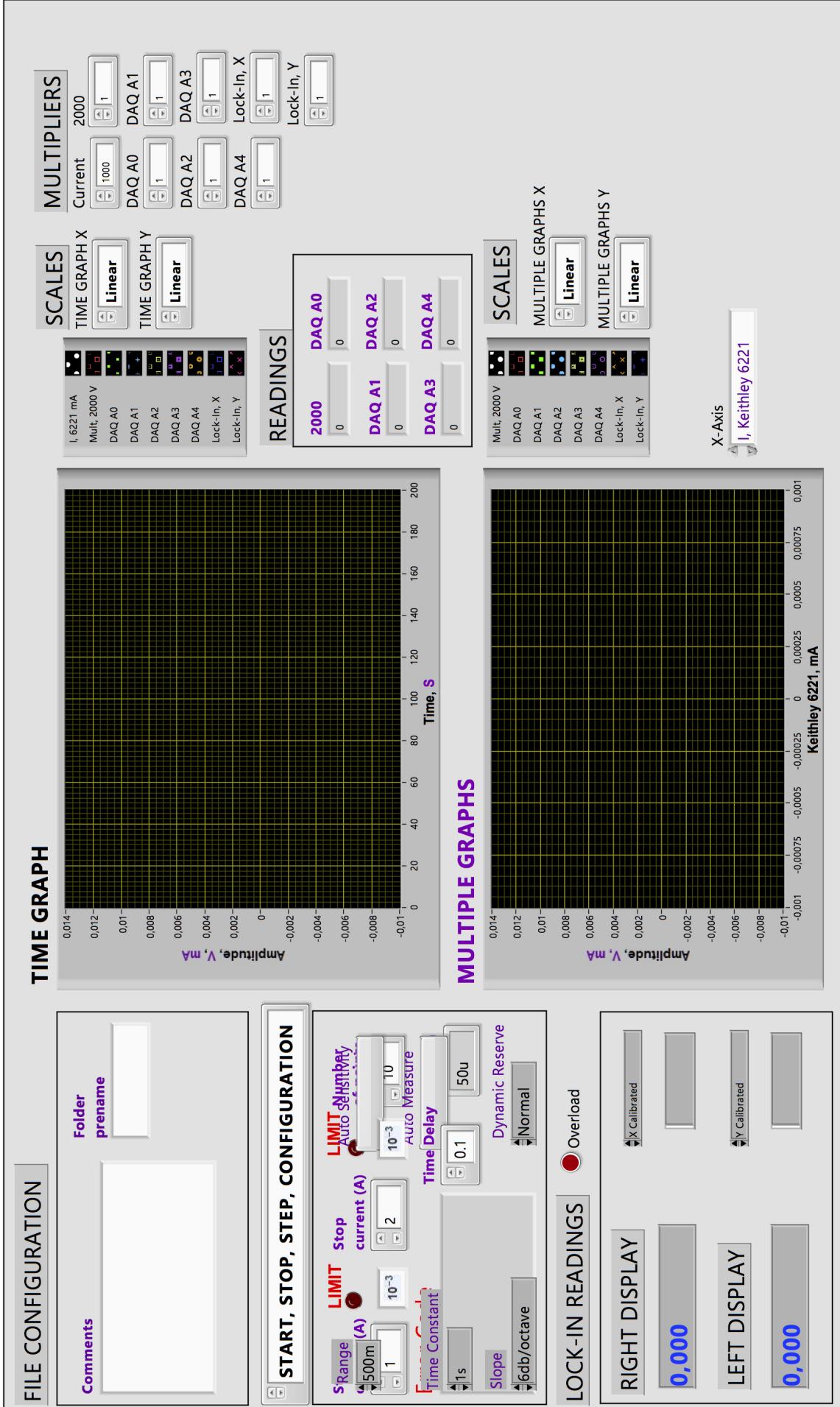


Рис. 8: Фронтальная панель ПО

4.1.1 Комментарии и папка

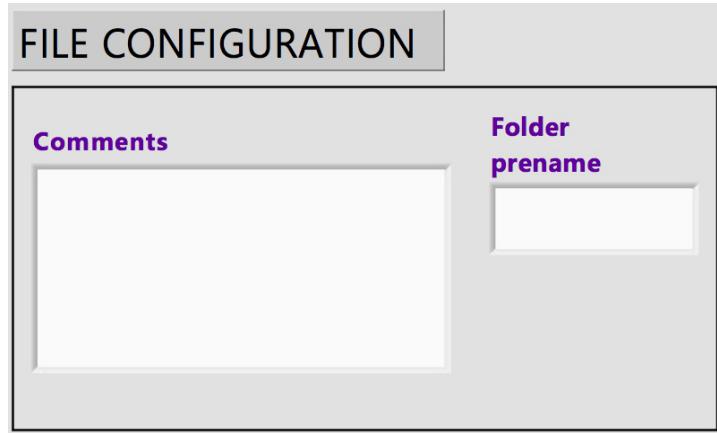


Рис. 9: Запись комментариев и создание названия папки

Перед тем как писать в файл эксперимента значения измерительных приборов надо указать краткие комментарии, которые могут быть полезны для дальнейшей обработки результатов. Как пример можно привести название образца или режима измерения. Именно эта информация записывается пользователем в окошке "Comments"

"Folder prename" позволяет пользователю добавить к названию папки, где хранятся файлы с экспериментальными данными, некоторое слово, которое позволит в дальнейшем быстро найти те или иные результаты измерений.

Подробно о том как создаются папка и файл смотрите ниже в пункте 4.2.4.

4.1.2 Переключатель между начальными настройками ПО и Lock-In

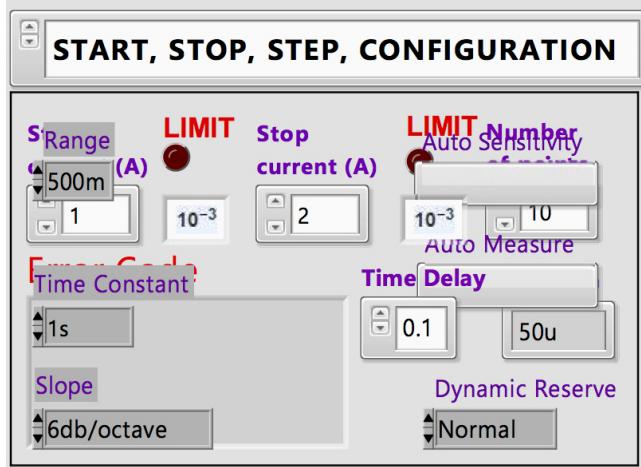


Рис. 10: Установка начальных параметров работы ПО и переключение на настройку Lock-In

Данная "каша" из индикаторов и контроллеров может вызвать эстетический ужас, но во время работы программы такого эффекта не будет, но начнем объяснение данной особенности самого верхнего переключателя:



Рис. 11: Переключатель между начальными настройками ПО и параметрами Lock-In

Данный переключатель позволяет переходить из вкладки настройки начальных значений работы программы (**START, STOP, STEP, CONFIGURATION**) к вкладке конфигурации Lock-In (**PRINCETON LOCK-IN AMP**). Далее я расскажу о каждой из этих вкладок и зачем было создано такое переключение.

Рассмотрим сначала START, STOP, STEP, CONFIGURATION:

Эта вкладка позволяет нам задать начальное **Start current (A)** и конечное значение тока **Stop current (A)** для источника тока, количество точек на графике **Number of points** и временную задержку между этими точками в секундах **Time Delay**. Так как в научном сообществе любят использовать физические приставки не в виде сочетаний букв (мк, н, м и тд), а писать степень десятки, то было решено сделать контроллер выбора соответствующей приставки. К сожалению LabVIEW 2015 не умеет распознавать верхний и нижний регистры, поэтому был создан контроллер выбора картинок, где каждой картинке ставилась в соответствие физическая приставка (Рис. 12). Подробнее об этом будет рассказано при обсуждении блок-схемы в пункте 4.2.2.

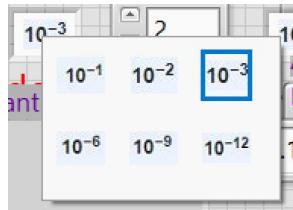


Рис. 12: Контроллер выбора физических приставок

Также имеется пару индикаторов. Две лампочки **LIMIT**, которые загораются, когда введенное значение начального или конечного тока превышают 10mA. Данное ограничение было согласовано с научным руководителем дабы предотвратить несчастные случаи.

Step - индикатор рассчитанного шага инкрементирования для тока по введенному значению количества точек.
Error Code - в данном окне выводится сообщение о настройки значений тока, а точнее "Stop(Start) current level exceeds the limit: 10mA" если начальное или конечное значение тока превышают 10mA соответственно или "Stop(Start) current level is OK" если все в порядке.

Теперь перейдем к рассмотрению PRINCETON LOCK-IN AMP:

Данная вкладка имеет контроллеры управления Lock-In, которые можно будет изменять в течение работы программы. К ним относятся следующие настройки:

Range - настройка чувствительности измерений
Time Constant - временная задержка между измерениями Lock-In
Slope - настройка фильтра низких частот
Dynamic Reserve - настройка динамического резерва сигнала

Теперь ответим на основной вопрос: зачем такая "каша" из настроек? Изначально они были разделены, но научным руководителем была поставлена задача выноса настроек для Lock-In в отдельную подпрограмму (subVI). Данный подход мог вызвать трудности по синхронизации работы двух программ, а также передачи данных, поэтому был предложен более элегантный способ.

Значения начального и конечного тока, количество точек и временная задержка между ними конфигурируются один раз перед запуском программы, поэтому их можно скрыть, так как корректировать мы их уже не будем. Именно так родилась идея о двух вкладках, благодаря которым мы можем переходить от одних настроек к другим, не загромождая экран. Lock-In настраивается с фронтальной панели прибора, поэтому не нуждается в конфигурации перед запуском.

4.1.3 Показания Lock-In

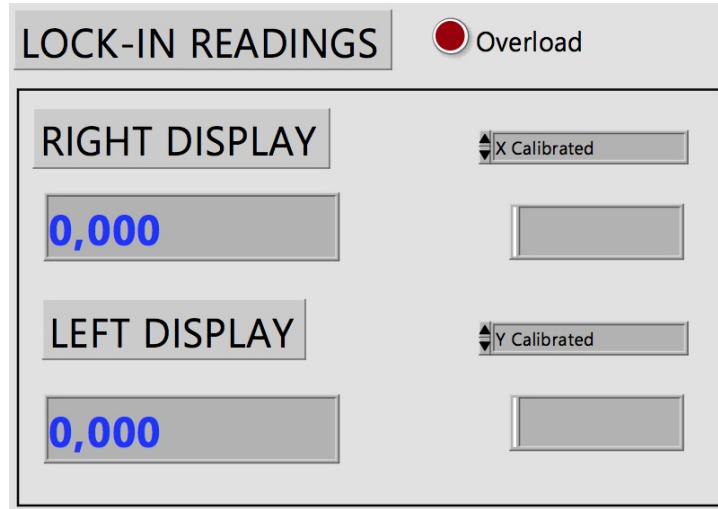


Рис. 13: Индикаторы для отображения показаний Lock-In

Теперь рассмотрим часть графического интерфейса ПО с показаниями измерений Lock-In. У данного прибора имеется два дисплея, поэтому, сохранив такой же вид, было сделано два больших индикатора, где отображаются те величины, которые выбраны с помощью контроллеров справа (Рис. 14). Также имеются два небольших окна справа от основных дисплеев с измерениями для отображения единиц измерения. Конечно же можно сделать сколько угодно дисплеев, меняя лишь небольшой кусок кода, но для наших целей достаточно лишь два.

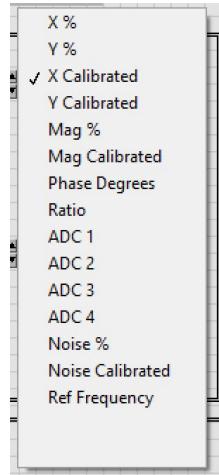


Рис. 14: Выбор измерений Lock-In

Верхняя лампочка Overload загорается при перегрузке прибора.

4.1.4 График измерений относительно времени TIME GRAPH

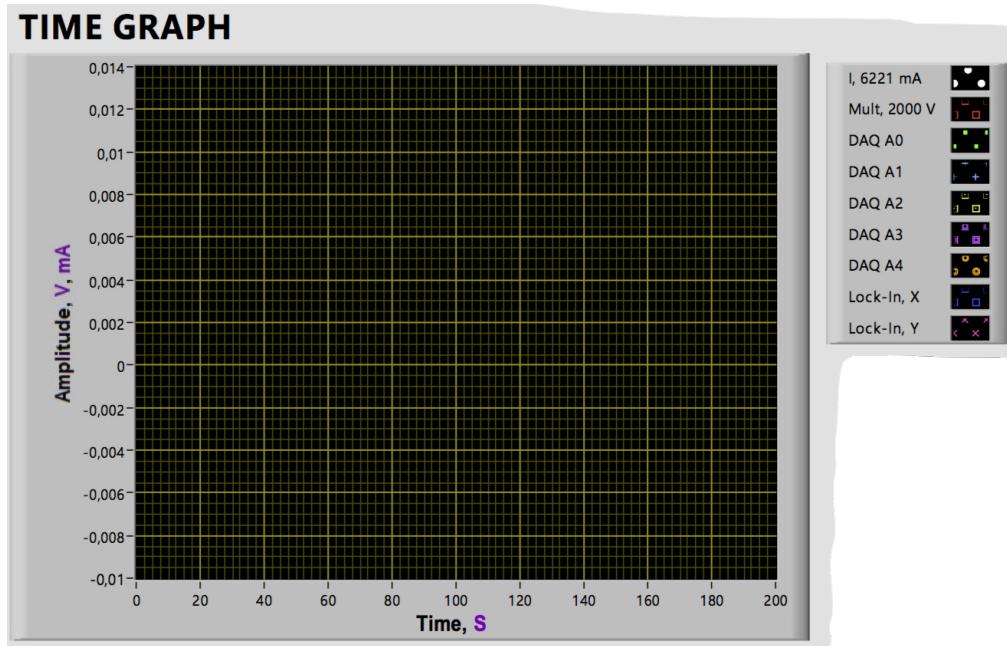


Рис. 15: График относительно времени проведения эксперимента

В этом окне строятся графики тока (I, 6221 mA), показаний мультиметра (Mult, 2000 V), показания 5 каналов DAQ (A0 - A4) и двое каналов Lock-In (X и Y) относительно времени проведения измерений.

Хочется отметить, что не все заявленные каналы DAQ могут быть использованы в эксперименте, поэтому легенда находящаяся справа от графиков, позволяет не отображать те или иные показатели, тем самым не загромождая окно TIME GRAPH.

4.1.5 График показателей приборов относительно других показателей MULTIPLE GRAPHS

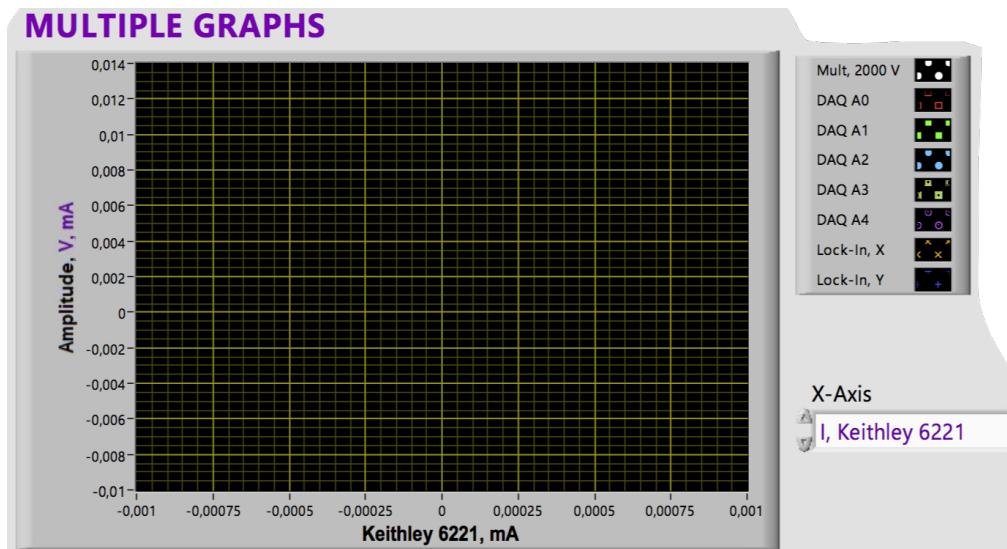


Рис. 16: График измерений относительно других измерений

В данном окне, как уже видно из названия, строятся показания приборов относительно других показаний. Пользователь может сам выбрать ось абсцисс с помощью контроллера справа (X-Axis) Рис. 17. Исходя из выбранной опции меняется и легенда.

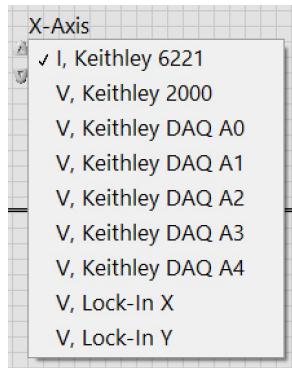


Рис. 17: Выбор оси абсцисс

4.1.6 Масштабы

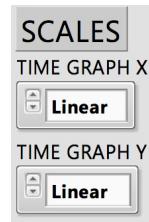


Рис. 18: Масштабы

Для обоих графиков, описанных выше, имеются окошки для выбора масштаба осей (логарифмического или линейного) Рис. 19.

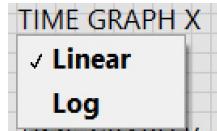


Рис. 19: Выбор между линейным или логарифмическим масштабом оси

4.1.7 Коэффициенты подстраиваемости



Рис. 20: Коэффициенты на которые домножаются показатели приборов

Допустим мультиметр измеряет сигнал порядка одного вольта, а DAQ порядка одного милливольта. Амплитуда этих двух сигналов отличается в тысячу раз, поэтому их линии на обоих графиках будут разделять значительное пустое пространство. Дабы избежать такое неэффективное использование места и лучше видеть изменения сигнала на маленьких амплитудах были придуманы фиктивные коэффициенты на которые домножаются показатели приборов, чтобы быть более детально представленными на графиках.

4.1.8 Индикаторы измерений

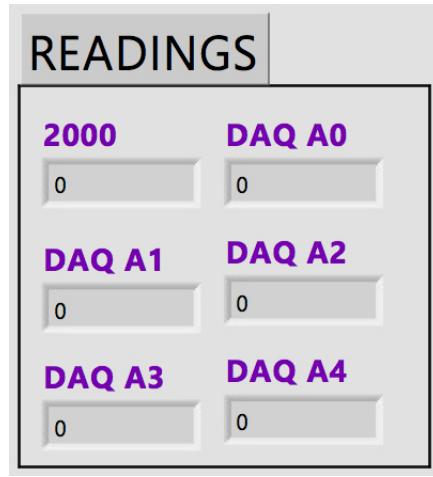


Рис. 21: Показания измерительных приборов

Помимо графического представления результатов имеются индикаторы показаний измерительных приборов (мультиметр и 5 каналов DAQ: A0-A4).

4.1.9 Панель админа

Справа от основной панели с графиками и контроллерами имеется еще панель админа, которая может быть задействована в самых крайних случаях. На ней представлены пару индикаторов и контроллеров для экстренной откладки приборов.

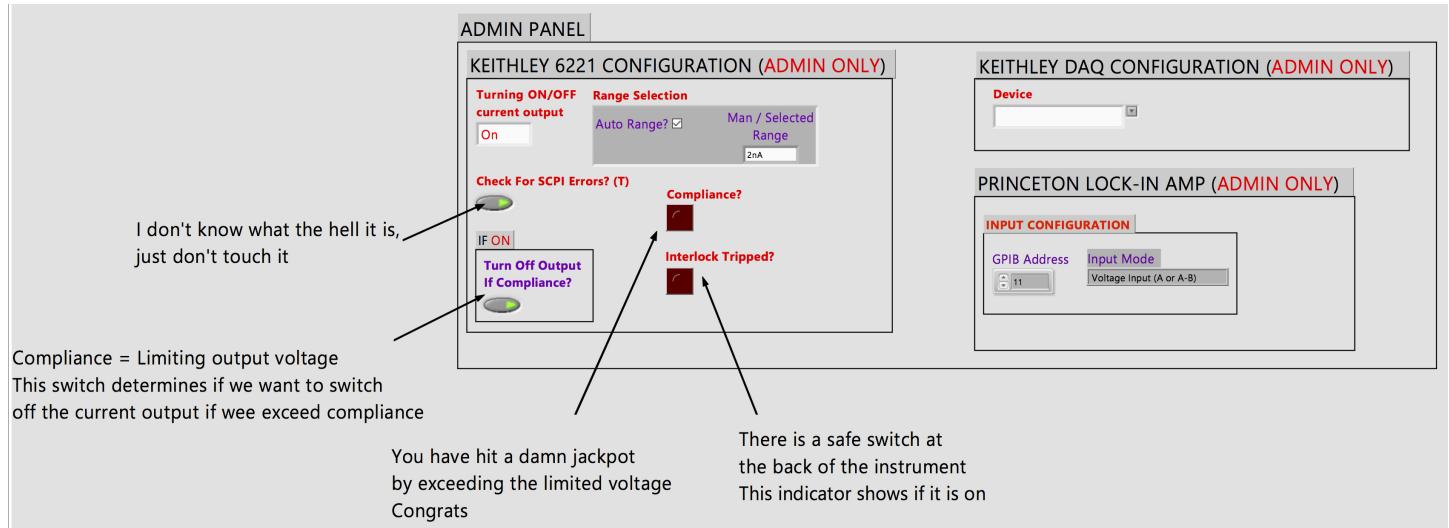


Рис. 22: Панель админа

4.2 Блок-Схема



Рис. 23: Блок-схема алгоритма ПО

Как можно увидеть, блок-схема настолько длинная, что нельзя даже разглядеть отдельные элементы на фотографии, поэтому ее отдельные части будут разобраны ниже по порядку в увеличенном виде.

Основой блок-схемы является Flat Sequence Structure или Кадровая Структура, где в каждом кадре выполняется отдельная часть кода, следующий кадр не наступит, пока не завершится предыдущий. Данный подход позволяет расставить порядок действий, дабы избежать таких ошибок как выполнение каких-либо операций с переменными перед тем, как они были проинициализированы и тд. К сожалению, этот подход не является оптимальным и рекомендуется использовать Simple State Machine или Конченный Автомат, чтобы улучшить производительность, алгоритм и представление кода.

Так как в начале была работа только с мультиметром и источником тока, то Кадровая Структура позволила быстро создать программу для тестирования инструментов. В дальнейшем появилось еще два прибора и больше функций, Кадровая Структура разрасталась, что привело к "колбасе" на Рис. 23. Ни одно линейное преобразование не переведет кадры в Конечный Автомат, поэтому в будущем в качестве улучшения предполагается полная переработка кода для данного подхода и сравнения с Кадровой Структурой для анализа быстродействия и удобства.

4.2.1 Инициализация приборов

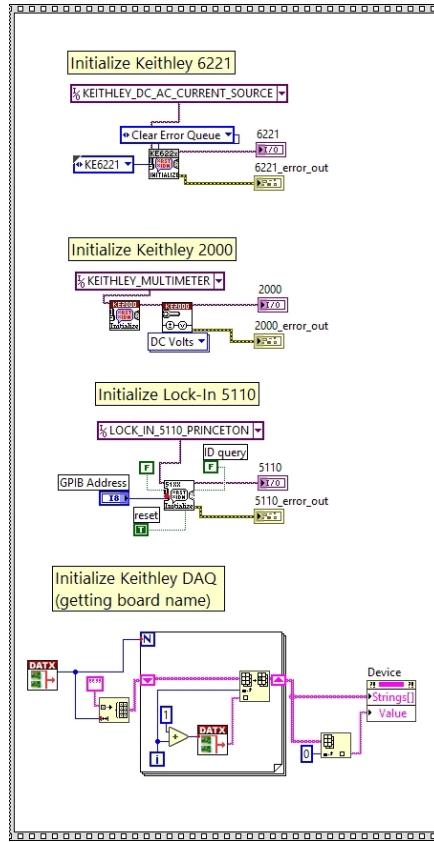


Рис. 24: Кадр инициализации приборов

В данном кадре происходит инициализация источника тока (Keithley 6221), мультиметра (Keithley 2000), фазочувствительного детектора (Lock-In 5110) и DAQ. Для первых трех это не представляет особой сложности. Взяв из библиотеки соответствующие функции, мы с помощью констант типа VISA resource name определяем название прибора, которое мы настроили с помощью MAX. VISA resource name - особый тип, который позволяет передавать информацию о приборе вдоль всей блок-схемы. На выходе функций инициализации источника тока (Keithley 6221), мультиметра (Keithley 2000) и фазочувствительного детектора (Lock-In 5110) мы получаем скрытые индикаторы VISA session и ERROR,

которым впоследствии будут сопоставлены локальные переменные, дабы не тащить проводки через всю блок-схему.

DAQ же настраивается хитрее, потому что MAX его не определяет. Отвечающий за него блок просматривает все порты (степень вложенности subVI для DAQ настолько большая, что даже непонятно какие порты он опрашивает, потому что логичнее всего искать только USB) и составляет массив из имен всех подключенных DAQ. Так как у нас одно такое устройство, то на выходе мы имеем одномерный массив с одним нулевым элементом. Данная информация передается в переменную Device с помощью Property Nodes: Strings[] и Value. Соответствующая переменная будет создана дальше в программе.

Теперь хотелось бы высказать пару замечаний:

1. Для всех приборов имеются библиотеки с примерами, поэтому некоторые куски кода были выдернуты из них, чтобы не изобретать велосипед.
2. Используя Property Nodes, мы можем взаимодействовать с переменными до их изначальной инициализации в Кадровой Структуре.

4.2.2 Определение начального и конечного тока и их проверка

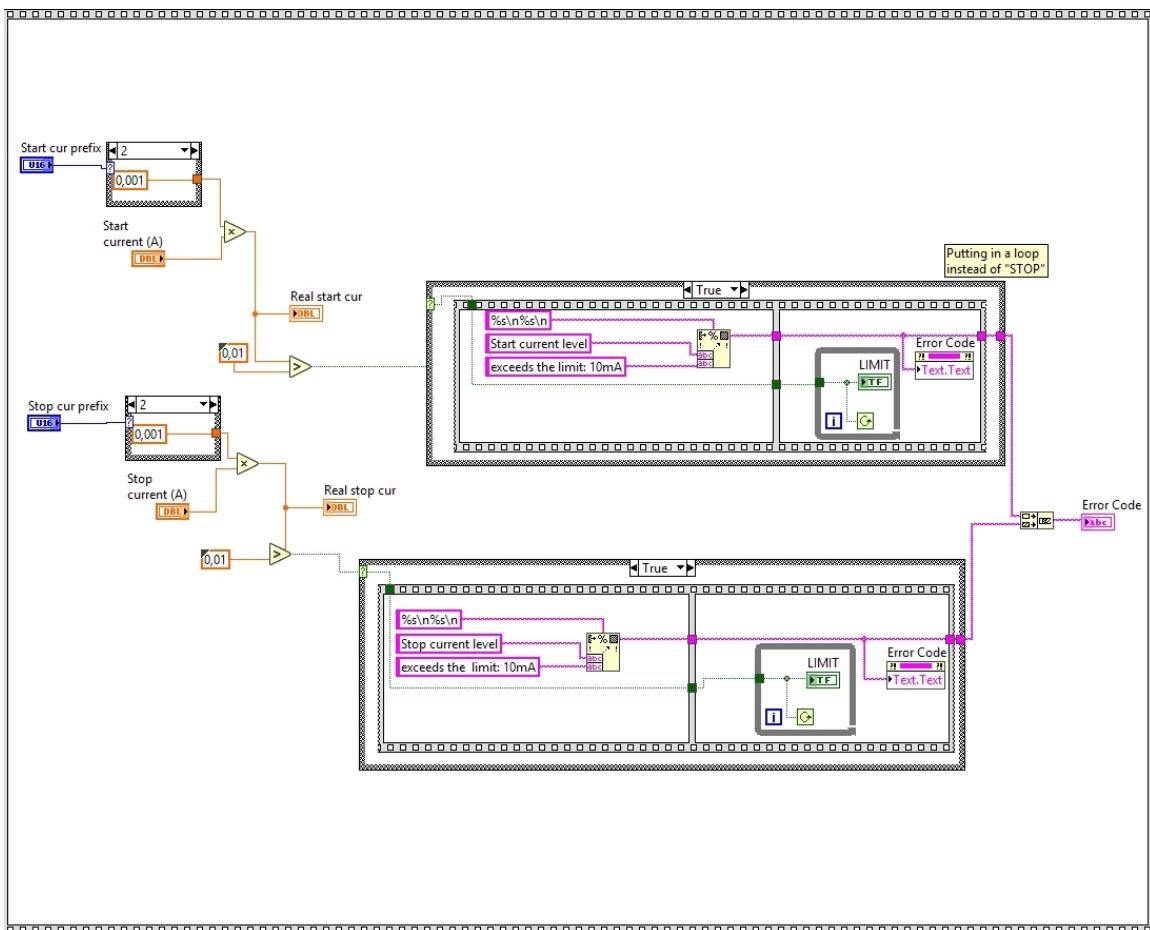


Рис. 25: Кадр проверки и инициализации токов

В данном кадре происходит обработка введенных начального и конечного тока. Начнем рассматривать этот кусок сверху вниз только для начального тока, как алгоритм для конечного аналогичен. Сначала программа определяет выбранную пользователем картинку с физической приставкой с помощью контроллера "Start cur prefix". Каждой картинке ставится в соответствие число, на которое домножается введенное пользователем значение начального тока "Start current (A)" без приставки. На выходе мы получаем скрытую переменную "Real start cur" для дальнейшего использования в блок-схеме.

После этого происходит сравнение с предельным значением 10mA, и результат данного сравнения (булевый True или False) идет на Case Structure или Кейс Структура, где в зависимости от результата происходит либо один сценарий либо другой:

Если результат сравнения True, т.е. введенное значение больше предельного, то сначала происходит создание string потока с сообщением о ошибке, а затем загорается лампочка **LIMIT** и программа переходит в режим зависания с помощью цикла While True до тех пор, пока пользователь не изменит значение. Также поток с сообщением об ошибке выходит из Case Structure, и происходит конкатенация двух сообщений для начального и конечного тока с дальнейшего отображения в **Error code**. Данный сценарий представлен на Рис. 25.

В противном случае просто выводится сообщение о том, что значение тока в порядке и программа продолжит работу.

4.2.3 Расчет шага

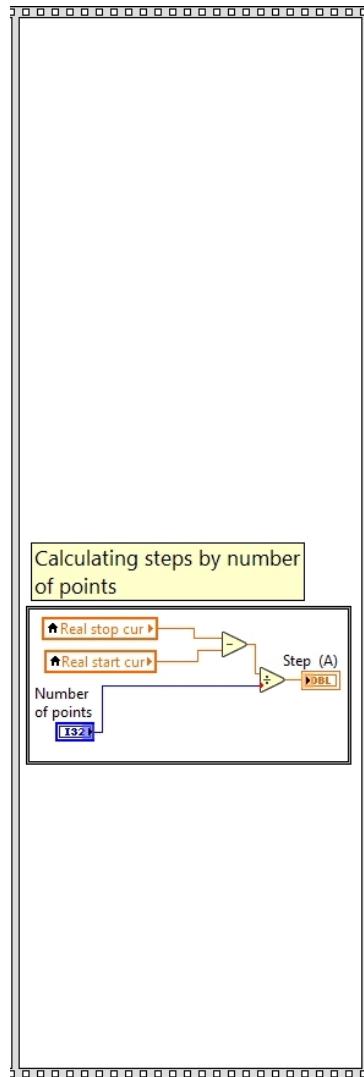


Рис. 26: Кадр расчета инкремента тока

В этом кадре с помощью локальных переменных для созданных раньше "Real stop(start) cur" и введенного пользователем количества точек происходит расчет инкремента для тока, т.е. значение которое будет прибавляться на каждой основной итерации цикла.

4.2.4 Создание папки и файла измерений

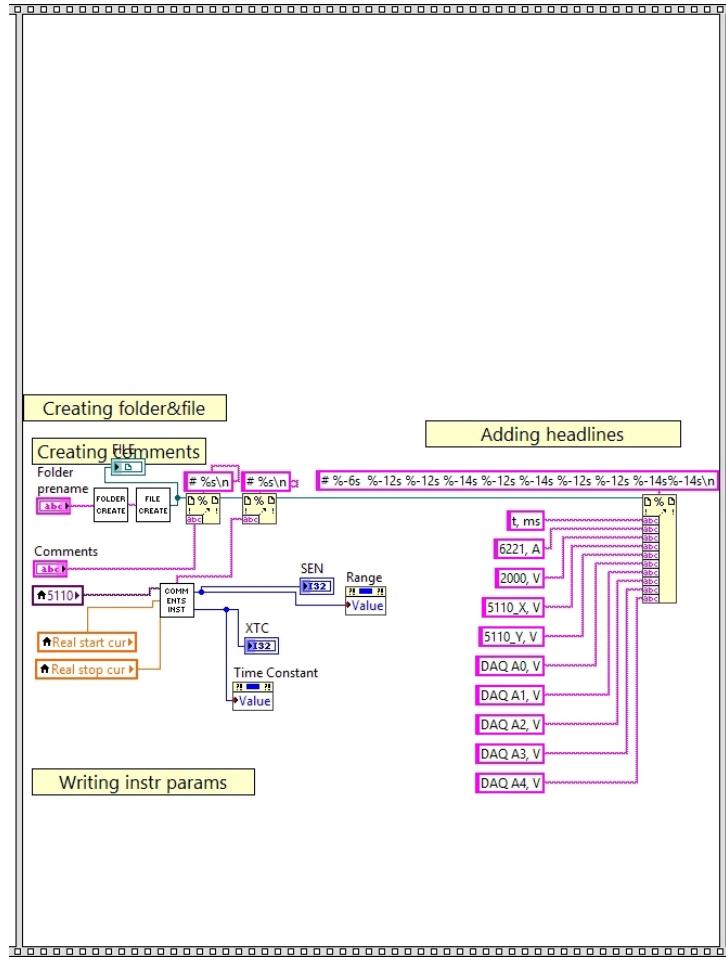


Рис. 27: Кадр создания папки эксперимента и файла для записи измерений

Любой эксперименте подразумевает запись измерений в некоторый файл, которую в свою очередь должен находиться в определенной папке, а не быть брошен где-то на жестком диске. Как же добиться такой организации и как отличать один эксперимент от другого ? Эти два вопроса получили элегантное решение в отдельном кадре нашей блок-схемы, представленного на Рис. 27.

Как уже раньше говорилось, пользователь определяет часть названия папки на фронтальной панели с помощью string control "Folder prename". Из блок-схемы на Рис. 27 видно, что данный string передается в некоторый блок "FOLDER CREATE". Рассмотрим его подробнее.

Данный блок является отдельной подпрограммой или subVI, который и создает папку. Создание subVI сделало код универсальным для использования в любом ПО, а также сэкономило место на блок-схеме.

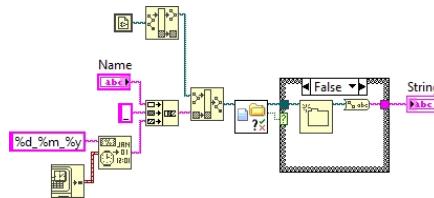


Рис. 28: Блок-схема алгоритма создания папки

Рассмотрим алгоритм создания папки, представленный на Рис. 28, но сначала объясним его принцип. Чтобы избежать повторяющихся названий и непонятных индексов было предложено элегантное решение по созданию папки, основываясь на текущую дату. Если текущая дата была, например, 25 Декабря 2017 года, то папка создавалась

со следующим названием: '10_12_17'. Теперь можно перейти к рассмотрению блок-схемы.

Сначала рассмотрим самую нижнюю функцию. Она выдает нам текущие значения времени и даты и передаёт кластером в следующий блок, где при помощи вспомогательного string, который определяет работу функции, на выходе мы получаем желаемый string вида 'дд_мм_гггг'. Это значение собирается вместе с пользовательским названием, введенным с фронтальной панели и переданным в данную subVI, в один общий string при помощи конкатенации. В результате мы получаем наше название папки вида 'Имя_дд_мм_гггг'.

Теперь с помощью функции определения текущего расположения subVI и определения пути, мы создаем путь к нашей папке и передаем данное значение в блок перед Case Structure.

Есть один небольшой момент, который мы забыли обсудить. А что если уже такая папка создана? Ведь в этом случае повторное её создание приведет либо к появлению к непонятной новой папке, либо к ошибке работы ПО. Именно поэтому была введена Case Structure для двух вариантов действий: если такая папка уже существует и если нет.

Блок перед структурой принимает путь к папке и проверяет есть ли уже такой путь. Если нет, то он выдает булевое значение False, которое передается в Case Structure. После этого в кейсе False создается наша папка с помощью соответствующей функции и полученное значение пути мы конвертируем в string, как значение, которое выдает наша subVI. Если же проверка пути выдает True, значит такая папка уже создана и мы просто конвертируем путь в string без всяких лишних папок.

Теперь вернемся к Рис. 27. 'FOLDER CREATE' передает путь к папке в виде string далее в новую subVI 'FILE CREATE', которая и создает наш файл измерений.

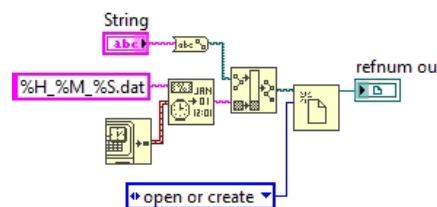


Рис. 29: Блок-схема алгоритма создания файла

Алгоритм создания уникального файла практически такой же как и папки, только мы будем использовать не дату, а время. Как видно из блок-схемы сначала создается путь к файлу с названием вида 'чч_мм_сс', а после этого создается и сам файл. При его создания мы указали 'open or create' дабы избежать ошибок с чтением файлов. По окончанию работы данной subVI мы получаем путь к файлу в виде байтового потока 'refnum out'.

Возвращаясь снов к блок-схеме на Рис. 27, что результат работы последней subVI записывается в скрытый индикатор FILE, который в дальнейшем понадобится нам для записи результатов измерений. После этого мы записываем в файл введенные пользовательские комментарии отдельной строкой. Переходя на следующую строку мы записываем начальные значения инструментов с помощью subVI 'COMMENTS INST'. Разберемся что эта подпрограмма делает и что выдает.

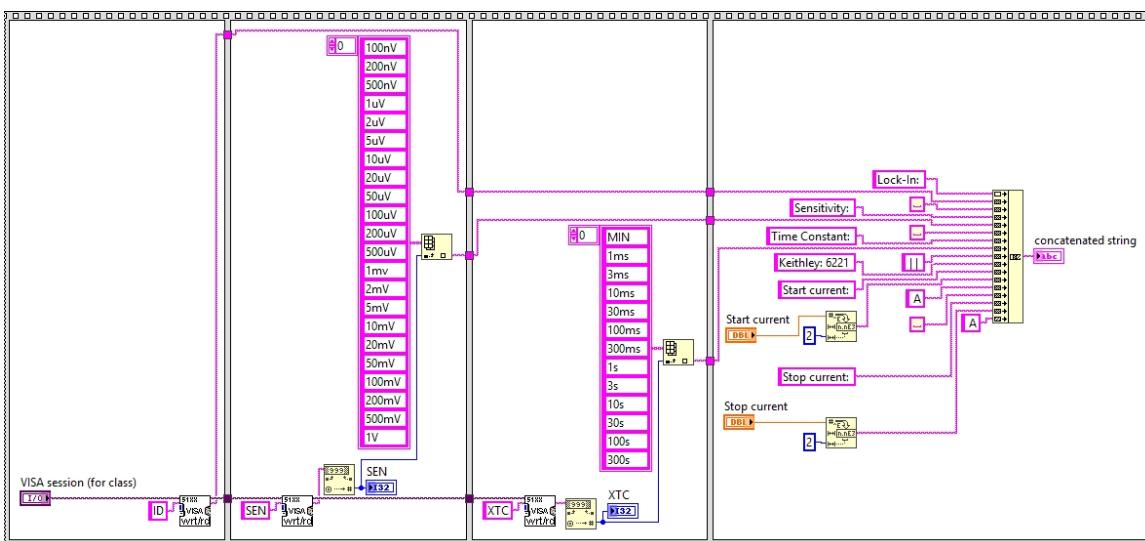


Рис. 30: Блок-схема работы subVI 'COMMENTS INST'

Как уже раньше упоминалось, Lock-In настраивается с фронтальной панели самого прибора, поэтому соответствующие начальные значения должны быть считаны и переданы в отдельные кадры основной блок-схемы ПО. Также было решено записывать эти самые параметры прибора в файл измерений, чтобы знать как он был настроен в самом начале эксперимента. Для этого с помощью приборных команд мы получаем его ID; Sensitivity (чувствительность); Time constant (временная константа). Это представлено на Рис. 30 в отдельных кадрах. При получении значений чувствительности и временной константы блок взаимодействия с прибором при помощи команд выдал нам индексы соответствующих значений в массивах, поэтому пришлось самим создать такие массивы для удобной интерпретации результатов (2ой и 3ий кадры Рис. 30). После этого все собирались в конечный string вместе со значениями начального и конечного токов.

Посмотрим снова на Рис. 27. Наша subVI на входе получает VISA session для Lock-In через локальную переменную 5110, которая является привязанной к основной переменной, полученной при инициализации прибора (см. 4.2.1). Так же она получает локальные переменные начального и конечного тока.

На выходе мы имеем string с начальными настройками интересующих нас приборов, который записывается отдельной строкой в файл с измерениями; отдельные переменные SEN (чувствительность) и XTC (временная константа), которые потребуются нам в дальнейшем для контроля изменений соответствующих параметров Lock-In во время эксперимента. Их же значения передаются с помощью Property node переменным Range (чувствительность) и Time Constant (временная константа), которые используются при работе с Lock-In, но их будет рассказано чуть позже. Последней частью кадра, представленного на Рис. 27 является добавление соответствующих заголовков для результатов эксперимента в файл с измерениями:

1. t,ms - время в миллисекундах
2. 6221, A - ток
3. 2000, V - значение мультиметра
4. 5110_X, V - амплитуда X сигнала Lock-In
5. 5110_Y, V - амплитуда Y сигнала Lock-In
6. DAQ A0, V - значение A0 канала DAQ
7. DAQ A1, V - значение A1 канала DAQ
8. DAQ A2, V - значение A2 канала DAQ
9. DAQ A3, V - значение A3 канала DAQ
10. DAQ A4, V - значение A4 канала DAQ

4.2.5 Временная задержка и настройка параметров

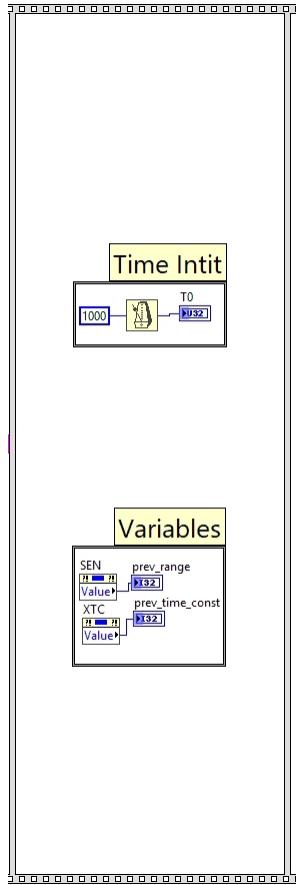


Рис. 31: Кадр временной задержки перед основными измерениями и конфигурация параметров

Перед основными рекомендуется делать небольшую временную задержку, что отображено в верхней части кадра на Рис. 31. Решено было сделать задержку в одну секунду, по истечению которой мы получаем переменную T_0 , которая будет использоваться в основном цикле измерений.

В нижней части кадра значения чувствительности и временной константы, которые мы считали с приборной панели Lock-In, передаются в скрытые индикаторы `prev_range` и `prev_time_const`, которые будут потом использоваться для проверки настройки Lock-In.

4.2.6 Предизмерительные проверки

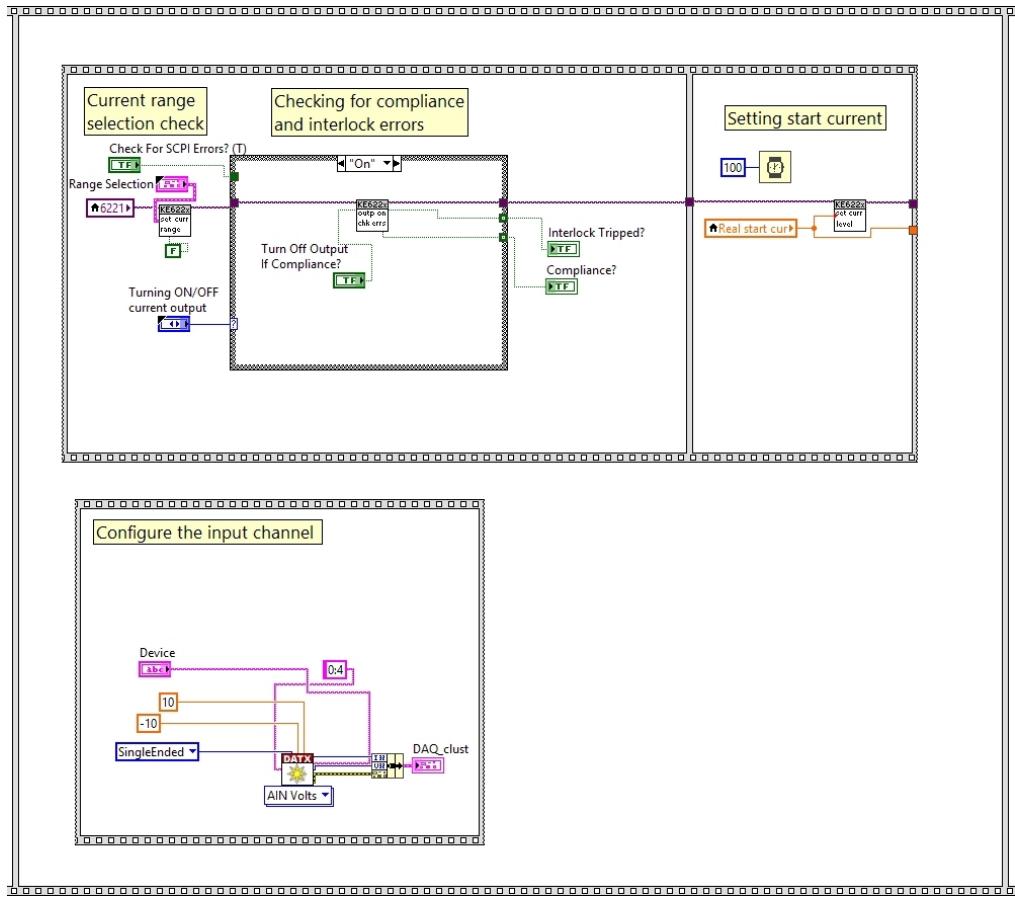


Рис. 32: Кадр проверки источника тока и DAQ

В данном кадре мы проверяем источник тока на некоторые системные ошибки и параметры, чтобы потом не столкнуться с ними во время измерений (превышение допустимого напряжения, снятие предохранителя на задней панели и т.д.). После этих проверок мы с помощью локальной переменной Real_start_current задаем начальный ток нашему источнику.

В нижней же части кадра на Рис. 32 мы в последний раз конфигурируем наш DAQ. С помощью property node мы уже определили переменную Device в самом начале при инициализации данного устройства. Далее мы указываем список каналов, с которыми хотим работать (0:4); диапазон входных сигналов (от -10 вольт до 10 вольт, чем уже диапазон, тем чувствительнее прибор) и тип входного канала (SingleEnded или Differential). На выходе мы получаем отдельно поток ошибки (errout), список каналов (Actual Number of Channels) и Task out, который используется для того, чтобы передать некий код другим блокам библиотеки DAQ о его состоянии и режиме работы. Чтобы не тащить отдельно все три потока, они были объединены в отдельный кластер DAQ_clust, локальные переменные которого будут использоваться в дальнейшем.

4.2.7 Основной измерительный цикл

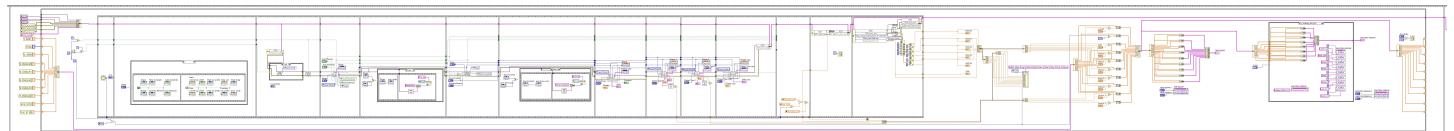


Рис. 33: Кадр с основным циклом измерений, записи всех результатов в файл и отображением их на фронтальной панели

Основной кадр нашей блок-схемы оказался тоже достаточно длинным, поэтому будем рассматривать его по частям, но сперва рассмотрим основной принцип измерений.

Основой всего этого кадра является While цикл, который заканчивает свою работу, когда текущее значение тока не становится равным конечному. Сначала идет сбор информации с приборов, а затем запись в файл и отображение на графиках. Часть блок-схемы будет заключена в Flat Sequence Structure. Теперь рассмотрим основные части.

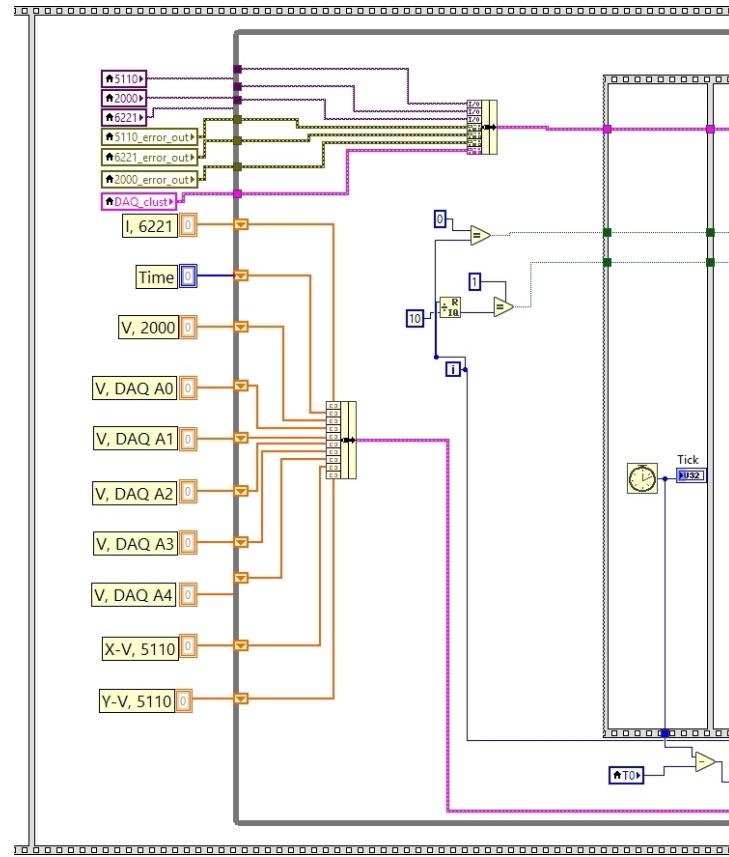


Рис. 34: Shift регистры, VISA session и первая часть цикла

Перед тем как начать разрабатывать цикл, в него были переданы VISA sessions всех наших приборов вместе с потоками об ошибках, что видно в верхней части на Рис. 34. Далее они все были собраны в один кластер, чтобы не тянуть охапку потоков через весь цикл.

То же самое было сделано с shift регистрами измеряемых величин. Shift регистры используются для того, чтобы передавать значения от одной итерации цикла к другой. Инициализация каждого из них происходит с помощью 1D массива, чтобы можно было сохранять значения и строить графики.

Далее происходит проверка на количество произведенных итераций цикла с помощью счетчика 'i'. Если их число равно 10, то программа производит проверку состояния Lock-In. Данная проверка была выдернута из одного из примеров библиотеки данного прибора. Также от индекса идет голубой проводок дальше внутрь цикла, и мы вернемся к нему позже.

Как уже было раньше сказано, часть цикла заключена в Flat Sequence Structure дабы избежать ошибок порядка выполнения действий. В первом кадре стоит секундомер, из которого после вычета времени синхронизации T_0 мы получаем текущее время нашего эксперимента, которое передается дальше в цикл для записи в файл.

Стоит сразу отметить, что сейчас мы разбираем алгоритм, который будет повторяться на каждой итерации цикла.

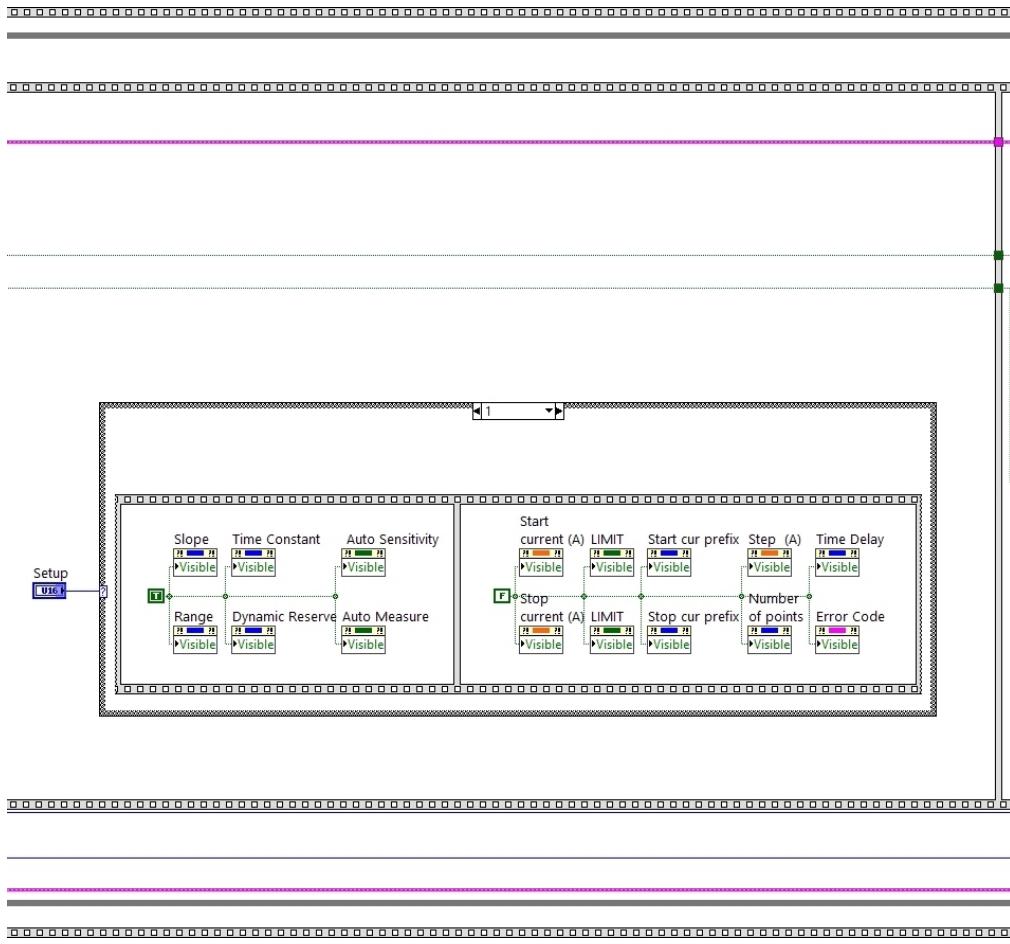


Рис. 35: Реализация алгоритма переключения между начальными настройками ПО и Lock-In

В пункте 4.2.1 мы рассматривали некую 'кашу' из контроллеров и индикаторов и как это все работает. Её реализация представлена на Рис. 35. С помощью контроллера и Case Structure у нас происходит включение видимости (Property node - Visible) одних элементов и отключение видимости других и наоборот. Реализовано это в цикле, так как нам может потребоваться изменить некоторые параметры Lock-In в течение эксперимента.

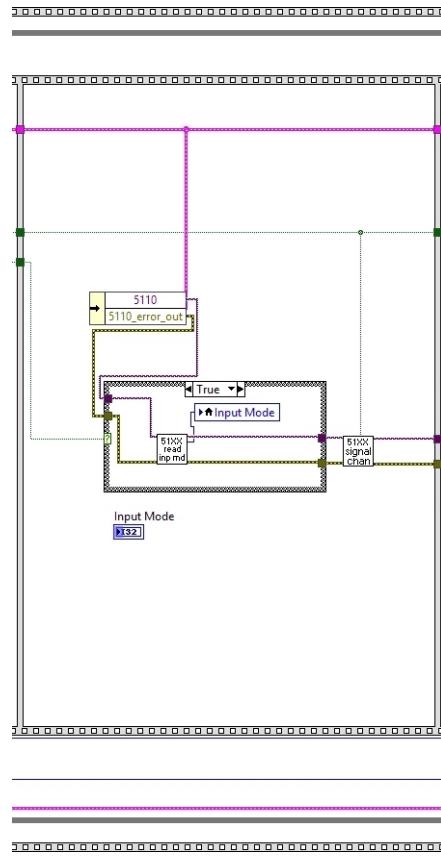


Рис. 36: Кадр основного цикла с конфигурацией входного режима и фильтра Lock-In

В данном кадре происходит конфигурация входного режима (напряжение или ток), а также полосно-заграждающего фильтра Lock-In. Здесь же создается индикатор входного режима (Input Mode), а затем ведется работа только с его локальными переменными.

При разработке программы возник вопрос о отделении данного кадра от основного цикла, чтобы на каждой итерации не переопределять настройки и переменные. Было решено ничего не менять, так как этот кадр в точности повторяет кадр одного из примеров из библиотеки Lock-In и не хотелось бы потом выяснять большее количество ошибок и править их.

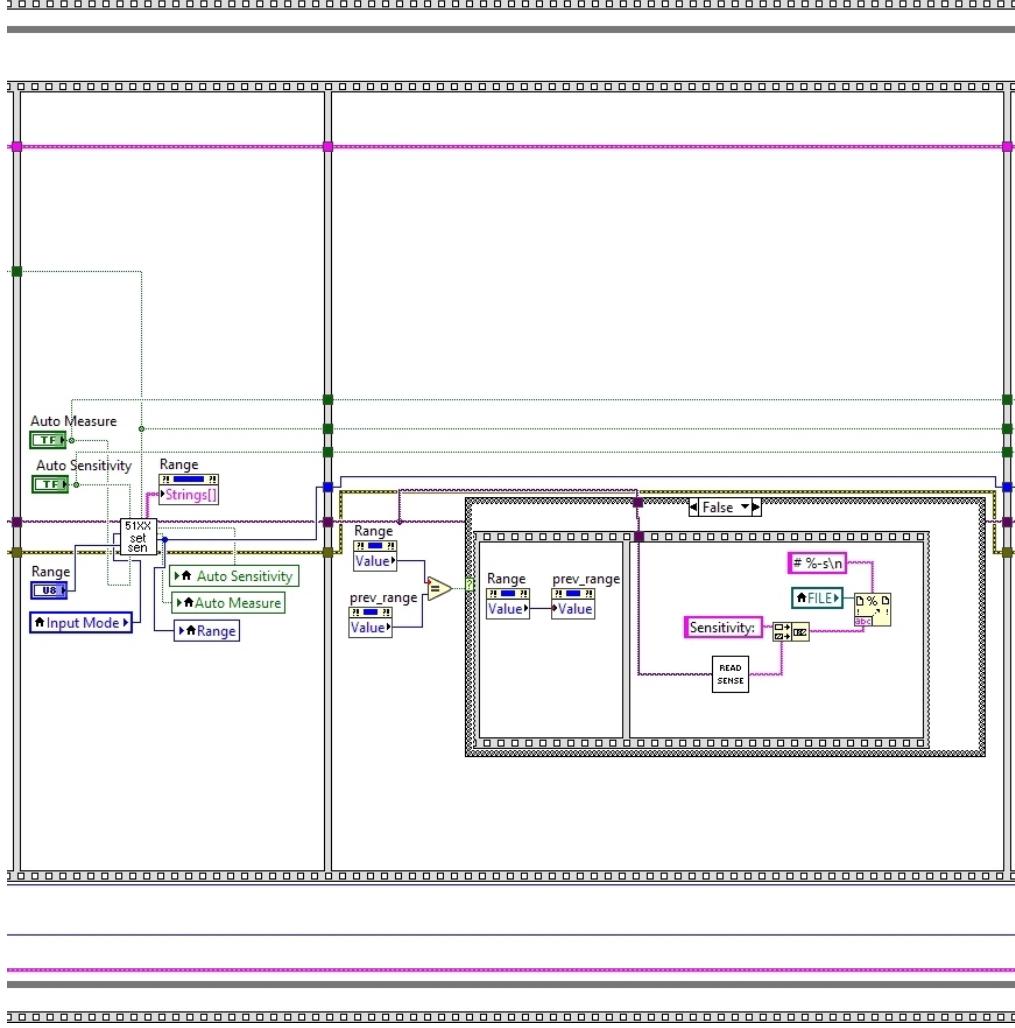


Рис. 37: Кадры настройки временной константы и фильтра низких частот и проверка на изменения

В первом из двух кадров, представленных на Рис. 37, происходит настройка чувствительности, о которой говорилось в пункте 4.1.2. Это значение пользователь вводит с фронтальной панели, и оно передаётся в конфигурационный блок. Также с помощью локальной переменной 'Range' соответствующие настройки отображаются обратно на фронтальной панели ПО.

В этом кадре также представлена обработка кнопок автоматической настройки чувствительности 'Auto Sensitivity' и автоматических измерений 'Auto Measure', но мы их не будем использовать из соображений точности результатов и небольших неприятностей из-за полной автоматизации процесса измерения. Во втором кадре происходит проверка на изменения чувствительности. Если во время эксперимента пришлось изменить данную настройку, это изменения надо обязательно зафиксировать и записать в файл. Для этого с помощью Property nodes мы сравниваем значения двух переменных 'Range' и 'prev_range', о которых говорилось выше. Если они не равны, то есть произошли изменения настройки, то текущее значение 'Range' записывается в 'prev_range'. Затем с помощью subVI, чей код полностью повторяет второй кадр на Рис. 30, получаем новое значение и записываем его в файл отдельной строкой. Результат работы этого кадра основного цикла измерений представлен ниже:

1	00010571	1.08E-3	1.082975	-2.06000E-3	-2.6000E-3	1.0822E+0	3.0518E-4	-3.0518E-4	-6.1035E↔
2	-4	0.000305							
3	# Sensitivity: 500mV								
4	00010962	1.09E-3	1.086971	-4.45000E-3	-6.2000E-3	1.0870E+0	-3.0518E-4	-3.0518E-4	-3.0518E↔
	-4	-0.000305							

Строка '# Sensitivity: 500mV' или аналогичная ей появляется между строками с результатами измерений всякий раз, когда мы изменили настройку чувствительности на фронтальной панели.

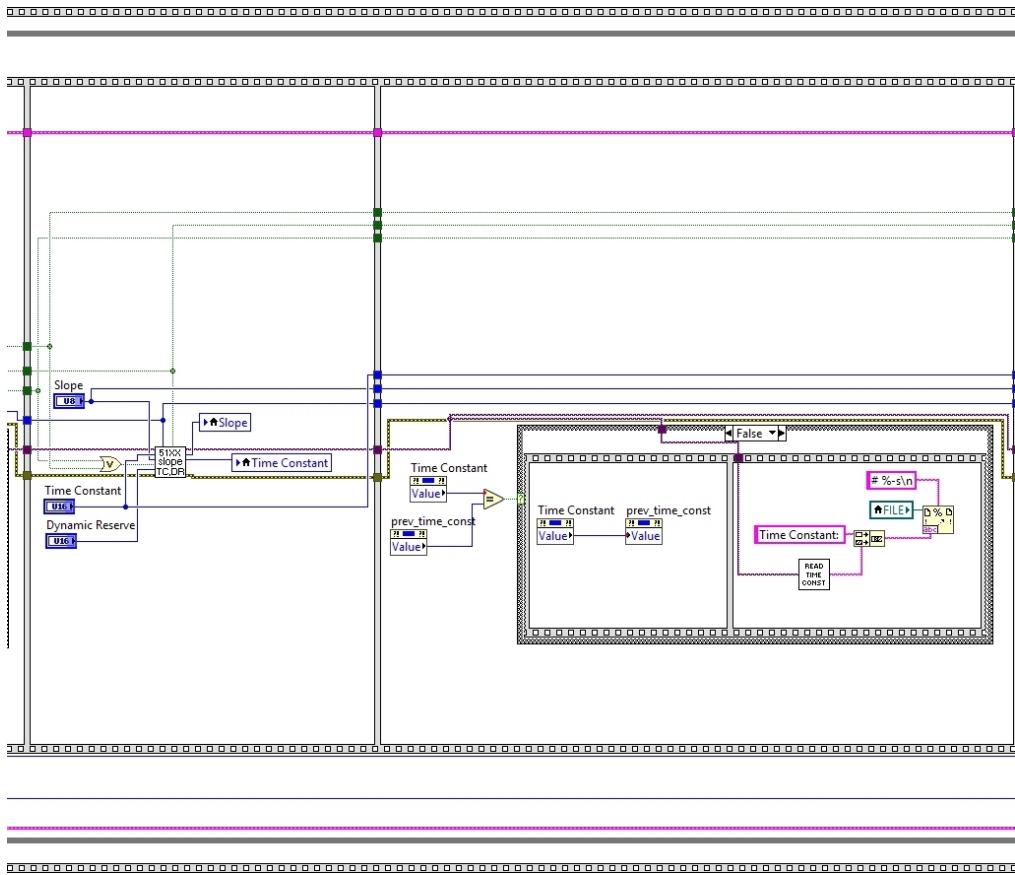


Рис. 38: Кадры настройки временной константы и фильтра низких частот и проверка на изменения

В первом из двух кадров, представленных на Рис. 38, происходит настройка временной константы, фильтра низких частот и динамического резерва сигнала, о которых говорилось в пункте 4.1.2. Все эти значения пользователь вводит с фронтальной панели, и они передаются в конфигурационный блок. Также с помощью локальных переменных 'Slope' и 'Time Constant' соответствующие настройки отображаются обратно на фронтальной панели ПО.

Далее идет проверка на изменение временной константы, которая аналогична представленной на Рис. 37. Код subVI для получения значения Time Constant полностью повторяет третий кадр на Рис. 30. Результат работы данного кадра основного цикла аналогичен результату, представленному выше для чувствительности.

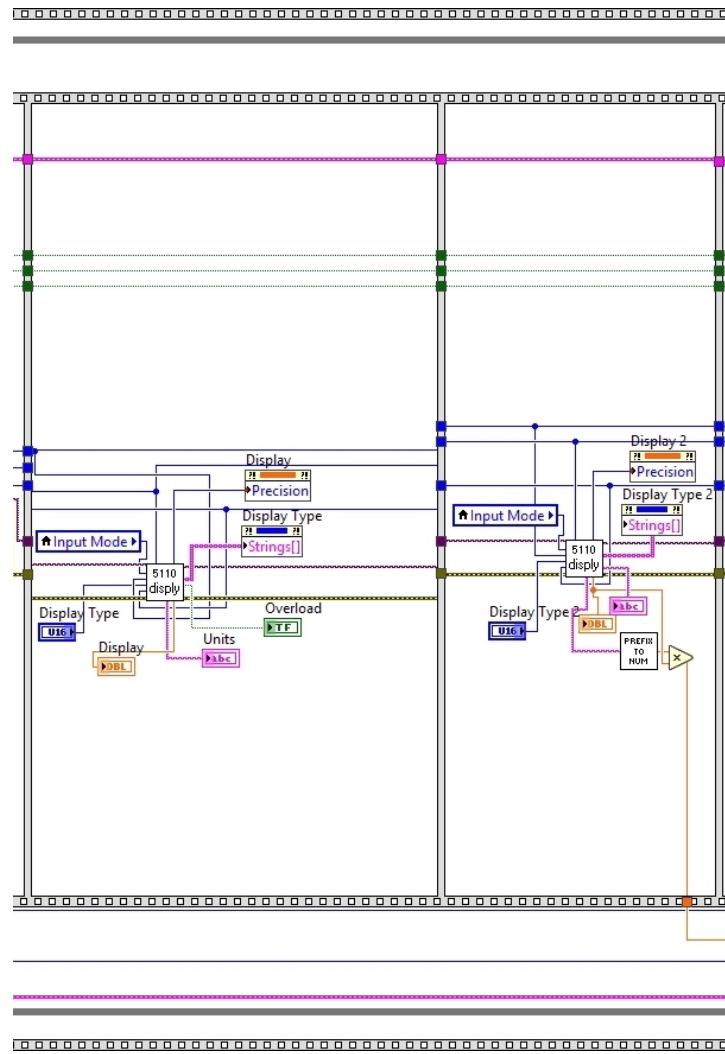


Рис. 39: Кадры настройки отображаемых результатов измерения Lock-In

Как уже говорилось раньше на фронтальной панели ПО как и самого Lock-In имеется два дисплея для отображения результатов. На Рис. 39 представлен алгоритм отображения результатов для первого дисплея. Для второго данный код аналогичен.

В первом кадре мы задаем то, что хотим отображать при помощи контроллера 'Display Type'. На выходе мы получаем само значение измеряемой величины (Property node - String[]), точность измерений (Property node - Precision) и размерность как отдельный индикатор (Units).

Во втором кадре код аналогичен первому, только все индикаторы скрыты и на выходе мы получаем только значение X-сигнала (X Calibrated), потому что именно он будет записываться в файл и отображаться на графиках.

Так как размерность отображается отдельно от значения в виде string, то при помощи subVI 'Prefix to num' мы получаем истинное значение измеряемой величины.

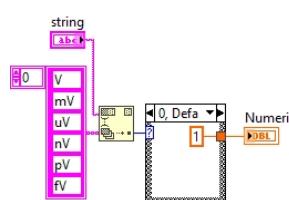


Рис. 40: Алгоритм перевода размерности в число

На Рис. 40 представлен алгоритм той самой subVI. Полученный string с размерностью сравнивается с каждым элементом нашего массива строк, и в зависимости от выбранного элемента с помощью Case Structure выбирается соответствующий коэффициент, на который надо домножить значение X Calibrated, выдаваемого Lock-In.

Следующие кадры нашего основного цикла измерений полностью повторяют описанные выше, только уже для другого дисплея и для Y Calibrated.

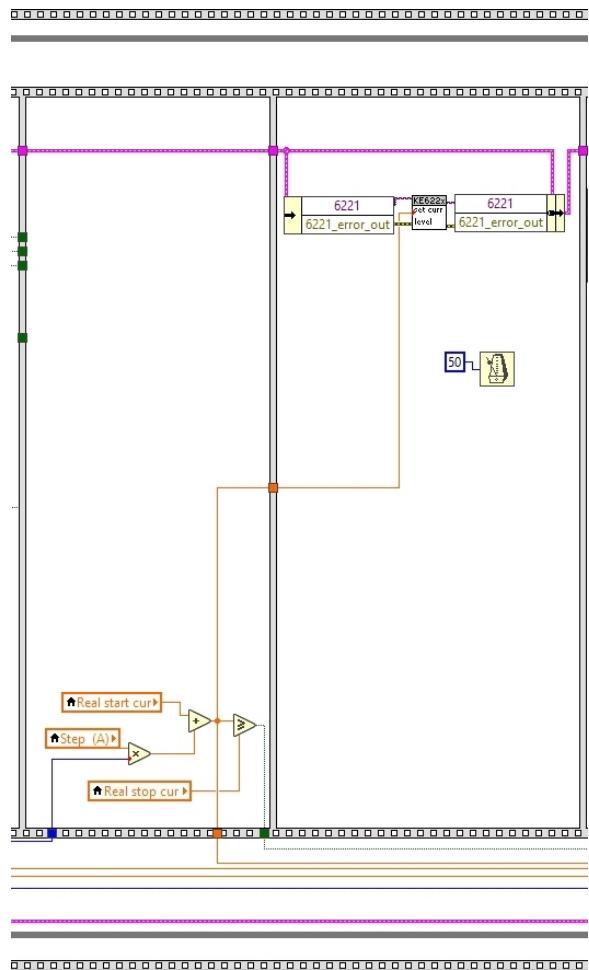


Рис. 41: Кадры с расчетом и установкой значения тока

На двух кадрах, представленных на Рис. 41, происходит расчет нового значения тока и его установка. На первом кадре к текущему значению тока прибавляют шаг и проверяют не равно ли новое значение конченого тока. Если да, то программа заканчивает свою работу через остановку цикла. В противном случае новое значение передается в блок установления тока на Keithley 6221 в следующем кадре. Как можно заметить, кластеры очень удобны в использовании, так как мы не тянем отдельные проводки для ошибки и состояния прибора, а лишь используем функции отвязки по имени (Unbundle by Name) и связки по имени (Bundle by Name), чтобы вынуть или передать соответствующие значения в кластер. Также во втором кадре стоит небольшая задержка на 50 миллисекунд для того, чтобы произошла синхронизация всех приборов и не возникла ошибка в следующем кадре при чтении DAQ и мультиметра.

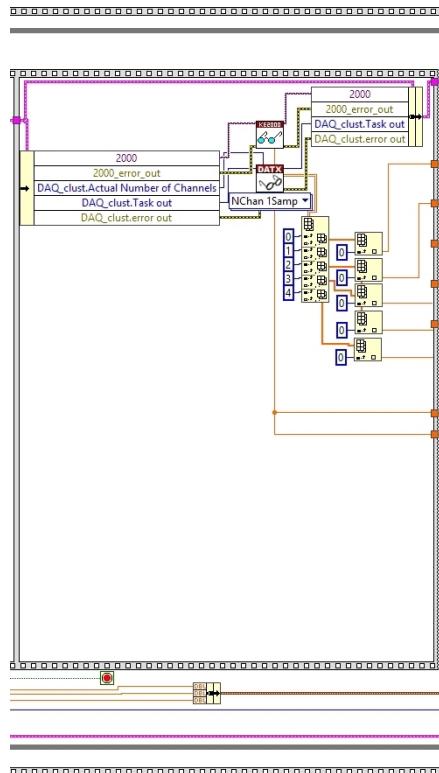


Рис. 42: Получение значений с DAQ и мультиметра

В последнем кадре основного цикла мы получаем значения с 4x каналов DAQ и мультиметра. Стоит обратить внимание, что для DAQ мы используем поднастройку NChan 1Samp, т.е. на каждой итерации с цикла мы получаем лишь одно значение с каждого канала, то что нам как раз и нужно.

Внизу Рис. 42 можно увидеть образование еще одного кластера, который состоит из трех значений: ток, X Calibrated и Y Calibrated.

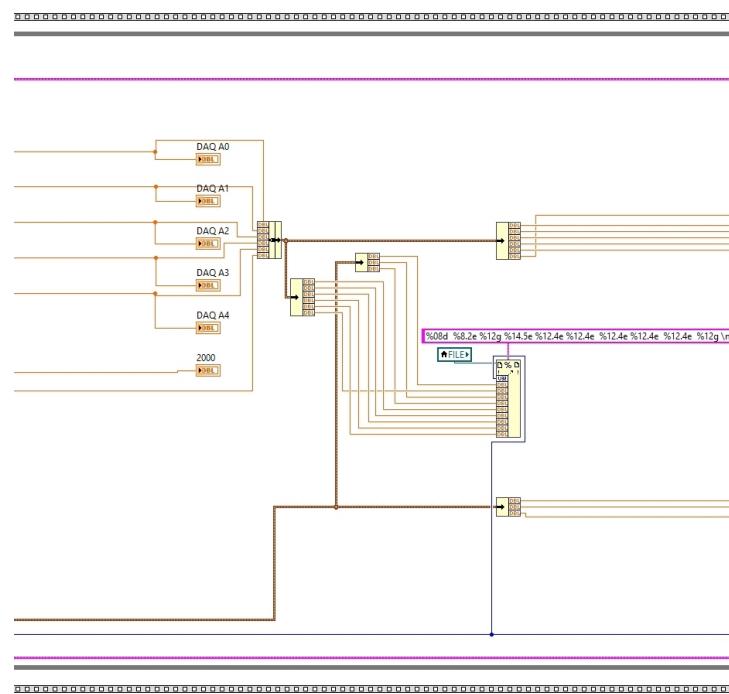


Рис. 43: Запись всех значений в файл

На выходе последнего кадра основного цикла измерений мы получили значения 4x каналов DAQ и мультиметра. Их мы выводим на фронтальной панели при помощи соответствующих индикаторов (см. пункт 4.1.8), а также собираем в кластер, чтобы избавить от множества проводков.

На Рис. 43 также видно, как используя два кластера с измерениями, о которых говорилось раньше, мы записываем значения в файл.

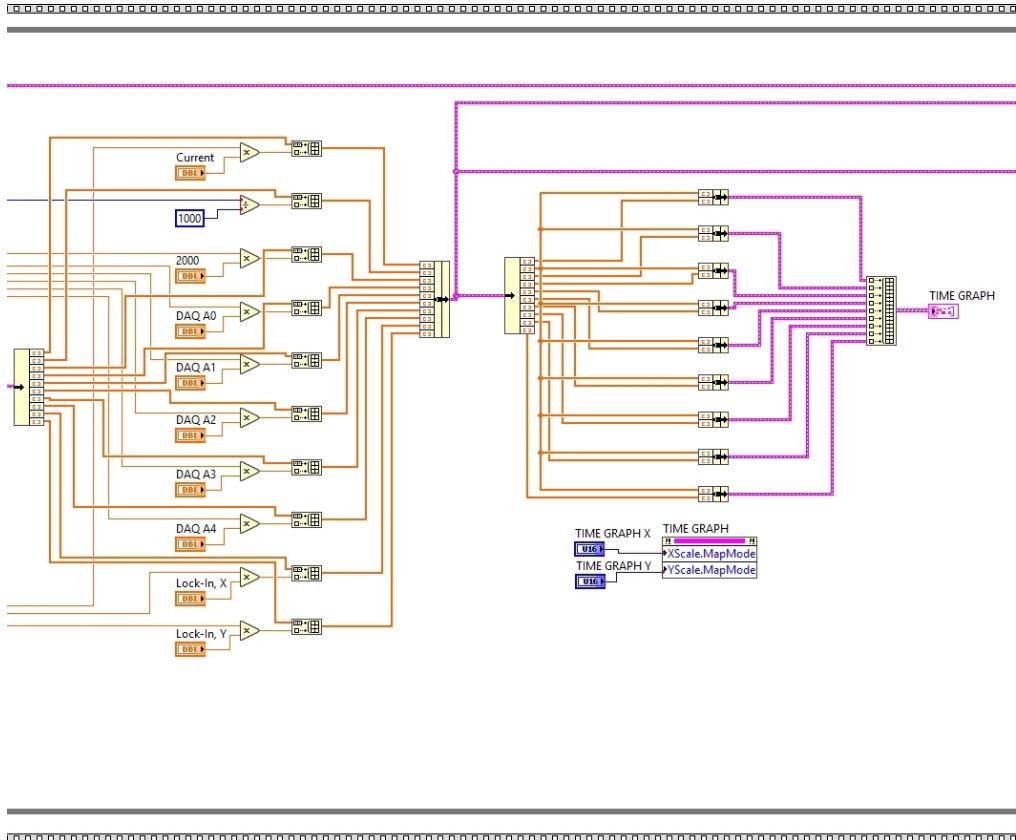


Рис. 44: Алгоритм реализации графика относительно времени эксперимента

На Рис. 44 показано как реализовать временной график по времени. Собрав все значения через кластер shift регистров, мы каждое из них умножаем на коэффициент, о котором говорилось в пункте 4.1.7. После этого создается новый кластер, который будет использоваться для построения обоих графиков, но сначала рассмотрим временной. Создается он следующим образом. Сначала мы собираем массивы значений по двое в кластеры, где на первом месте ставится значение по оси X (всегда массив времени), а на втором значение по оси Y (то, изменение которого мы хотим представить относительно времени). После этого все кластеры собираются в массив и передаются в график. Внизу Рис. 44 имеются контроллеры переключения с линейной шкалой на логарифмическую и обратно для обоих осей.

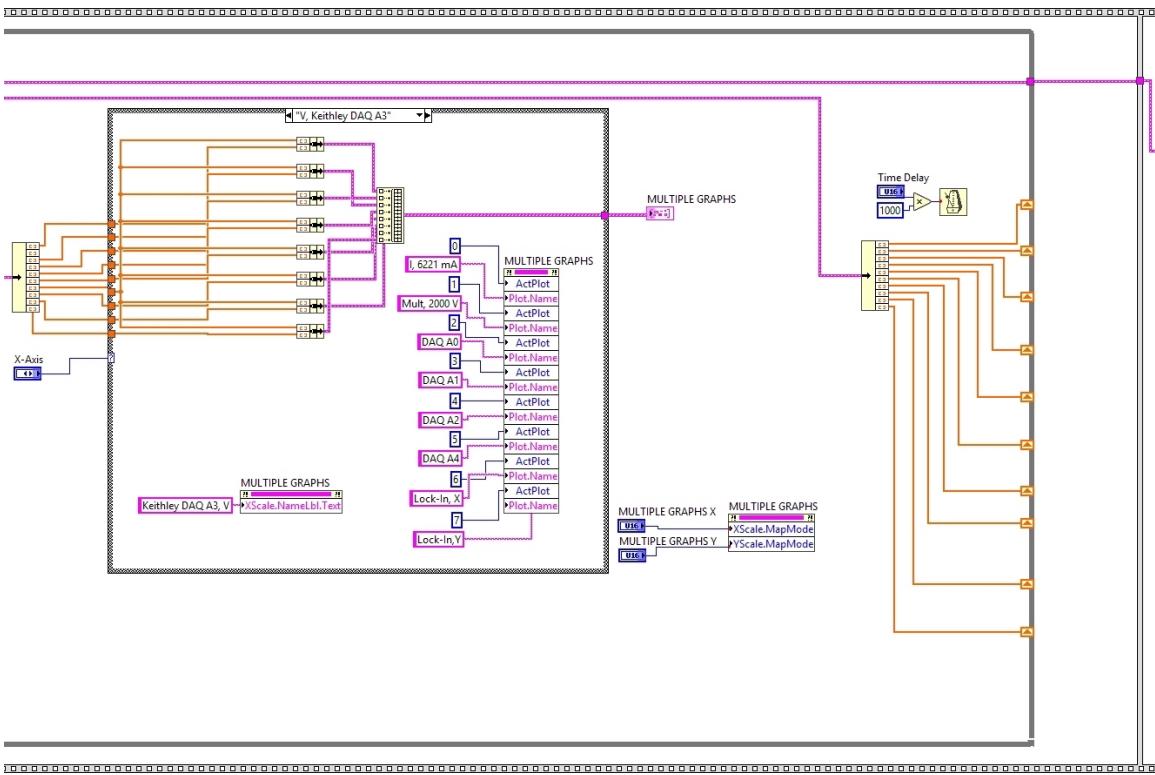


Рис. 45: Алгоритм реализации графика с переменной осью X

На Рис. 45 представлен последний кусок основного цикла измерений, а именно алгоритм построения графика с переменной осью X.

Для его реализации используется уже созданный нами кластер для временного графика, только массивы значений собираются все пары не относительно времени, а уже относительно той величины, имя которой указано в заголовке соответствующего кейса Case Structure. На Рис. 45 приведен пример когда, все величины строятся относительно третьего канала DAQ.

Помимо изменения величины по оси X меняется также её название при помощи Property Node 'XSCale.NameLbl.Text', а также легенда графика.

Для данного графика имеется возможность менять шкалы осей как и для временного.

В самой правой части Рис. 45 shift регистры замыкаются, кластер с ошибками и сессиями выходит из цикла и кадра общей блок-схемы, а также имеется настраиваемая задержка между точками на графике, о которой упоминалось в пункте 4.1.2.

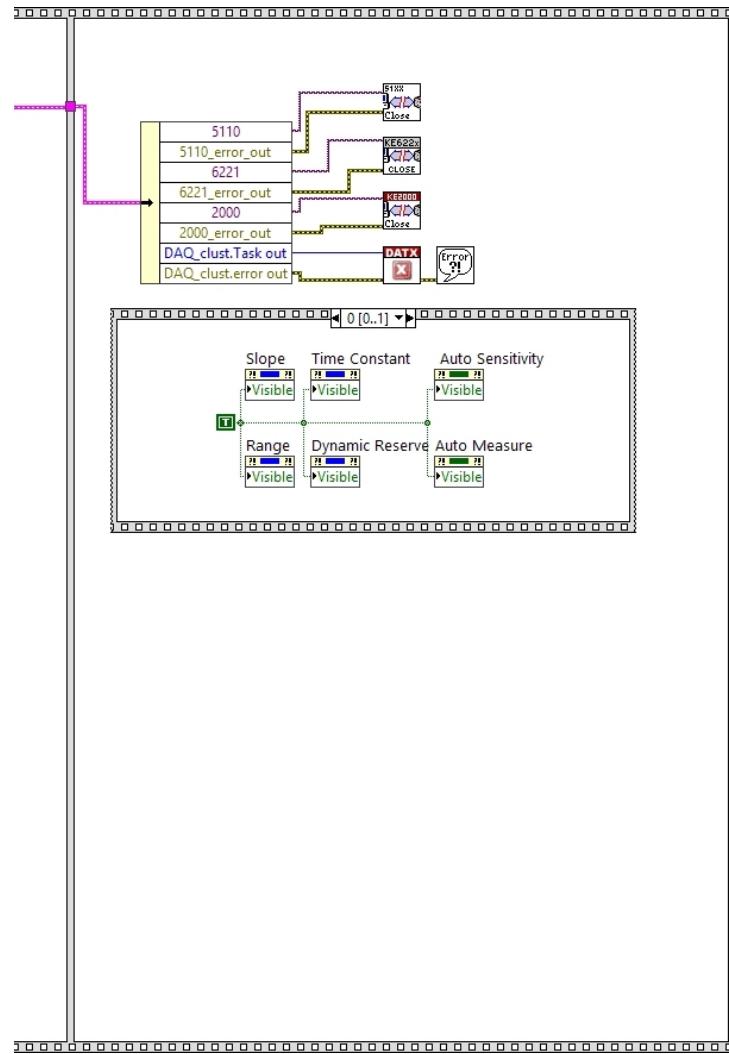


Рис. 46: Закрытие каналов общения с приборами

Это последний кадр нашего ПО, где происходит раскрытие элементов кластера (VISA sessions и errors out), и их передача на соответствующие блоки закрытия сессий с приборами.

Также мы делаем видимыми все контроллеры и индикаторы, описанные в пункте 4.1.2.

На этом наша программа прекращает свою работу. Хочется отметить, что закрытие сессий является очень важным пунктом, иначе система может думать, что прибор еще работает, но он уже давно ничего не измеряет. Из-за такой ошибки при последующих использований ПО может вылезать сообщение о том, что прибор уже используется, но это не так.

5 Список использованных источников

Список литературы

- [1] P. A. Blume **The LabVIEW Style Book** Prentice Hall; 1 edition (March 9, 2007).
- [2] National Instruments Forums
<https://forums.ni.com/t5/Discussion-Forums/ct-p/discussion-forums>
- [3] Мануалы приборов

6 Благодарности

Хочется выразить огромную благодарность моему научному руководителю Арутюнову Константину Юрьевичу за предоставленную возможность поработать над столь интересной и непростой задачей

Хочется поблагодарить Завьялова Виталия Вадимовича за плодотворные идеи по поводуDAQ и самого ПО в целом, а также Халдеева Станислава Ивановича за важные замечания во время разработки программы.