

УДК 004.413:621.317.444

**РАЗРАБОТКА ПРОГРАММНОЙ ЧАСТИ УСТРОЙСТВА
ДЛЯ ИЗМЕРЕНИЯ ПАРАМЕТРОВ ИНДУКТИВНЫХ КОМПОНЕНТОВ МАГНИТОМЕТРОВ**

С.Н. Верзунов

Описываются программные средства, предназначенные для управления устройством определения основного и побочных параметров катушек индукционных магнитометров, отличающихся тем, что они являются активными компонентами и определение их параметров стандартными LCR-метрами не представляется возможным ввиду того, что в них наводится значительная ЭДС индукции. Разработан метод, основанный на определении полного комплексного сопротивления индуктивного компонента магнитометра на различных частотах и подбор определяемых параметров исходя из наилучшего соответствия их наблюдаемым значениям с помощью генетического алгоритма. Программные средства, реализующие этот метод, разработаны на базе фреймворка Kivy и используют разработанную автором библиотеку-драйвер устройства сбора данных ЛКард E502 для языка Python.

Ключевые слова: генетический алгоритм; индуктивный компонент магнитометров; полное комплексное сопротивление; графический интерфейс пользователя; многомерная оптимизация; паразитная емкость; фазовый детектор; драйвер ЛКард; устройство сбора данных.

**МАГНИТОМЕТРЛЕРДИН ИНДУКЦИЯЛЫК КОМПОНЕНТТЕРИНИИ
ПАРАМЕТРЛЕРИН ӨЛЧӨӨ ҮЧҮН ШАЙМАНДЫН
ПРОГРАММАЛЫК БӨЛҮГҮН ИШТЕП ЧЫГУУ**

С.Н. Верзунов

Макалада индукциялык магнитометр катушкаларынын негизги жана кошумча параметрлерин аныктоодо шайманды башкаруу үчүн арналган программалык каражаттар сүрөттөлгөн, алар активдүү компоненттер экендиги менен айырмаланат жана алардын параметрлерин стандарттык LCR өлчөгүчтөрү менен аныктоо мүмкүн эмес, анткени аларда олуттуу индукциялык ЭКК пайда болгон. Ар кандай жыштыктарда магнитометрдин индуктивдүү компонентинин жалпы комплекстүү каршылыгын аныктоого жана генетикалык алгоритмдин жардамы менен алардын байкалган маанилерине эң жакшы дал келгендиктин негизинде аныкталган параметрлерди тандоого негизделген ыкма иштелип чыккан. Бул ыкманы ишке ашырган программалык камсыздоо Kivy фреймворкунун негизинде иштелип чыккан жана автор тарабынан иштелип чыккан Python тили үчүн LCard E502 маалыматтарды топтоо шайманынын китепканасы колдонулган.

Түйүндүү сөздөр: генетикалык алгоритм; магнитометрлердин индуктивдүү компоненти; толук комплекстүү каршылык көрсөтүү; колдонуучунун графикалык интерфейси; көп өлчөмдүү оптимизация; мите сыйымдуулук; фаза детектору; LCard драйвери; маалыматтарды чогултуучу шайман.

**DEVELOPMENT OF THE SOFTWARE PART OF THE DEVICE
FOR MEASURING THE PARAMETERS OF INDUCTIVE COMPONENTS OF MAGNETOMETERS**

S.N. Verzunov

The paper describes software designed to control the device for determining the main and secondary parameters of the coils of induction magnetometers, which differ in that they are active components and the determination of their parameters by standard LCR meters is not possible due to the fact that a significant induction EMF is induced in them. In connection with this, a method was developed based on the determination of the total complex resistance of the inductive component of the magnetometer at various frequencies and the selection of the determined parameters based on the best correspondence to their observed values using a genetic algorithm. The software that implements

this method is developed on the basis of the Kivy framework and uses the LCard E502 data acquisition device driver library for the Python language developed by the author.

Keywords: genetic algorithm; inductive component; inductive component of magnetometers; total impedance; graphical user interface; multivariate optimization; parasitic capacitance; phase detector; LCard driver; data acquisition device.

Индукционные компоненты магнитометров отличаются от типовых индуктивностей тем, что содержат, как правило, сотни тысяч витков, в связи с чем обладают достаточно большой индуктивностью, паразитной емкостью и активным сопротивлением, выходящим за пределы измерений серийных приборов. Кроме того, индуктивный компонент магнитометра с большим числом витков является активным элементом, в котором генерируется значительная ЭДС индукции, оказывающая сильное влияние на работу серийных LCR-метров любого типа. Тем не менее, задача определения параметров индуктивных элементов часто возникает при создании, отладке и ремонте магнитометров. Поэтому необходимо устройство, способное работать в широких диапазонах измерения L , C , R , несмотря на присутствие ЭДС в измеряемом элементе [1, 2].

Устройство для реализации способа преобразования полного сопротивления в напряжение, как правило, состоит из таких основных узлов как генератор, АЦП, фазовый детектор и МПС. Все эти узлы могут быть заменены устройством сбора данных E502 производства «ЛКард». Устройство сбора данных ЛКард E502, представляет собой универсальный 16-ти битный модуль ввода/вывода до 32-х аналоговых и 17-ти цифровых сигналов в компьютер через интерфейсы USB 2.0 (high-speed) и Ethernet (100 Мбит) с частотой преобразования до 2 МГц и возможностью их цифровой обработки в реальном времени. Кроме того, ООО «ЛКард» приветствует интеграцию своих модулей в пользовательские системы [3].

Структурная схема предлагаемого прибора показана на рисунке 1.

Так как амплитуда напряжения ЦАП задается программно и АЦП имеет программируемые пределы измерения, то формула для определения полного комплексного сопротивления имеет вид:

$$Z_x = R_0 (U_{zx} - U_{R0}) / U_{R0}. \quad (1)$$

Напряжение U_{zx} генерируется ЦАП устройства в режиме синхронного циклического вывода с аппаратно заданной частотой дискретизации, равной 1 МГц; амплитуда, частота и фаза при этом известны, т. к. задаются программно, поэтому:

$$U_{zx} = A_0 \sin(2\pi\nu t_0 + \phi_0),$$

где A_0 – амплитуда; ν – частота; ϕ_0 – фаза U_{zx} ; $t_0 = 0, \Delta\sigma, 2\Delta\sigma, 3\Delta\sigma$.

Определение параметров индукционного компонента магнитометра с помощью генетического алгоритма. Описанный в работе [4] способ определения параметров индукционного компонента магнитометра на основе алгоритма Бройдена–Флетчера–Гольдфарба–Шанно и данных о первых производных оптимизируемой функции, обладает одним существенным недостатком. В связи с тем,

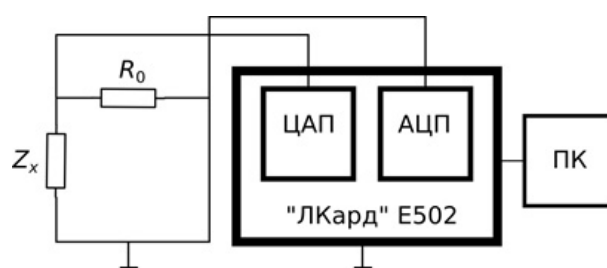


Рисунок 1 – Реализация измерителя параметров индукционных компонентов магнитометров на базе платы сбора данных ЛКард E502

что функция нелинейная, возникает проблема локальных минимумов, когда за верное решение алгоритмом принимается не глобальный, а локальный минимум.

Для определения параметров индукционного компонента магнитометров предлагается использовать генетический алгоритм. Генетический алгоритм оперирует набором особей (так называемой популяцией). Они представляют собой записи (хромосомы), кодирующие одно из решений оптимизационной задачи. Этим генетический алгоритм отличается от алгоритма Бройдена–Флетчера–Гольдфарба–Шанно и других подобных алгоритмов оптимизации, которые работают только с одним решением, улучшая его.

В рассматриваемом случае, в терминах практики применения генетических алгоритмов каждая особь кодируется одной хромосомой, содержащей четыре вещественных параметра: $x = \{L, C, R_c, R_f\}$, а функция потерь имеет вид:

$$W(x) = \frac{1}{2m} \sum_{s=1}^m |Z_{x,s} - \hat{Z}_{x,s}|. \quad (2)$$

Задача заключается в том, что в пространстве поиска X требуется найти

$$x^* = \arg \min_{x \in X} W(x),$$

где \hat{Z}_x вычисляется по формуле (2).

Используемый генетический алгоритм детально выглядит следующим образом:

1. Случайным образом генерируется конечный набор пробных решений: $p^1 = \{p_1^1 \dots p_n^1\}$, $p_i^1 \in X$ (первое поколение, $n = 200$ – размер популяции).

2. Выполняется оценка приспособленности текущего поколения: $F^k = \{f_1^k \dots f_n^k\}$, $f_i^k = W(p_i^k)$. При этом выполняется нормировка к виду: $F^{k+1} = \{f_1^{k+1} \dots f_n^{k+1}\}$, $f_i^{k+1} = (f_i^k - f_0) / (f_1 - f_0)$, где f_1 и f_0 соответственно, лучший и худший показатель в текущей популяции.

3. Выход, если выполняется критерий останова, иначе переход к следующему пункту.

4. Генерация нового поколения посредством операторов селекции S , скрещивания C и мутации M : $p^{k+1} = M \cdot C \cdot S(p^k, F^k)$ и переход к пункту 2.

В процессе селекции отбирают только небольшое количество, а именно 10 % лучших пробных решений, другие же далее не используются, при этом особи с одинаковым набором генов отбрасываются. Скрещивание выполняется над половиной лучших особей, попарно. Результатом являются также две особи с компонентами, взятыми от родителей: $p_i^k = \{a_1, a_2, \dots, a_n\}$, $p_j^k = \{b_1, b_2, \dots, b_n\} \in P^k$ в k -ой популяции являются два элемента популяции $k+1$, такие что $p_i^{k+1} = \{a_1, \dots, a_c, b_{c+1}, b_n\}$, $p_j^{k+1} = \{b_1, \dots, b_c, a_{c+1}, a_n\} \in P^{k+1}$, где точка c выбирается случайно. Оператор мутации изменяет 20 % особей в популяции, изменяя значения кодируемых в хромосомах признаков на некоторую небольшую случайную величину. В качестве критерия останова используется достижение максимального числа поколений, отпущенных на эволюцию [5].

На рисунке 2 показан процесс минимизации функции потерь (2) с помощью генетического алгоритма в виде зависимости значения функции потерь (Objective function) от номера итерации (Iteration count) на том же примере, ранее использовавшемся для демонстрации применения алгоритма Бройдена–Флетчера–Гольдфарба–Шанно в предыдущем пункте.

Значение средней ошибки, достигнутой с использованием предложенных здесь гиперпараметров генетического алгоритма, таких как размер популяции, вероятность мутации, количество отбираемых для следующего поколения особей, удалось вычислить по формуле (2), оно равно 3.2 Ом, что на порядок меньше, чем было получено с помощью алгоритма Бройдена–Флетчера–Гольдфарба–Шанно.

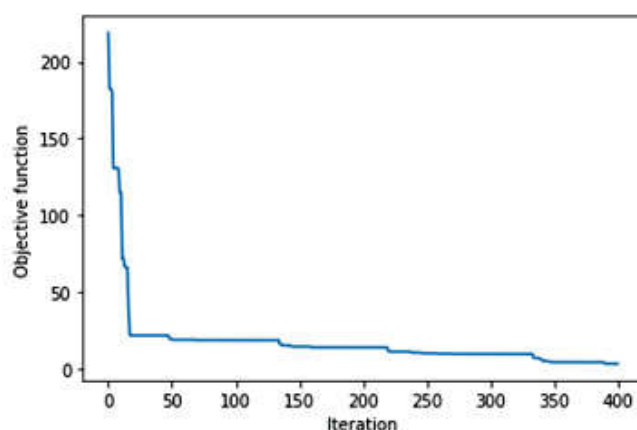


Рисунок 2 – Зависимость фазы полного сопротивления от частоты (50–200 Гц).

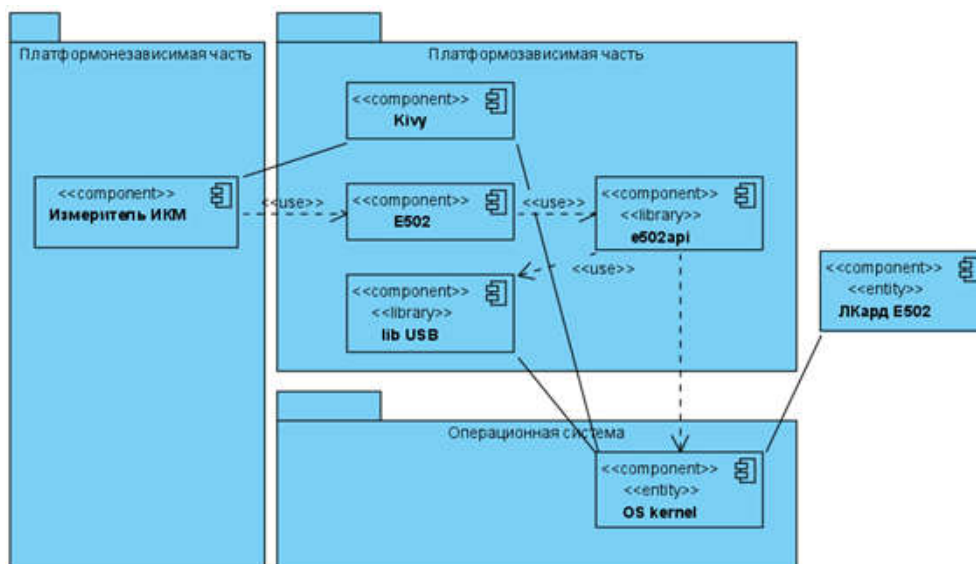


Рисунок 3 – Диаграмма компонентов программных средств для измерения параметров индуктивного компонента магнитометров

Программные средства для измерения параметров индуктивного компонента магнитометров. Приведен пример практической реализации архитектуры программных средств для измерения индуктивного компонента магнитометров на базе устройства сбора данных ЛКард E502, выбор оптимальных методов и инструментальных средств создания программного обеспечения, позволяющих с минимальными усилиями, практически без изменения исходного кода использовать разработанные программные средства на вычислительных платформах с разнообразным аппаратным окружением, работающим под управлением различных операционных систем (рисунок 3).

Архитектура программных средств. Графический интерфейс пользователя, драйвер устройств и некоторые другие части программных средств тесно связаны с оборудованием, поэтому должны быть соответствующим образом адаптированы для обеспечения возможности его работы на различных операционных системах: Windows, Linux, Android. Схематически архитектура программных средств, способных работать на различных операционных системах, выглядит так, как показано на

рисунке 3. Для использования на другой платформе необходимо переработать показанную на рисунке платформозависимую часть. Остальной же код программных средств, написанный на языке Python, можно будет запускать на любой платформе без изменений.

Сетевой стек TCP/IP, необходимый для связи с устройством сбора данных с помощью Ethernet или Wi-fi, в настоящее время имеется в любой современной мобильной и настольной операционной системе, однако не все функции поддерживаются одинаково хорошо. Для того чтобы обеспечить работу на различных платформах в драйвере ЛКард E502, необходимо отключить поддержку автоматического поиска устройств в локальной сети, недоступную в настоящее время в операционной системе Android, изменив в файле *CMakeLists.txt* библиотеки *e502api* опцию

```
option (E502API_ENABLE_DNSSD "enable dns-sd service discovery" OFF),
```

а в настройках программных средств, в случае подключения устройства сбора данных по сети, необходимо указать IP адрес используемого устройства.

В программных средствах был использован стандартный драйвер USB, который в настоящее время поддерживается всеми популярными операционными системами. В главный CMake-файл драйвера платы сбора данных для поддержки Android устройств следует добавить код:

```
if(${CMAKE_SYSTEM_NAME} MATCHES Android)
  set(CMAKE_SYSTEM_VERSION 21) # уровень API
  set(CMAKE_ANDROID_ARCH_ABI armeabi)
  set(CMAKE_ANDROID_STL_TYPE gnustl_static)
endif(),
```

обеспечивающий корректную сборку для устройств на базе операционной системы Android. Как видно из приведенного выше кода, гарантируется работа на устройствах с API 21 и выше, что соответствует версии Android не менее чем 5.0. Работа на более старых устройствах, к сожалению невозможна, т. к. для них отсутствует поддержка стандарта POSIX Threads, необходимого для работы драйвера ЛКАРД E502, реализующего поддержку многопоточных программ, преимущественно ориентированных на исполнение на системах с общей памятью (Symmetric Multiprocessing, сокращённо SMP). Как известно, это такие системы, где установлено несколько процессоров или/и многоядерные процессоры, и каждое ядро имеет доступ ко всей оперативной памяти компьютера.

Кроме того, в файл */lib/osspec/osspec.c* драйвера платы сбора данных необходимо внести исправление для компиляции драйвера платы сбора данных для операционной системы Android:

```
if (0) { /*timeout != OSSPEC_TIMEOUT_INFINITY*/
  struct timespec timeToWait;
  f_get_abs_time(timeout, &timeToWait);
  /*wt_res = pthread_timedjoin_np(thread, NULL, &timeToWait);*/
} else {
  wt_res = pthread_join(thread, NULL),
```

заменив тем самым неподдерживаемую в Android функцию стандарта POSIX Threads *pthread_timedjoin_np* на поддерживаемую во всех операционных системах функцию *pthread_join*.

Библиотека для работы с устройством ЛКард E502. На рисунке 4 показана архитектура библиотеки для работы с устройством ЛКард E502.

Для начала работы с модулем необходимо установить с ним связь с помощью функции *open_usb*. Для идентификации устройств используются их серийные номера. Получить список серийных номеров всех подключенных устройств E502 можно с помощью *get_usb_serial_list*. Данная функция возвращает список, в котором сохранены найденные серийные номера, а принимает максимальное количество присоединенных модулей (по умолчанию это значение равно 16). Следует отметить, что с одним модулем одновременно может быть установлено только одно соединение. При попытке открыть модуль, с которым уже установлено соединение через другой описатель *_hnd* (возможно, в другой программе), *open_usb* вернет ошибку. При этом *get_usb_serial_list* по умолчанию возвращает список всех

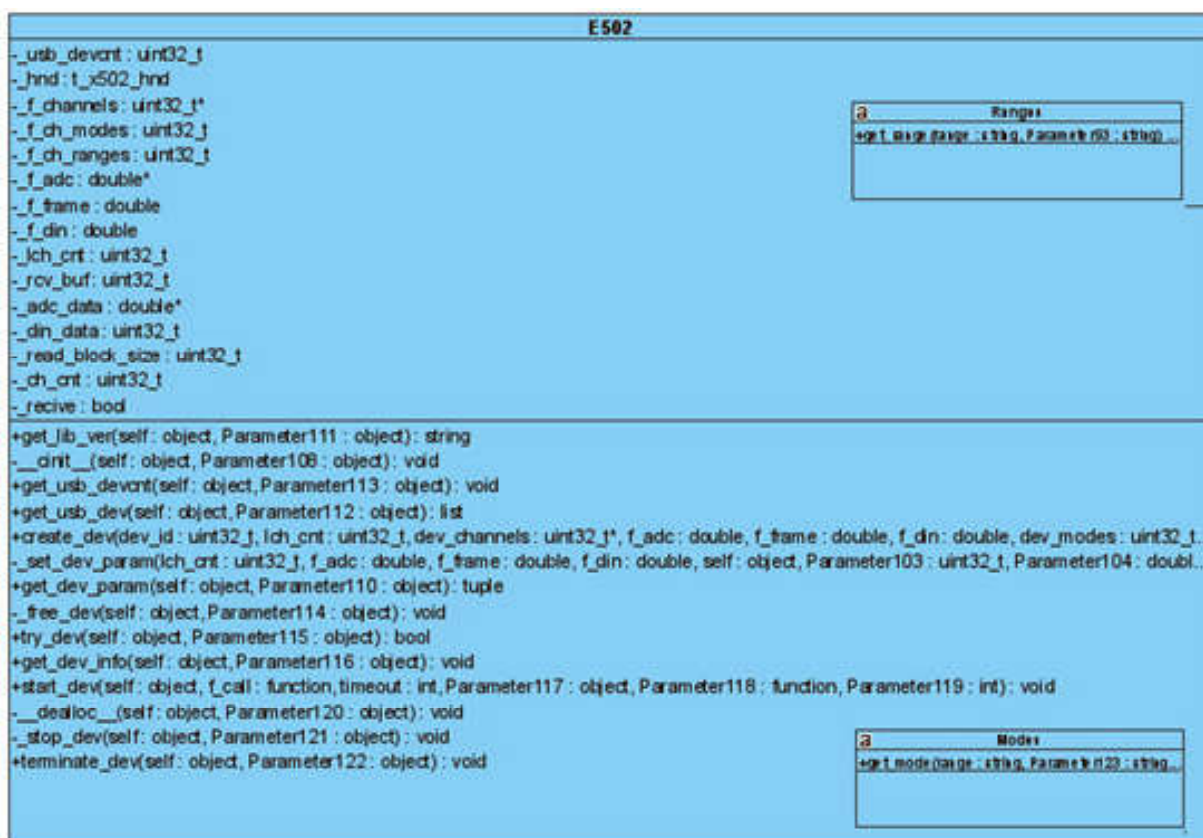


Рисунок 4 – Архитектура библиотеки для работы с устройством ЛКард E502

серийных номеров устройств, включая те, с которыми уже установлено соединение. Если нужно получить список только тех устройств, с которыми еще не установлено соединение, то в `get_usb_serial_list` можно передать аргумент `only_not_opened=True`.

Установить связь с модулем по Ethernet можно только по явно заданному IP-адресу устройства. Следует отметить, что в отличие от интерфейса USB, для корректной работы по Ethernet должны быть настроены необходимые параметры. Если USB-интерфейс в модуле E502 всегда работает и не требует дополнительной конфигурации, то для работы по интерфейсу Ethernet необходимо выполнить настройку параметров интерфейса и разрешить данный интерфейс. Следует отметить, что при разрешенном Ethernet-интерфейсе с модулем можно работать как по USB, так и по Ethernet. При этом сбор/генерация данных может выполняться одновременно только по одному интерфейсу (по которому пришла команда на запуск сбора/выдачи данных).

Выбор внешнего сигнала для задания опорной частоты синхронизации задается с помощью `set_sync_mode`, а условие запуска с помощью функции `set_sync_start_mode`. Следует отметить, что если задано внешнее событие запуска, то для того чтобы модуль перешел в режим ожидания этого события, необходимо вызвать функцию `streams_start`. Останов синхронного сбора/выдачи данных всегда осуществляется программно с помощью `streams_stop`. При использовании разъема синхронизации для организации сбора данных по принципу ведущий-ведомые, для ведущего модуля источником опорной частоты синхронизации остается внутренняя частота (режим «*internal*»), а каждый ведомый модуль использует опорную частоту и/или признак запуска сбора от внешнего мастера, т. е. для каждого ведомого модуля должен быть установлен режим «*external_master*».

При включении питания все каналы находятся в асинхронном режиме. В асинхронном режиме при вызове функции асинхронного ввода/вывода производится однократный ввод или вывод указанной информации. При этом задержка от вызова функции до непосредственно момента измерения данных для ввода или выставления указанного значения на выходе, для вывода точно не определена. Так же точно не может быть определена задержка между двумя последовательными операциями ввода/вывода.

Для однократного ввода данных с АЦП используется функция `async_get_adc_frame`, которая выполняет ввод одного кадра данных АЦП. В отличие от других функций асинхронного ввода-вывода, перед вызовом данной функции необходимо выполнить настройку модуля - необходимо задать управляющую таблицу АЦП. Измерение логических каналов внутри кадра происходит синхронно с заданной частотой сбора АЦП. Асинхронным является ввод самих кадров, то есть задержка между измерением кадров при последовательном вызове `async_get_adc_frame` не определена.

В синхронном режиме ввод или вывод данных осуществляется с заданной частотой, то есть время между соседними измерениями или выводом соседних отсчетов определено. Частоты сбора для каждого канала задаются относительно общей опорной частоты синхронизации, и запуск синхронного ввода-вывода для всех каналов осуществляется одновременно. Для запуска синхронного режима необходимо сперва с помощью функции `streams_enable` разрешить синхронный режим по требуемым каналам, а затем запустить синхронный ввод/вывод по всем разрешенным каналам с помощью `streams_start`. При синхронном вводе модуль производит измерения с заданной частотой и сам передает данные по интерфейсу в буфер. Принятые в буфер данные могут быть прочитаны программой с помощью `get_data`. Аналогично, для синхронного вывода, модуль сам по мере необходимости считывает данные из буфера и выводит считанные отсчеты с заданной частотой. Данные в буфер драйвера должны быть предварительно записаны с помощью `send_data`. При этом, если к моменту вывода очередного отсчета данные в буфер драйвера не поступили, то будет выведено предыдущее значение.

Библиотекой выделяется всего один буфер на прием и один на передачу данных. То есть значения для синхронного ввода с цифровых линий и отсчеты АЦП передаются одним потоком, также одним потоком передаются все данные на вывод. Каждый отсчет передается в виде 32-битного слова, содержащего дополнительную информацию, включающую в себя признак, к какому типу данных относится данный отсчет. Далее осуществляется разбор принятых данных на отсчеты АЦП и значения цифровых выводов и перевод отсчетов АЦП в вольты. Данные от АЦП приходят в том порядке, в котором производятся измерения, т. е. сначала измерения, соответствующие всем логическим каналам первого кадра, затем второго и т. д. При этом можно получить и нецелое количество кадров (например, если запущен синхронный ввод с цифровых линий, то заранее сложно определить, сколько в принятом блоке данных содержится отсчетов с цифровых линий, а сколько отсчетов с АЦП), в этом случае выдаются все отсчеты, включая отсчеты нецелого кадра, дополняя отсутствующие значения нецелого кадра неопределенными значениями *None*, так, чтобы в итоге было получено целое число кадров. Это необходимо для того, чтобы заранее можно было вычислить размер памяти, необходимый для хранения принятых данных, при дальнейшей обработке неопределенные значения могут быть отброшены.

Следует отметить, что если для синхронного ввода инициализация потока передачи происходит по `streams_start`, так как данные начнут поступать только после запуска синхронного ввода, то с синхронным выводом дело обстоит несколько иначе. Так как по `streams_start` уже должна начаться выдача синхронных данных, то часть данных уже должна быть загружена в модуль. Таким образом, после разрешения синхронного вывода по нужным каналам с помощью `streams_enable` и до запуска синхронного вывода с помощью `streams_start` необходимо осуществить загрузку данных синхронного потока. Для этого следует использовать функцию `send_data`, по которой будет выделен буфер на передачу и инициализирован поток на передачу, если этого не сделать, то синхронный вывод начнется лишь после того, как данные будут записаны в модуль и не будут привязаны к началу синхронного сбора/выдачи данных [6].

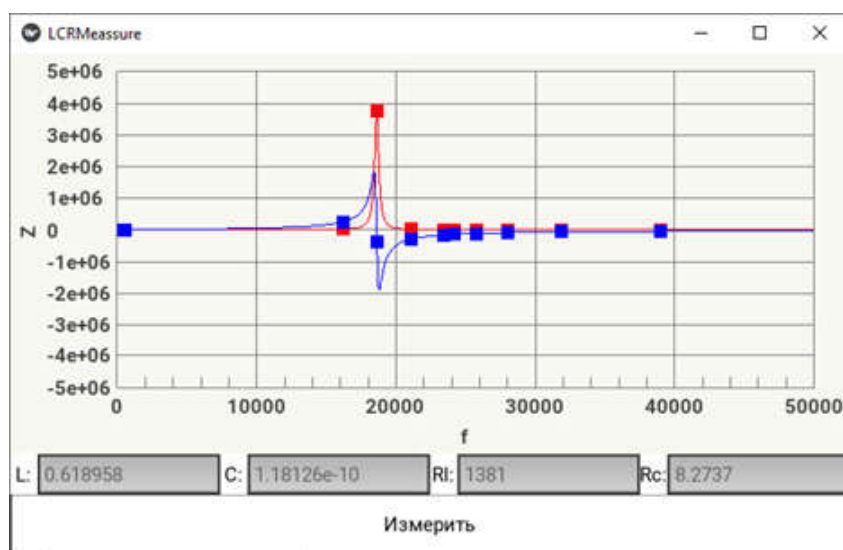


Рисунок 5 – Графический интерфейс программных средств для измерения параметров индуктивного компонента магнитометров

Кроме того, для синхронной выдачи данных на ЦАП необходимо предварительно установить начальные значения на ЦАП с помощью функции асинхронного вывода. В противном случае при начале синхронного вывода может быть небольшой переходный процесс от значения на ЦАП, которое было до запуска синхронного вывода, до выставления первых нужных значений, так как ЦАП имеет свой фильтр и ограничения на скорость изменения сигнала.

Сборка и развертывание разработанных программных средств. Для повторяемой сборки программных средств разработана программа на языке Python, автоматизирующая создание контейнера Docker, и установку в него операционной системы Ubuntu 20.04 и все перечисленные выше инструментальные средства сборки. В этом контейнере собирается драйвер ЛКард E502 и библиотека для работы с платой сбора данных ЛКард E502, а также библиотека Kivy, предоставляющая графический интерфейс пользователя.

Для сборки и развертывания программных средств был разработан файл `buildozer.spec`, необходимый для сборки APK-файла для операционной системы Android с использованием сборочной системы Buildozer. В этом файле, кроме списка зависимостей и описания приложения, о которых говорилось выше, были указаны скомпилированные файлы провайдера E502; расширения файлов, содержащих необходимые ресурсы; минимальная версия Android API, необходимая для работы приложения; файлы драйвера ЛКард E502 и архитектура процессора целевой платформы:

```
source.include_patterns = aarch64/python/*.so, aarch64/python/*.py
source.include_exts = py,kv,so
android.minapi = 21
android.add_libs_armeabi_v7a = aarch64/*.so, aarch64/devs/e502/*.so
android.arch = armeabi-v7a
garden_requirements = graph
```

Графический интерфейс пользователя и обсуждение. Графический интерфейс пользователя построен на основе библиотеки Kivy – это современное инструментальное средство построения графического интерфейса пользователя, позволяющее создавать эргономичные интерфейсы для широкого спектра устройств. Скорость выполнения Kivy сопоставима с нативной мобильной альтернативой Java для Android или Objective C для iOS, кроме того, приложения, разработанные на основе фреймворка

Kivy, могут работать и на персональных компьютерах Основные элементы графического интерфейса пользователя – это график, на котором отображается зависимость полного комплексного сопротивления от частоты, поля и кнопка для запуска процесса измерения (рисунок 5).

Заключение. Таким образом, рассмотрены различные методы измерения параметров индуктивности и предложен способ определения параметров индукционных компонентов магнитометров на основе существующего метода преобразования полного комплексного сопротивления в напряжения и его реализация на основе платы сбора данных ЛКкард E502. Рассмотрены программные средства для определения амплитуды и фазы измеряемого напряжения, основанные на минимизации среднеквадратического отклонения между измеряемым напряжением и моделью гармонического сигнала. Разработаны программные средства для измерения параметров активного индуктивного компонента, на который воздействуют фоновые электромагнитные поля. В предложенном способе это влияние нивелируется тем, что параметры индуктивности вычисляются по множеству измерений полного комплексного сопротивления на различных частотах, и находятся исходя из наилучшего соответствия измеренным значениям в смысле минимума суммарного квадратического отклонения.

Литература

1. *Bryakin I.V.* Two-Axis Fluxgate Magnetometer with a New Principle of Excitation / I.V. Bryakin, I.V. Bochkarev, R.R. Khramshin // Proceedings 2020 International Russian Automation Conference RusAutoCon 2020. С. 693–698.
2. *Bochkarev I.V.* Control of operational condition of electromagnetic devices of automation systems / I.V. Bryakin, I.V. Bochkarev, R.R. Khramshin // Proceedings 2018 International Conference on Industrial Engineering, Applications and Manufacturing. ICIEAM 2018.
3. *Верзунов С.Н.* Программные средства для сбора и анализа геоэкологических данных с помощью устройства ЛКкард / С.Н. Верзунов, В.В. Кириллова // Вестник КРСУ. 2018. Т. 18. № 4. С. 43–47.
4. *Верзунов С.Н.* Способ измерения параметров катушек индуктивности магнитометров и его реализация на основе платы сбора данных ЛКкард E 502 / С.Н. Верзунов // Проблемы автоматизации и управления. 2018. № 2 (35). С. 94–102.
5. *Сабанин В.Р.* Модифицированный генетический алгоритм для задач оптимизации в управлении / В.Р. Сабанин, Н. Смирнов, А.И. Репин // Exponenta Pro: Математика в приложениях. 2004. № 3–4. С. 78–85.
6. *Verzunov S.N.* Device for Measuring Parameters of Coils of Induction Magnetometers / S.N. Verzunov, I.V. Bochkarev, V.R. Khramshin // International Russian Automation Conference (RusAutoCon). 2020. P. 699–703.