УДК 621.389
DOI: 10.22213/2413-1172-2019-3-72-81

К ВОПРОСУ ВЫБОРА ПРОГРАММНЫХ ИНСТРУМЕНТОВ ДЛЯ МОДЕЛИРОВАНИЯ РАДИОЛОКАЦИОННЫХ СИСТЕМ *

А. С. Раев, магистрант, ИжГТУ имени М. Т. Калашникова, Ижевск, Россия
А. Н. Копысов, кандидат технических наук, доцент, ИжГТУ имени М. Т. Калашникова, Ижевск, Россия
В. В. Хворенков, доктор технических наук, профессор, ИжГТУ имени М. Т. Калашникова, Ижевск, Россия

Ключевым трендом современных радиотехнических систем является переход к цифровым технологиям. Особенно ярко это выражено в процессе развития и широкого внедрения технологии программно-определяемых радиосистем, или иначе software-defined radio(SDR). В результате формирования данной технологии появилась возможность разработки современных радиолокационных систем, используя программные модели, базирующиеся на технологии SDR .

В работе рассматривается вопрос выбора разработчиком программных инструментов для моделирования и проектирования радиолокационных систем. Для ответа на данный вопрос выполняется оценка двух сред программирования SDR-систем – Lab VIEW и GNU Radio, производится их общие описание. Рассматриваются их структуры, модели основных компонентов, интерфейсов, а также системы программного обеспечения и системные требования.

Приведены примеры использования инструментов в исследовании радиолокационных систем на основе сигнала с линейной частотной модуляцией. На основе алгоритма генерации такого сигнала разработаны модели генераторов и проведена оценка работы исследуемых сред программирования SDR. Построены графики генерируемого сигнала с выбранными параметрами как во временной, так и в частотной областях. В процессе работы и конфигурации примеров проведен анализ вычислительных возможностей и временных затрат, сопряженных с работой оцениваемых сред программирования SDR.

Сформулированы основные достоинства; к их числу относятся универсальность и доступность, а также недостатки, включающие затраты времени на программирование и отсутствие гибкости проектов, описанных оболочек, проведено их сравнение в совокупности между собой.

Ключевые слова: Lab VIEW, GNU Radio, программно-определяемая радиосистема, периферийная радиосистема, C++, Python, Gnu Octave.

Введение

Основная тенденция развития радиотехники на данный момент основывается на переходе к цифровым технологиям. Особенно ярко это выражено в процессе развития технологии программно-определяемых радиосистем *(software-defined radio*). Программно-определяемая радиосистема – радиоприемник/передатчик, использующий технологию, позволяющую с помощью программного обеспечения устанавливать и изменять рабочие параметры.

В результате развития данной технологии появилась возможность разработки радиотехнических систем, используя программные модели. Это привело к появлению на мировом рынке *SDR*-устройств. За короткое время появилось множество производителей программируемых структур. К популярным моделям данных систем относятся *universal software radio peripheral* (USRP) (URL: https://www.ettus.com/wp-content/uploads/2019/01/X300_X310_Spec_Sheet.pdf), *Hack RF* (URL: https://github.com/mossmann/wiki/HackRF-One/) и другие платформы. Широкое использование технологии *SDR* привело к такому же стремительному развитию сред программирования и конфигурации.

В работах последних лет [1–4], можно найти большое число статей, посвященных моделированию радиолокационных систем с использованием *SDR*-устройств. Однако большинство статей отражают исследования, проводимые авторами с использованием лишь одного из инструментов программирования. Среди существующих оболочек программирования в публикациях часто встречаются такие программные продукты, как *GNU Radio* (URL: https://wiki.gnuradio.org/index.php/Main_Page) и *LabVIEW* [5]. При этом каждая из оболочек заняла свою нишу в программном обеспечении *SDR*-устройств. Таким образом, перед разработчиком (инженером-исследователем) нередко

* Исследование выполнено в рамках проекта № 11.04.02/18КАН.

встает вопрос выбора программных инструментов для моделирования и проектирования радиолокационных систем.

С целью поиска ответа на данный вопрос в работе проводится оценка систем проектирования *SDR*-устройств и анализ особенностей использования систем при создании программных моделей, в том числе и их вычислительных возможностей. Для достижения поставленной цели в работе решаются следующие задачи. Во-первых, проводится сравнение структур и системных особенностей систем проектирования *GNU Radio* и *LabVIEW*. Во-вторых, дается анализ результатов моделирования на примере радиолокационной задачи, в виде формирования зондирующего сигнала с линейной частотной модуляцией. В-третьих, определяются вычислительные возможности и затраты, связанные с системами проектирования *GNU Radio* и *LabVIEW*.

Среда программирования *GNU Radio*

GNU Radio является программным инструментарием как для моделирования процессов, которые наблюдаются в радиосистемах, так и для симуляции таковых в *SDR*-устройствах. Данный продукт включает в себя библиотеки компонентов, разработанных на языках *C++* [6] и *Python* [7].

Основные компоненты приведенной оболочки имеют довольно простую иерархию. Классификация компонентов *GNU Radio* представлена на рис. 1.

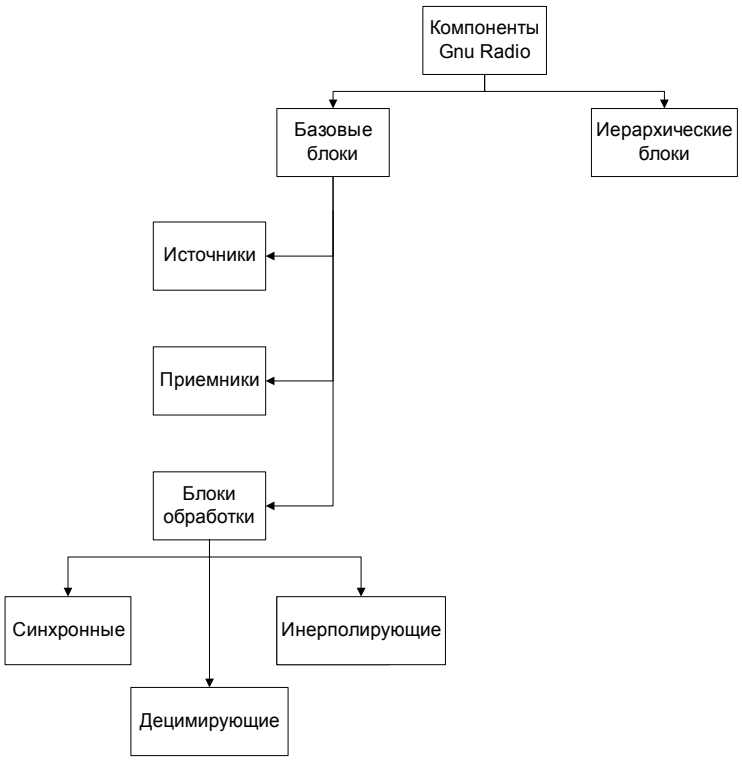


Рис. 1. Классификация компонентов *GNU Radio*

Fig. 1. Classification of *GNU Radio* blocks

Первые из базовых компонентов системы представляют собой блоки-источники сигналов, предназначенные для генерации отчетов по заданному алгоритму. К основным блокам можно отнести: источники сигнала синусоидальной, пилообразной или прямоугольной формы (*Signal source*), источники белого шума, источники постоянного сигнала, источники файлов с набором данных в двоичном формате (*File Source*) или звуковых файлов в формате *WAV*.

Блоки-приемники, имеющиеся в системе *GNU Radio*, позволяют отобразить принимаемый сигнал во временной или частотной области. Их функционал можно сопоставить с работой осциллографа (*Time sink*) или анализатора спектра (*Frequency sink*) [8]. Также имеются компоненты данного вида, отвечающие за запись данных в бинарные файлы (*File sink*) или файлы типа *WAV*.

К третьей части базовых блоков относятся блоки обработки. Их можно разделить на синхронные (количество отсчетов входных сигналов равно количеству отсчетов в выходных сигналах), децимирующие и интерполирующие

(понижение или повышение количества выходных отсчетов). Среди блоков обработки имеются как простые, так и более сложные. К числу простых модулей относятся модули, реализующие задержку сигнала на заданное количество отсчетов, или же модули, чьей задачей является выполнение элементарной арифметической операции над каждым входным отсчетом. В свою очередь, более сложными являются фильтры или блоки, выполняющие преобразование Фурье [9].

В общем случае каждый блок обработки принимает один или несколько входных потоков данных, обрабатывает их и генерирует один или несколько выходных потоков данных (количество которых может отличаться от количества входных потоков). Некоторые блоки обработки принимают на вход или передают на выход цифровые отсчеты только в виде выборок фиксированной длины (векторов). Так происходит, например, в блоке, выполняющем преобразование Фурье. В цепочке соединенных друг с другом блоков тип выходных данных одного блока должен совпадать с типом входных данных следующего [10].

Иерархические блоки представляют собой совокупность нескольких базовых блоков с их системой связи и параметрами, зарезервированными в одной структурной ячейке. Иерархичность системы приводит к уменьшению загруженности графа без изменения его функционала.

В качестве примера рассмотрен потоковый граф в программе *GNU Radio* (рис. 2, *а*) и соответствующий данному графу иерархический блок (рис. 2, *б*).

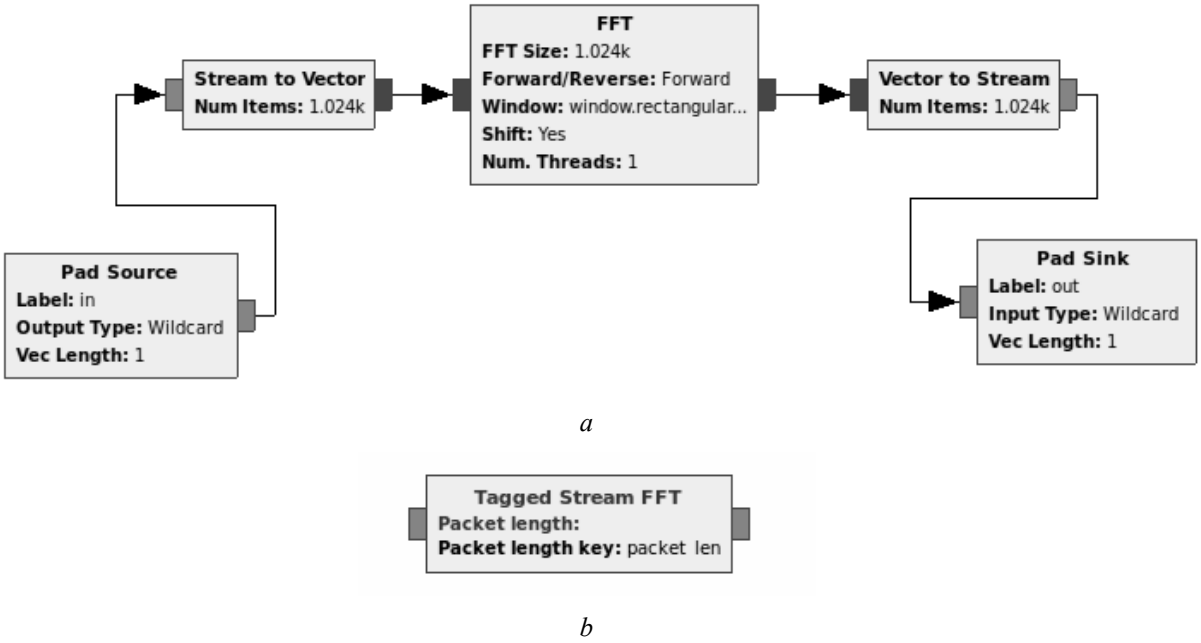


Рис. 2. Пример иерархического блока: *а* – потоковый граф, *b* – соответствующий ему иерархический блок

Fig. 2. Example of hier-block: *a* - flow graf; *b* - hier-block

Также для анализа процессов используется *GNU Octave* [11] – система математических вычислений, которая является одной из популярных и используемых при работе с *GNU Radio*, так как данный пакет имеет собственный набор скриптов для чтения и анализа выходных данных. Кроме того, данный продукт также является бесплатным и совместим с *MATLAB*, так как их синтаксис в большей части совпадает.

Программная платформа *LabVIEW*

Среда разработки *LabVIEW* представляет собой высокоэффективную оболочку графического программирования, в которой можно создавать гибкие и масштабируемые приложения измерений, управления и тестирования с минимальными временными затратами. *LabVIEW* сочетает в себе гибкость традиционного языка программирования с интерактивной технологией виртуального прибора (ВП).

Основным структурным элементом *LabVIEW* являются файлы проектов. Данные структуры предназначены для группирования файлов и создания спецификации построения приложений, а также для загрузки файлов в целевое устройство.

Виртуальные приборы – концепция, в соответствии с которой организуются программно-управляемые системы сбора данных и управления техническими объектами и технологическими процессами.

Сущность данной концепции состоит в том, что система организуется в виде программной модели некоторого реально существующего или гипотетического прибора, причем программно реализуются не только средства управления, но и логика работы прибора.

Программирование в *LabVIEW* основано на архитектуре потоков данных. Последовательность выполнения операторов в таких языках определяется наличием данных на входах этих операторов. Операторы, не связанные по данным, выполняются параллельно в произвольном порядке. Принцип потока данных упрощает разработку многопоточных и многозадачных программ.

Еще одной особенностью *LabVIEW* является узел связи с приложением *MATLAB* [12]. Создание, загрузка и редактирование скриптов *MATLAB* выполняется в блоке *MATLAB scrip node*. Для выполнения операций с использованием *MATLAB scrip node* система должна быть установлена на компьютере.

Кроме основного программного продукта, который является универсальным инструментом моделирования, имеется среда разработки *LabVIEW Communication System Design*, созданная для конфигурирования систем беспроводной связи. Оболочка предназначена для уменьшения времени прототипирования систем связи, в том числе пятого поколения 5*G* [13].

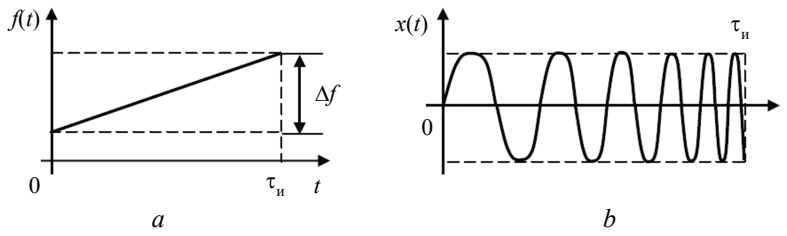
Основной из отличительных особенностей данной программы от *GNU Radio* являются автономные приложения. Автономное приложение позволяет запускать *VI*, не инсталлируя на компьютер среду проектирования *LabVIEW*. Данный тип приложений позволяет разработчикам использовать уже имеющиеся программы для моделирования систем без установки *LabVIEW*, но при условии, что их функционал полностью соответствует требованиям. В противном случае может потребоваться корректировка работы данных приложений, а она возможна лишь при инсталляции *LabVIEW*.

Построение генератора ЛЧМ-сигнала в *GNU Radio* и *LabVIEW*

В качестве примера для сравнения *GNU Radio* и *LabVIEW* для анализа радиолокационной задачи рассмотрим вариант реализации зондирующего сигнала с линейной частотной модуляцией (ЛЧМ/*FMCW*).

Из литературы известно, что частота сигнала ЛЧМ в пределах длительности импульса изменяется по линейному закону, а фаза – по параболическому.

Сигнал представлен на рис. 3, *a*, *b*.



a *b*

Рис. 3. ЛЧМ-радиоимпульс: *a* – изменение частоты; *b* – временное представление

Fig. 3. Pulse of frequency-modulated continuous-wave: *a* - frequency change; *b* - time representation

Для ЛЧМ-импульсов закон частотной модуляции описывается выражением

$$f = f_0 + \Delta f \frac{t}{\tau},$$

где f_0 – минимальная частота; $\Delta f = F_{\max} - F_{\min}$ – девиация частоты.

Ему соответствует закон изменения фазы:

$$\varphi(t) = f_0 + \int_0^t \Delta f \frac{t}{\tau} dt.$$

Для дискретизированного сигнала с частотой дискретизации *Fd* выражение преобразуется в сумму:

$$\varphi(k) = \left(f_0 + \sum_{k=0}^{N} \Delta f \frac{k}{N} \right) Td,$$

где N – количество отсчетов в импульсе; Td – период дискретизации.

Комплексная амплитуда дискретизированного ЛЧМ-импульса выражается как

$$X(k) = A_0 \exp\left(i 2\pi \varphi(k)\right), \qquad (1)$$

где A_0 – амплитуда импульса.

Генерация ЛЧМ-сигнала в *GNU Radio*

Рассмотрим реализацию генератора ЛЧМ-импульсов в программе *GNU Radio*. В данной среде имеется стандартный блок генерации ЛЧМ-импульсов, использующий представленный выше алгоритм, под названием *Signal Generator FMCW*. Формируемый блоком сигнал имеет закон изменения частоты, представленный на рис. 4.

Особенностью данного сигнала является наличие области линейного спада частоты и постоянной частоты в совокупности с линейным подъемом частоты.

Вводными данными для блока являются:

Sample rate – частота дискретизации;

Samples CW – количество отсчетов для постоянной частоты;

Samples up-chrip – количество отсчетов для подъема частоты;

Samples down-chrip – количество отсчетов для спада частоты;

Frequency CW – минимальная частота линейных участков;

Sweep frequency – максимальная частота линейных участков;

Amplitude – амплитуда комплексного сигнала.

Реализация потокового графа данной системы приведена на рис. 5.

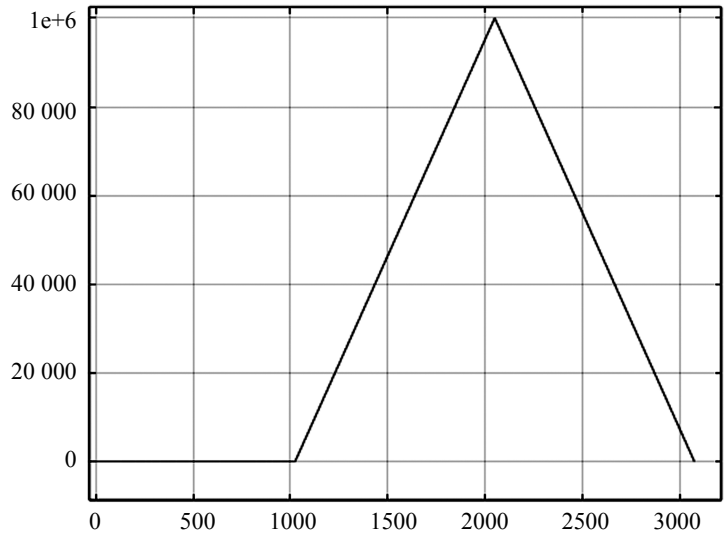


Рис. 4. Закон изменения частоты формируемого сигнала блоком *Signal Generator FMCW*

Fig. 4. Function of changing frequency of the signal block *Signal Generator FMCW*

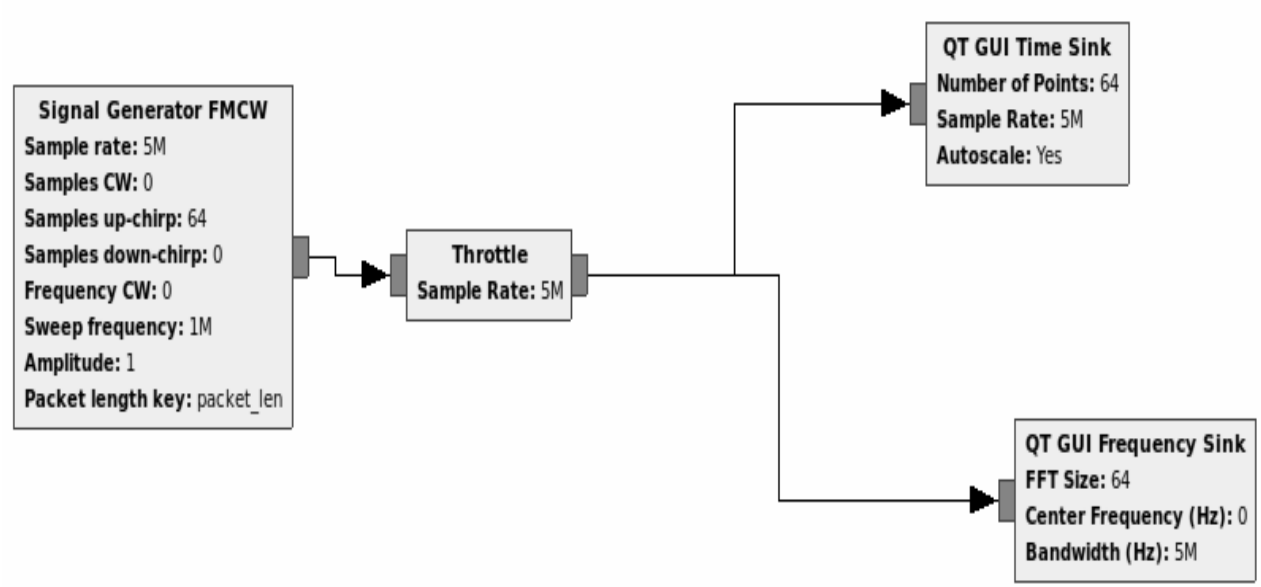


Рис. 5. Блок генерации ЛЧМ-импульсов *Signal Generator FMCW*

Fig. 5. Flow graf generation *FMCW* pulse

Для генерации сигнала пилообразной формы необходимо избавиться от участков постоянной частоты и спада частоты. Это можно реализовать обнулением количества отсчетов данных участков. Характеристики генерируемого сигнала выбраны для большей наглядности:

Fd = 5 МГц – частота дискретизации;

$\Delta f = F_{max} - F_{min} = 1 - 0 = 1$ МГц – девиация;

$\tau = nTd = 2^6/5 \cdot 10^6 = 12{,}8$ мкс – длительность ЛЧМ-импульса.

Во временной области сигнал представлен на рис. 6.

В частотной области сигнал изображен на рис. 7.

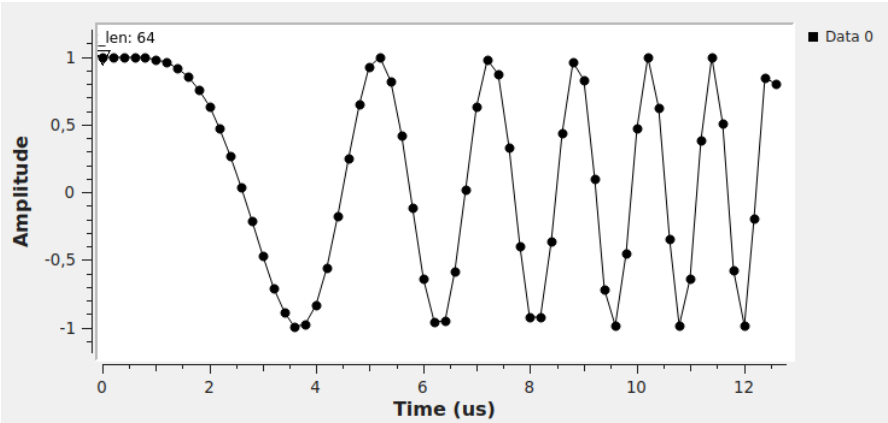


Рис. 6. Осциллограмма ЛЧМ-импульса в программе *GNU Radio*

Fig. 6. Time scope *FMCW* pulse in *GNU Radio*

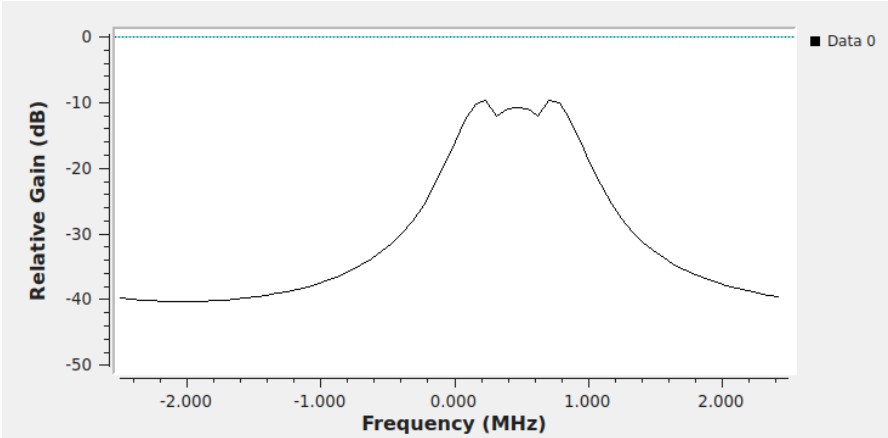


Рис. 7. Спектрограмма ЛЧМ-импульса в программе *GNU Radio*

Fig. 7. Frequency scope *FMCW* pulse in *GNU Radio*

Генерация ЛЧМ-сигнала в *LabVIEW*

Для сравнения сред рассмотрим реализацию блока генерации ЛЧМ-сигнал в программе *LabVIEW*. Модель, которая работает по описанному алгоритму, отсутствует в данной оболочке. Для решения данной проблемы было создано приложение, выдающее отчеты согласно формуле (1).

Блок-схема блока представлена на рис. 8.

В результате работы алгоритма были выведены осциллограмма и спектр реализованного сигнала. Осциллограмма сигнала представлена на рис. 9.

Спектрограмма сигнала представлена на рис. 10.

Сопоставление технических требований и анализ вычислительных возможностей сред моделирования

Для сравнения вычислительных возможностей оцениваемых программных продуктов прежде всего необходимо определить предъявляемые разработчиками системные требования и сравнить их в совокупности между собой. К ключевым сравниваемым параметрам системных требований относятся: тип процессора, операционная система и используемое для установки дисковое пространство.

В официальных источниках (URL: https://www.ni.com) указано, что для работы *LabVIEW* требуется процессор *Pentium 4M*

(или эквивалентный) для 32-разрадяной системы и *Pentium 4G1* (или эквивалентный) для 64-разрядной системы. В свою очередь, *GNU Radio* (URL: https://www.wiki.gnuradio.com) поддерживают процессоры *Intel x86* или более поздние. В рамках проводимой работы были использованы ПК с процессорами *Intel Core i7*.

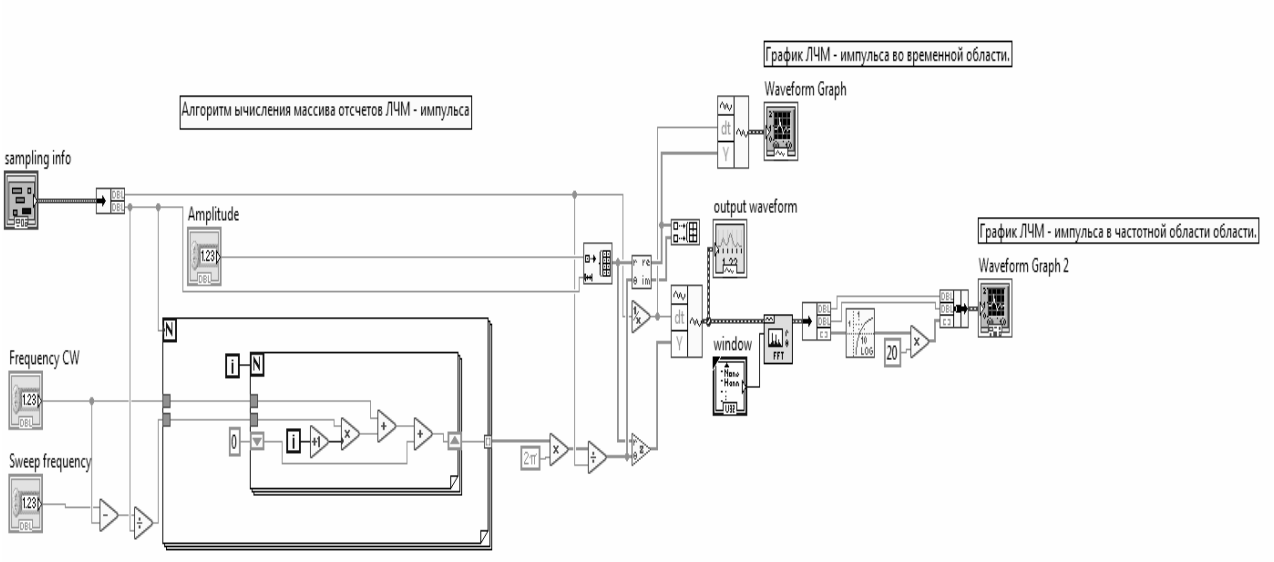


Рис. 8. Блок-схема генерации ЛЧМ-импульса в программе *LabVIEW*

Fig. 8. Block diagram of generation *FMCW* pulse in *LabVIEW*

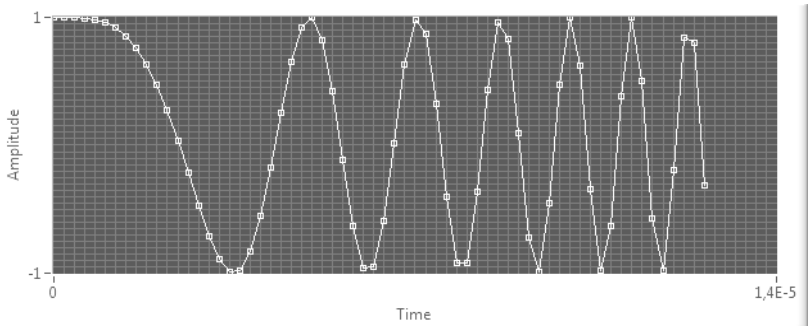


Рис. 9. Осциллограмма ЛЧМ-импульса в программе *LabVIEW*

Fig. 9. Time scope *FMCW* pulse in *LabVIEW*

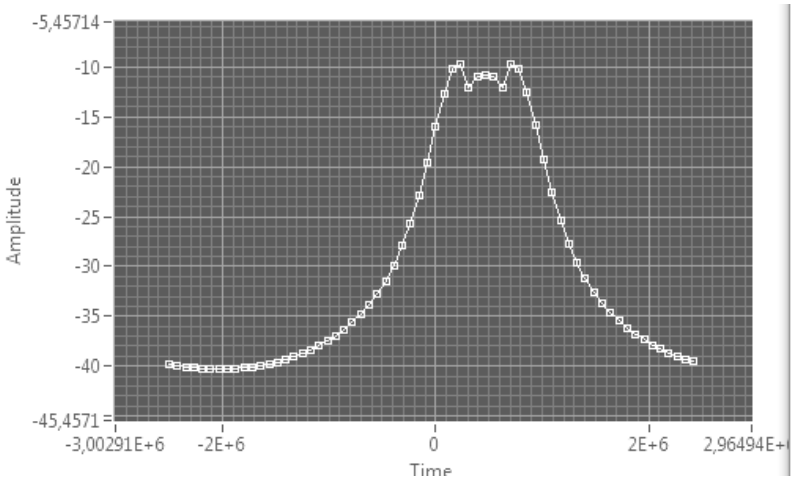


Рис. 10. Спектрограмма ЛЧМ-импульса в программе *LabVIEW*

Fig. 10. Frequency scope *FMCW* pulse in *LabVIEW*

Согласно документации *LabVIEW*, как и *GNU Radio*, поддерживают 32-разрядные и 64-разрядные операционные системы (ОС). Однако *LabVIEW* имеет возможность установки во всех базовых ОС, таких как *Windows*, *MAC OS* и *Linux*, тогда как *GNU Radio* в основном разрабатывалась под *Linux* и поэтому основной способ для работы данной среды в других ОС является использование виртуальных машин.

Объем памяти, необходимый для установки *LabVIEW*, составляет 1 ГБ операционной памяти и 5 ГБ свободного дискового пространства. В свою очередь, *GNU Radio* требует лишь 3 ГБ свободного дискового пространства.

На этапе анализа вычислительных возможностей основными параметрами выступали загрузка центрального процессора (ЦП) и загрузка физической памяти, используемой при работе примеров генерации зондирующих ЛЧМ-импульсов. Для оценки использовались стандартные инструменты ОС, такие как «Диспетчер задач» в *Windows* для анализа *Lab VIEW* и *System Monitor* в *Linux* для анализа *GNU Radio*.

Загрузка ЦП в динамике представлена на рис. 11 для *LabVIEW*. Уровень загрузки ЦП при работе в *LabVIEW* составил 16 %. Аналогичный график построен для *GNU Radio* (рис. 12). В результате анализа видно, что загрузка ЦП во втором случае составила также 16 %. Однако по времени процесс компиляции программы занял больше времени.

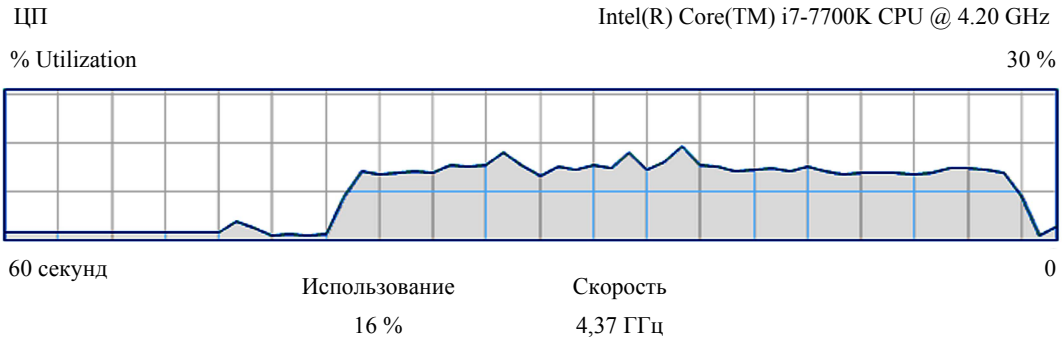


Рис. 11. Загрузка ЦП при работе в *LabVIEW*

Fig. 11. CPU load in *LabVIEW*

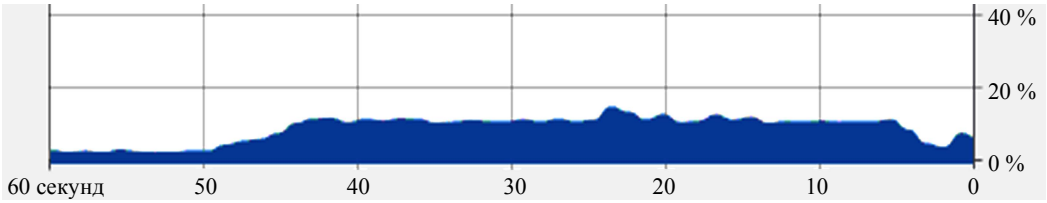


Рис. 12. Загрузка ЦП при работе в *GNU Radio*

Fig. 12. CPU load in *GNU Radio*

Временные затраты в общем случае выражаются в затратах на создание программ анализируемых примеров. Для описания собственных блоков в языке *C++* и *Python* используется подход объектно ориентированного программирования. Описание классов, их объектов и методов дает возможность разработать модули в достаточной точности, кроме того, имеется возможность из существующих элементов создать новые модификации. Множество интернет-ресурсов предоставляют информацию о структуре описания и создания собственных блоков в программе *GNU Radio* в достаточно большом объеме, причем в большей части это иностранные ресурсы. К ним можно отнести *Gi tHub* (Git Hub. URL: https://github.com/gnuradio/), *Wiki GNU Radio*, *Hubr* (Hubr. URL: https://habr.com/) и др. При этом для разбора документации и создания основного блока генерации ЛЧМ-сигнала в среде *GNU Radio* потребовалось потратить около 40 мин, тогда как в *LabVIEW* за счет использования языка *G* и концепции ВП на аналогичную разработку потребовалось всего 10-15 мин.

Заключение и выводы

В работе произведено сравнение двух сред программирования *SDR*-устройств. При анализе теоретического описания данных оболочек в зарубежной и отечественной литературе выявлены основные достоинства и недос-

татки каждой из них в совокупности между собой.

Во-первых, полноценный язык графического программирования *LabVIEW* интуитивно понятен и имеет возможности интерактивной генерации кода. Поэтому процесс создания и компиляции кода, используя язык *G*, менее затратен по времени. Создание кода генерации ЛЧМ-импульса в *LabVIEW* заняло около 10-15 мин, в свою очередь, при создании и компиляции кода на *C++* в *GNU Radio* это заняло бы около 40 мин. Однако загрузка кода *LabVIEW* в разработанное устройство потребует специализированного программного обеспечения, что для кода в *C++* не имеет необходимости.

Во-вторых, *LabVIEW* включает в себя множество шаблонов приложений и тысячи примеров, как и *GNU Radio*. Однако без знания основ программирования на языках *C++* и *Python* в *GNU Radio* разработчик имеет возможность пользоваться лишь имеющимися блоками. Это приводит к отсутствию гибкости проектирования.

В-третьих, бóльшая часть информации по *GNU Radio* описана лишь в иностранной литературе. Для разработки и изучения потребуется немалое время на перевод и анализ информации. Однако *LabVIEW* имеет достаточно много информации в отечественной литературе, что позволяет не тратить свое время на ее обработку в процессе исследования.

В-четвертых, *LabVIEW* не является бесплатным инструментом в отличие от *GNU Radio*. Из этого следует, что для разработчика простых систем лучшим вариантом выбора является *GNU Radio* как менее затратный инструмент. Тогда как использование *LabVIEW* оправдывается лишь в обширных исследованиях с множеством различных программируемых моделей, а также для разработки многозадачных систем.

Последним пунктом можно выделить, что *LabVIEW* имеет возможность программирования *SDR*-систем, лишь произведенных компанией *Ettus Reserch* и *National Instruments*. Продукцией данного производителя являются все модели *USRP*-устройств. В свою очередь, *GNU Radio* дает нам возможность программировать *SDR*-устройства не только данной компании, но и множества других производителей данных систем. К ним относятся такие модели, как *Hack RF*, *RTL-SDR*, *ADFMCOMMS2-EBZ* и др.

На основе примера с генерацией ЛЧМ-импульса в приведенных программных продуктах была произведена оценка отличий между софтами в выводе данных. По полученным осциллограммам и спектрограммам сгенерированного сигнала в программах можно сделать вывод, что полученные результаты, выраженные в графиках, не имеют каких-либо конкретных отличий. Загрузка основных ресурсов ПК, таких как центральный процессор и физическая память, для работы разработанных примеров не имела сильных отличий. Например, загрузка ЦП как в первом, так и во втором случае была равна 16 %. Сказанное позволяет заключить, что эти программные оболочки хотя и имеют существенные отличия, однако выбор программного продукта зависит от разработчика и характера исследовательской работы.

Библиографические ссылки

1. Sundaresan S., Anjana C., Zacharia T., Gandhiraj R. [Real time implementation of FMCW radar for target detection using GNU radio and USRP]. *Communications and Signal Processing ICCSP*, Chengdu, 2015, pp. 1530-1534.

2. Krzysztof Stasiak, Piotr Samczynski. [FMCW Radar Implemented in SDR Architecture Using a USRP Device]. *Warsaw University of Technology, Institute of Electronic Systems*. Warsaw, Poland, 2017, 4 p.

3. Nuss B., Sit and Y.L., Zwick T. [3D Radar Image Fusion using OFDM-Based MIMO Radar]. *German Microwave Conference (GeMiC).* Bochum, 2016, pp. 209-212.

4. Gromek D., Krysik P., Ndini K., Samczyski P. [FMCW SAR based on USRP hardware platform]. *Proc. IEEE Radar Conference*, May 19th-23rd, 2014. Cincinnati, OH, USA, pp. 0552-0555.

5. *Васильев А. С.*, *Лашманов О. Ю.* Основы программирования в среде LabVIEW. СПб. : Ун-т ИТМО, 2015. 82 с.

6. *Бьерн Страуструп*. Язык программирования C++ для профессионалов. Изд. 2-е, испр. : пер. с англ. М. : ИНТУИТ, 2016. 670 с.

7. *Макграт М.* Программирование на Python для начинающи. М. : Эксмо, 2015. 192 с.

8. *Раушер К.*, *Йанссен Ф.*, *Минхольд Р.* Основы спектрального анализа / пер. с англ. С. М. Смольского ; под ред. Ю. А. Гребенко. Изд. 2-е, испр. М. : Горячая линия – Телеком, 2014. 226 с. ISBN 978-5-9912-0429-3.

9. *Ershad Junus Amin*, *Andriyan Bayu Suksmono*, *Achmad Munir*. [Accuracy Analysis of FM Chirp in GNU Radio-based FMCW Radar for Multiple Target Detection]. *Radio Telecommunication and Microwave Laboratory, School of Electrical Engineering and Informatics, Institut Teknologi Bandung*. Bandung, Indonesia, 2014, pp. 115-119.

10. *Липатников В. С.* Отладка алгоритмов обработки радиолокационных сигналов в системе GNU Radio // Цифровая обработка сигналов. 2015. № 4. с. 1–68.

11. *John W.* [Eaton, David Bateman]. *GNU Octave. Free your number*. Free Software Foundation Publ., Inc., Boston, USA, 2018, 1043 p.

12. *Гадзиковский В. И.* Цифровая обработка сигналов. М. : СОЛОН-ПРЕСС, 2015. 766 с.

13. *Bruce Black.* [Introductory communications systems]. National Instruments Publ., 2014, 157 p.

References

1. Sundaresan S., Anjana C., Zacharia T., Gandhiraj R. [Real time implementation of FMCW radar for target detection using GNU radio and USRP]. *Communications and Signal Processing ICCSP*, Chengdu, 2015, pp. 1530-1534.

2. Krzysztof Stasiak, Piotr Samczynski. [FMCW Radar Implemented in SDR Architecture Using a USRP Device]. *Warsaw University of Technology, Institute of Electronic Systems*. Warsaw, Poland, 2017, 4 p.

3. Nuss B., Sit and Y.L., Zwick T. [3D Radar Image Fusion using OFDM-Based MIMO Radar]. German Microwave Conference (GeMiC), Bochum, 2016, pp. 209-212.

4. Gromek D., Krysik P., Ndini K., Samczyski P. [FMCW SAR based on USRP hardware platform]. *Proc. IEEE Radar Conference*, May 19th-23rd, 2014, Cincinnati, OH, USA, pp. 0552-0555.

5. Vasilev A.S., Lashmanov O.Y. *Osnovi programmirovaniya v srede LabVIEW* [LabVIEW programming basics]. St. Petersburg, ITMP University Publ., 2015, 82 p. (in Russ.).

6. Bjarne Stroustrup. *Yazyk programmirovaniya S++ dlya professionalov* [The C++ programming language for professionals]. Moscow, INTUIT Publ., 2016, 670 p. (in Russ.).

7. Makgrant M. *Programmirovanie na Python dlya nachinauschih*. [Programming on Python for beginners]. Moscow, Escimo Publ., 2015, 192 p. (in Russ.).

8. Christofer Rausher, Volker Janssen, Roland Minihold. *Osnovy spektral'nogo analiza* [Fundamentals of spectrum analysis]. Moscow, Telecom Publ., 2014, 226 p. (in Russ.). ISBN 978-5-9912-0429-3.

9. Ershad Junus Amin, Andriyan Bayu Suksmono, Achmad Munir. [Accuracy Analysis of FM Chirp in GNU Radio-based FMCW Radar for Multiple Target Detection]. *Radio Telecommunication and Microwave Laboratory, School of Electrical Engineering and Informatics, Institut Teknologi Bandung*. Bandung, Indonesia, 2014, pp. 115-119.

10. Lipatnicov V.S. [Radar signal processing algorithms debug in GNU radio system]. *Digital processing signals*, 2015, vol. 4, pp. 61-68 (in Russ.).

11. John W. [Eaton, David Bateman]. *GNU Octave. Free your number*. Free Software Foundation Publ., Inc., Boston, USA, 2018, 1043 p.

12. Gadzikovskii V.I. *Cifrovaja obrabotka signalov* [Digital signal processing]. Moscow, SOLON PRESS Publ., 2015, 766 p. (in Russ.).

13. Bruce Black. [Introductory communications systems]. National Instruments Publ., 2014, 157 p.

On the Question of Choice of Software Tools for Modeling Radio Systems

A.S. Raev, Master's Degree Student, Kalashnikov ISTU, Izhevsk, Russia
A.N. Kopysov, PhD in Engineering, Associate Professor, Kalashnikov ISTU, Izhevsk, Russia
V.V. Khvorenkov, DSc in Engineering, Professor, Kalashnikov ISTU, Izhevsk, Russia

The main trend of advanced radar systems is the transition to digital technology. This moment is clearly expressed in the process of development and wide implementation of the technology of the software-defined radio (SDR). The software-defined radio technology gives an opportunity for all radio engineers to make radar systems by using the SDR based program models. The paper deals with the question of choosing the software tools for modeling and design of radar systems. Two software tools for programming SDR systems are analyzed: Lab VIEW u GNU Radio. The paper describes their structures, interface, models of basic components and system requirements to these programs. Examples are given for application of these tools in research of radar systems based on the signal with frequency-modulated continuous wave (FMCW) generation. By means of the algorithm of such signal generation, models of FMCW generators are developed and operation of the studied SDR programming systems is estimated. Diagrams for the generated FMCW signals with the chosen parameters in time and frequency area are drawn. The computational capability and time consumption related to operation of the studied systems of SDR programming are analyzed within operation and configuration of examples. Basic advantages (including versatility and availability) and disadvantages (including time consumption and the absence of project flexibility) of the discussed systems are stated; their mutual comparison is carried out.

Keywords: Lab VIEW, GNU Radio, software-defined radio, peripheral radio system, C++, Python, Gnu Octave.

Получено 13.08.2019

Образец цитирования

Раев А. С., *Копысов А. Н.*, *Хворенков В. В.* К вопросу выбора программных инструментов для моделирования радиолокационных систем // Вестник ИжГТУ имени М. Т. Калашникова. 2019. Т. 22, № 3. С. 72–81. DOI: 10.22213/2413-1172-2019-3-72-81.

For Citation

Raev A.S., Kopysov A.N., Khvorenkov V.V. [On the issue of choosing software tools for modeling radar systems]. *Vestnik IzhGTU imeni M. T. Kalashnikova*, 2019, vol. 22, no. 3, pp. 72-81 (in Russ.). DOI: 10.22213/2413-1172-2019-3-72-81.