Постановка задачи Перед разработчиками (программистами/инженерами), которые занимаются разработкой собственных устройств / программно-аппаратных комплексов, часто возникают задачи их отладки, тестирования, модификации и реверс-инжиниринга. Под устройствами мы понимаем любые цифровые радиотехнические устройства и приборы с “жесткой” (непрограммируемой) логикой. Отличие программно-аппаратных комплексов в том, что они имеют в своем составе микросхемы “гибкой” (программируемой) логики, такие как микроконтроллеры или программируемые логические интегральные схемы. Для описания любого технического устройства лучше всего подходит концепция “черного ящика” /1, 2/. Если говорить обобщенно, то любая цифровая техника представляет собой “черный ящик”, с определенным количеством входов и выходов. Определенная последовательность входных сигналов приводит к инициализации огромного количества внутренних механизмов проверок и цепей обратных связей, которые в свою очередь приводят к формированию выходных сигналов технического устройства. Именно такой обобщенный механизм является наиболее удачным для описания тех факторов и процессов, которые формируют поведение любой цифровой системы. Мы немного расширим эту обобщенную модель тем, что введем такое понятие, как “время”. Одним словом многие цифровые системы привязаны ко времени. Под временем мы понимаем определенные сигналы тактовых частот, которые в простейших случаях формируются кварцевым тактовым генератором, либо могут быть получены от смежных блоков в составе одного устройства / отдельных независимых устройств. Одним словом в зависимых от времени цифровых системах, даже при явном отсутствии изменения внешних входных воздействий на нее, внутри непрерывно происходят различные поведенческие процессы, которые внешне могут никак не проявляться и не приводить к каким-либо внешним откликам данной системы. Опорная тактовая частота, которую формирует кварцевый тактовый генератор, представляет собой непрерывную последовательность сменяющих друг друга сигналов низкого и высокого логического уровня (логических нулей и единиц), с заданной скоростью в герцах и соответствующим этой скорости периодом. Внутри таких систем, каждый раз, когда сигнал тактовой частоты меняет свое состояние с логического нуля на логическую единицу (либо с логической единицы на логический ноль - зависит от идей разработчика), происходит инициализация каких-то внутренних событий. Например, по тактовой частоте будет происходить непрерывный опрос датчика температуры на случай перегрева или уменьшение/увеличение какого-либо внутреннего счетчика для реализации таймера и т.д. При решении задач, так или иначе связанных с проверкой работоспособности системы или реверс-инжинирингом, часто возникает задача генерации собственного времени, когда логический уровень тактового сигнала меняет свое состояние в определенное время и с заданной временной задержкой. Например, тактовый сигнал меняет

свое состояние каждый 10 секунд, для того чтобы мы могли снимать состояние исследуемой системы с помощью внешних контрольно-измерительных приборов. В процессе разработки наших собственных приборов и устройств мы широко применяем и активно используем программируемые логические интегральные схемы (как для целей и задач прототипирования и создания макетных образцов, так и для решения узкоспециализированных целей и задач в финальных версиях устройств и приборов). Данные программируемые микросхемы зарекомендовали себя как надежные и долговечные средства для решения инженерных задач /3/, при проектировании приборов и устройств. Нами было принято решение частично автоматизировать ряд повторяющихся рутинных действий, которые мы ежедневно выполняем в своей рабочей практике. В результате нами был разработан параметризованный автогенератор программного кода формирователя импульсов на языках описания аппаратуры Verilog/VHDL, который формирует последовательность импульсов по заданным параметрам. Весь процесс работы изложен ниже в дальнейших подразделах данной статьи. Математическая составляющая Прежде чем приступить непосредственно к описанию разработки генератора программного кода, необходимо дать краткое математической описание механизма формирования задержек в цифровой электронике /4/. Данное описание подходит не только для реализации на программируемых логических интегральных схемах, но также и для микроконтроллеров. Как правило, в программируемых логических интегральных схемах, микроконтроллерах, а также любой другой программируемой цифровой электронике, чтобы получить ту или иную задержку, применяют счетчики, которые увеличивают / уменьшают свое значение по фронту (положительному сигналу) тактовой частоты. Сигнал тактовой частоты может быть получен как с помощью внутреннего генератора тактовой частоты (который встроен в корпус микросхемы программируемой логической интегральной схемы / микроконтроллера), так и с внешнего кварцевого генератора (кварцевого резонатора). Внешний кварцевый резонатор может быть выполнен в виде отдельной микросхемы, которая может быть заложена в разрабатываемом устройстве. Мы можем посчитать задержку по следующей формуле: , (1) где задержка (измеряемая в секундах) , - размер счетчика , - период от тактовой частоты (измеряемый в секундах). Зная значение тактовой частоты, мы можем посчитать ее период: (2) где тактовая частота (измеряемая в герцах). Изменяя размер счетчика в формуле (1), мы можем в результате получить необходимую задержку. Введем дополнительное значение - число делитель для счетчика, для того, чтобы мы могли вычислить размер счетчика: , (3) где - число делитель для счетчика. Подставив формулу (3) в формулу (1), мы получим следующую формулу: , (4) Из формулы (4), мы можем получить конечную формулу, с помощью которой, зная значение тактовой частоты и задержки (которая нам необходима), а также вычислив по формуле (2) период от заданной тактовой

частоты, мы можем получить значение числа делителя для счетчика: , (5) Подставив значение числа делителя для счетчика, полученное по формуле (5) в формулу (3), мы получим размер счетчика, который необходимо использовать на программируемом цифровом устройстве, для того, чтобы при заданной тактовой частоте получить необходимую для нас задержку. Разработка программного обеспечения В процессе разработки параметризованного автогенератора программного кода формирователя импульсов с заданной задержкой на языках описания аппаратуры Verilog/VHDL нами было написано программное обеспечение на языке программирования C# под платформу .NET для запуска на операционных системах от Microsoft. Скриншот данного приложения представлен на рисунке 1. Мы не ставили перед собой задачу разработки сложного графического интерфейса для нашего приложения. Именно по этой причине приложение имеет упрощенный графический интерфейс пользователя, реализованный стандартными средствами Windows Forms. Рис. 1 - Скриншот приложения параметризованного автогенератора программного кода формирователя импульсов на языках описания аппаратуры Verilog/VHDL Обобщенный алгоритм работы приложения, изображенного на рис. 1, представлен на рис. 2. Рис. 2 - Обобщенный алгоритм работы приложения параметризованного автогенератора программного кода формирователя импульсов на языках описания аппаратуры Verilog/VHDL Как видно из рисунка 2, после ввода базовых параметров (входная частота, язык описания аппаратуры, папка для сохранения автоматически сгенерированного кода), пользователь выбирает один из двух предложенных вариантов формирователя импульсов для автоматической генерации кода (формирователь единичных импульсов / формирователь последовательности импульсов). В случае если пользователь выбрал автоматическую генерацию формирователя единичных импульсов, он должен ввести специальные параметры для данного варианта (задержку, длительность) и нажать кнопку для генерации кода. Пример работы автоматически сгенерированного кода для данного случая после прошивки программируемой логической интегральной схемы показан на рисунке 3. Рис. 3 - Пример работы автоматически сгенерированного кода для случая “Формирователя единичных импульсов” Если же пользователь выбрал автоматическую генерацию формирователя последовательности импульсов, он должен ввести специальные параметры для данного варианта (количество импульсов в последовательности, параметры для каждого импульса (длительность и задержка)) и нажать кнопку для генерации кода. Пример работы автоматически сгенерированного кода для данного случая после прошивки программируемой логической интегральной схемы показан на рисунке 4. Рис. 4 - Пример работы автоматически сгенерированного кода для случая “Формирователя последовательности импульсов” Проверка работоспособности Для проверки работоспособности автоматически сгенерированного кода

формирователей импульсов был сгенерирован код на обоих языках описания аппаратуры для двух вариантов формирователей и были заданы следующие параметры: для “Формирователя единичных импульсов”: о входная частота 50 МГц; о задержка между единичными импульсами 5с; о длительность единичных импульсов 5с; для “Формирователя последовательности импульсов”: о количество импульсов в последовательности 5; о для 1-го импульса - длительность 5с и задержка после 10с; о для 2-го импульса - длительность 5с и задержка после 20с; о для 3-го импульса - длительность 5с и задержка после 30с; о для 4-го импульса - длительность 5с и задержка после 40с; о для 5-го импульса - длительность 5с и задержка после 50с; После коды двух вариантов формирователей проверялись в проектах, которые были созданы в системах автоматизированного проектирования Altera Quartus II и Xilinx ISE. Автоматически сгенерированные модули формирователей импульсов подключались к светодиодам, которые загорались при появлении единичного импульса в соответствии заданным временным параметрам. Длительности и задержки в секундах были выбраны специально, чтобы можно было заметить визуально заданные временные промежутки на светодиодах. Была проведена проверка на программируемых логических интегральных схемах Cyclone II EP2C20F484C7 (данная микросхема входит в состав отладочной платы Altera DE1 Development & Education Board) и Xilinx Spartan-3A XC3S700A-FG484 (данная микросхема входит в состав отладочной платы Xilinx Spartan-3A FPGA Starter Kit). Заключение В результате проведенной работы было создано программное обеспечение генератора программного кода на языках описания аппаратуры Verilog/VHDL, которое создает программный код модуля формирования импульсов по заданным пользователем параметрам. Полученный код на языках описания аппаратуры использовался для тестирования и отладки прототипа программно-аппаратного комплекса для управления мощными источниками токов /5/. В будущем основные алгоритмы автоматической генерации программного кода для использования в “прошивках” программируемых логических интегральных схем будут использованы в более сложных проектах, таких как: параметризованный автогенератор кода на языках описания аппаратуры Verilog/VHDL для модуля асинхронного приемо-передатчика интерфейса RS-232 (UART); параметризованный автогенератор кода на языках описания аппаратуры Verilog/VHDL для модуля контроллера SPI интерфейса; параметризованный автогенератор кода на языках описания аппаратуры Verilog/VHDL для модуля работы с меню аппаратных графических интерфейсов низкого уровня;