ПЛИС фирмы Gowin Semiconductor
Часть 3. Инструментальные средства программной поддержки и разработка аппаратной части проектов ПЛИС GOWIN

Павел Редькин

Предлагаемая статья содержит сведения по инструментальным средствам программной поддержки ПЛИС фирмы GOWIN Semiconductor, предоставляемым как самой фирмой, так и сторонними производителями. В статье рассмотрен порядок создания проектов в системе разработки-отладки аппаратной конфигурации ПЛИС GOWIN, порядок и особенности программирования ПЛИС. Статья предназначена для разработчиков электронной аппаратуры на ПЛИС и студентов специальностей, связанных с цифровой электроникой.

1. Средства программной поддержки ПЛИС GOWIN от производителя

В качестве основных средств программной поддержки семейств ПЛИС своего производства GOWIN Semiconductor предоставляет две бесплатные программные среды для разработки-отладки приложений.

Программная среда GOWIN FPGA Designer представляет собой систему разработки-отладки приложений (IDE) на базе ПЛИС GOWIN и интегрирует в себе возможности синтеза, отображения, размещения, маршрутизации и генерации битового потока (конфигурации массива программируемой логики), а также инструменты программирования ПЛИС GOWIN, встроенный логический анализатор и калькулятор мощности. Среда GOWIN FPGA Designer поддерживает написание исходных кодов проектов ПЛИС на языках описания аппаратных средств VHDL и Verilog HDL.

Программная среда GOWIN MCU Designer представляет собой систему разработки-отладки приложений на основе программных и аппаратных процессорных IP-ядер на базе ПЛИС GOWIN и интегрирует в себе возможности компиляции, линковки и отладки встроенных программ для этих ядер. Среда GOWIN MCU Designer поддерживает написание исходных кодов программ для процессорных ядер ПЛИС GOWIN на языке C.

Отладка кода встроенного ПО для программных и аппаратных процессорных IP-ядер ПЛИС GOWIN в среде IDE GOWIN MCU Designer осуществляется с помощью сторонних аппаратных отладчиков, подключаемых к целевой ПЛИС через интерфейс JTAG.

Дистрибутивы IDE GOWIN FPGA Designer, предназначенные для работы под ОС Windows и под ОС Linux, могут быть бесплатно загружены с сайта www.gowinsemi.com. Для инсталляции этой IDE необходима лицензия, которая также предоставляется бесплатно при запросе через указанный сайт. В случае возникновения проблем с её получением рекомендуется обратиться за помощью к российскому дистрибьютору продукции GOWIN (АО «Восток», г. Санкт-Петербург) [1].

Дистрибутив IDE GOWIN MCU Designer, предназначенный для работы под ОС Windows (работа IDE под ОС Linux не поддерживается), также может быть бесплатно загружен с сайта www.gowinsemi.com. Для инсталляции IDE GOWIN MCU Designer необходима бесплатная лицензия, порядок получения которой аналогичен порядку получения лицензии для IDE GOWIN FPGA Designer.

Загрузка конфигурации в ПЛИС GOWIN осуществляется только из IDE GOWIN FPGA Designer с помощью встроенного программного инструмента Gowin Programmer, взаимодействующего с аппаратной частью целевой ПЛИС через интерфейс USB ПК.

2. Средства программной поддержки ПЛИС GOWIN от сторонних производителей

Создание приложений на базе программных и аппаратных процессорных IP-ядер с архитектурой ARM Cortex-M на базе ПЛИС GOWIN также поддерживается широко распространённой средой IDE ARM Keil MDK версии V5.26 и выше от фирмы Keil. Поддержка ПЛИС GOWIN в ARM Keil MDK охватывает не только принципиальную возможность разработки приложений для IP-ядер с архитектурой ARM на базе ПЛИС GOWIN с поддержкой режима отладки кода, но и наличие примеров проектов для IDE ARM Keil MDK, распространяемых GOWIN наряду с примерами проектов для IDE GOWIN MCU Designer. Фирменные руководства от GOWIN по разработке программных приложений на базе ПЛИС с IP-ядрами с архитектурой ARM, как правило, имеют по два раздела: разработка в IDE ARM Keil MDK и разработка в IDE GOWIN MCU Designer.

Отладка кода встроенного ПО для программных и аппаратных процессорных IP-ядер ПЛИС GOWIN в среде IDE ARM Keil MDK осуществляется с помощью сторонних аппаратных отладчиков, подключаемых к целевой ПЛИС через интерфейс JTAG.

3. Порядок разработки проекта ПЛИС GOWIN

3.1. Общая идеология разработки

Разработка проекта ПЛИС GOWNIN начинается с создания нового проекта в среде IDE GOWIN FPGA Designer. Данный проект выступает при разработке проекта ПЛИС в качестве головного. Порядок создания нового проекта в среде IDE GOWIN FPGA Designer подробно изложен в [2].

В случае если предполагается включение в проект ПЛИС программных или аппаратных процессорных IP-ядер, отдельные проекты для их встроенного ПО необходимо создать и откомпилировать в одной из сред IDE GOWIN MCU Designer или IDE ARM Keil MDK независимо от головного проекта ПЛИС. При сборке головного проекта ПЛИС выходные данные указанных проектов передаются в него в качестве

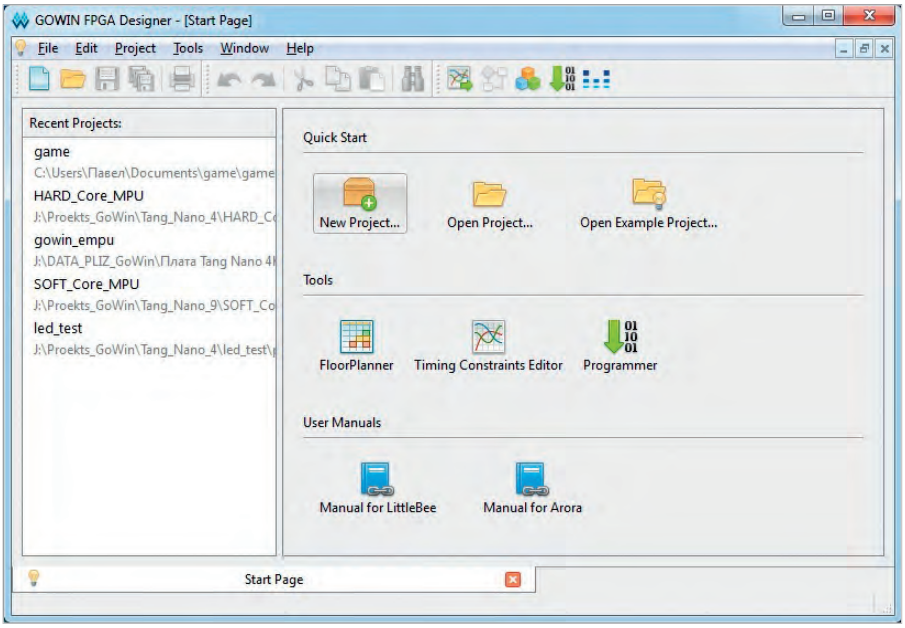

Рис. 1. Стартовое окно «Start Page» в GOWIN FPGA Designer

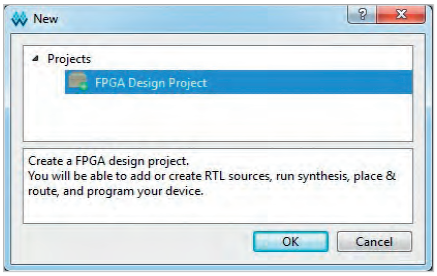

Рис. 2. Окно вариантов нового проекта

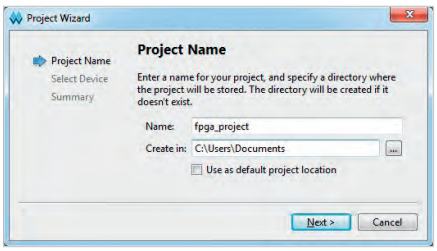

Рис. 3. Окно задания имени нового проекта и пути к нему

входных данных. В следующей части статьи будет описано, каким образом это делается.

В случае если включение в проект ПЛИС процессорных IP-ядер не предполагается, данный проект ПЛИС разрабатывается только в среде IDE GOWIN FPGA Designer и никаких входных данных из других проектов не использует.

3.2. Создание проекта ПЛИС в среде IDE GOWIN FPGA Designer

Порядок создания в среде IDE GOWIN FPGA Designer проекта ПЛИС проиллюстрируем на примере отладочной платы TangNano 9K с ПЛИС GW1NR-9. Для наглядности на начальном этапе создадим проект ПЛИС без программного IP-ядра. В дальнейшем программное IP-ядро можно будет добавить в готовый проект.

После запуска GOWIN FPGA Designer по умолчанию откроется стартовое окно Start Page, в котором нужно выбрать иконку «New Project...» и кликнуть по ней, как показано на рис. 1. Далее откроется окно, предлагающее выбрать варианты нового проекта (рис. 2). Из вариантов пока предлагается только один – «FPGA Design Project». Кликаем на кнопке «OK» в этом окне, после чего открывается окно, в котором необходимо задать имя нового проекта и путь к нему на диске ПК (рис. 3). Задаём имя и путь, после чего кликаем на кнопке «Next». Открывается окно задания целевой ПЛИС, показанное на рис. 4. В этом окне нужно как можно точнее специфицировать используемую нами ПЛИС, обращая внимание не только на её обозначение, но и на её корпусное исполнение. Для нашего проекта задаём установленную на плате Tang Nano 9K ПЛИС GW1NR-LV9QN88PC6/I5 в корпусе QFN88P, после чего кликаем на кнопке «Next». Открывается итоговое окно «Summary», в котором указаны все настройки нашего проекта (рис. 5). Если там всё верно, кликаем на кнопке «Finish». Проект создан.

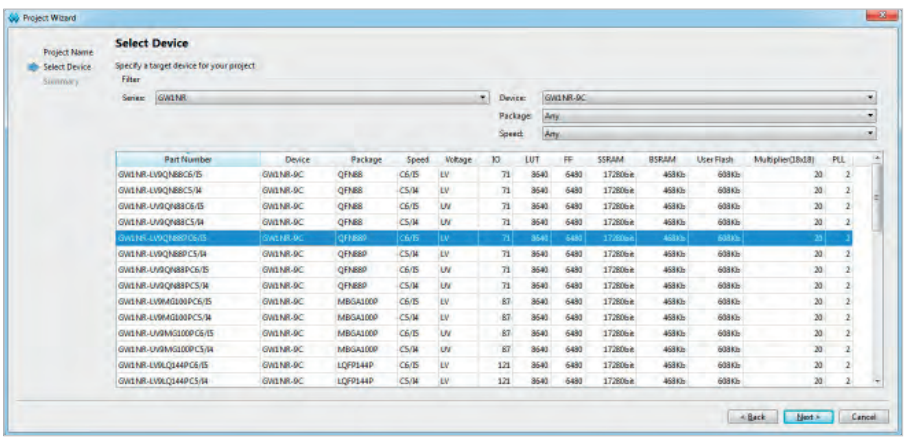

Рис. 4. Окно задания целевой ПЛИС

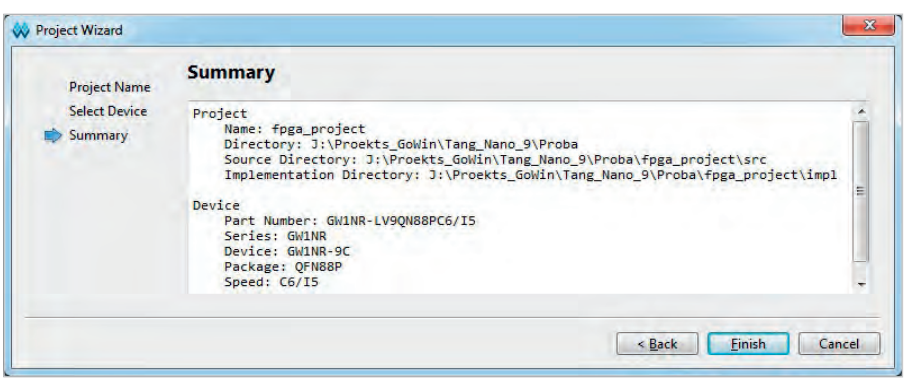

Рис. 5. Итоговое окно «Summary» с настройками проекта

Теперь необходимо наполнить созданный проект конкретным содержанием, то есть включить в него файлы исходных текстов. Для создания таких файлов «с нуля» в главном меню выбираем File > New, после чего открывается окно создания новых исходных файлов, показанное на рис. 6. Для начала

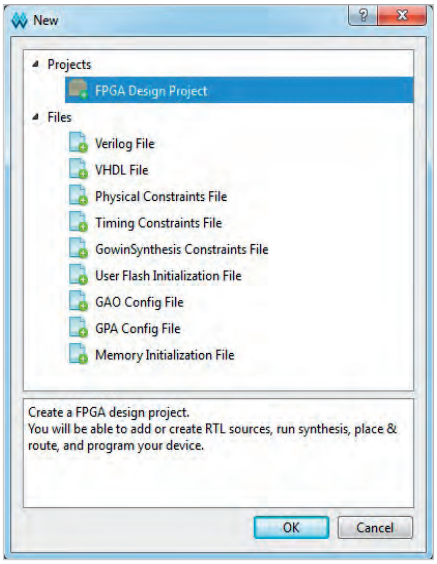

Рис. 6. Окно создания новых исходных файлов

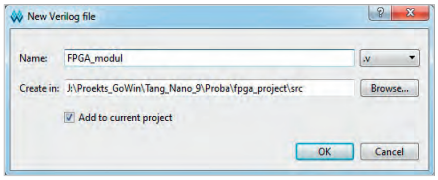

Рис. 7. Окно задания имени исходного файла и пути к нему

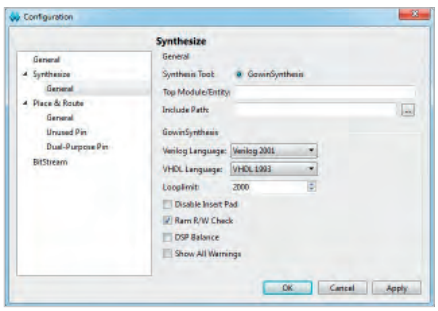

Рис. 8. Окно конфигурации проекта

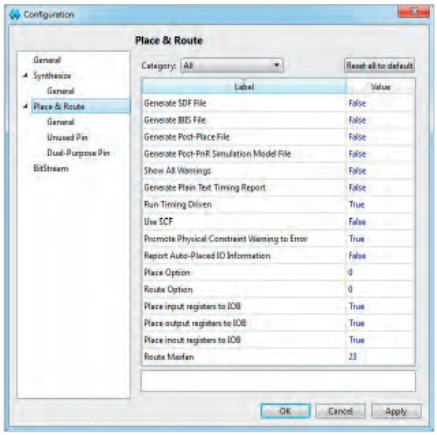

Рис. 9. Генерируемые в проекте выходные файлы

создадим файл модуля верхнего уровня проекта на языке Verilog HDL, выбрав в этом окне позицию Verilog File, после чего нам будет предложено задать имя файла и путь к нему, как показано на рис. 7. Задаём имя файла FPGA_modul. По умолчанию создаваемый файл будет находиться в автоматически созданном каталоге src нашего проекта и будет автоматически включён в его состав (установленная галочка в позиции «Add to current project»). Кликаем на кнопке «OK», после чего на вкладке Design в окне обозрений проекта появляется позиция созданного файла FPGA_modul с расширением «.v», а в окне просмотра файлов открывается его пока ещё пустое содержимое.

Теперь настроим конфигурацию нашего проекта ПЛИС, выбрав в главном меню Project > Configuration. Откроется окно Configuration (рис. 8), в котором необходимо выбрать позицию General в группе Synthesize и задать используемые в проекте языковые стандарты для языков описания аппаратных средств Verilog HDL и VHDL, как показано на рисунке.

Далее выбираем позицию General в группе Place & Route и задаём генерируемые в проекте выходные файлы (рис. 9).

Далее выбираем позицию Unused Pin в группе Place & Route и задаём подтяжку не используемых в проекте линий GPIO ПЛИС к плюсу источника питания, как показано на рис. 10.

Далее выбираем позицию Dual-Purpose Pin в группе Place & Route и задаём использование выводов двойного назначения (рис. 11). Каждая позиция в этом окне позволяет разрешить или запретить использование в качестве линий GPIO выводов ПЛИС двойного назначения, то есть таких, которые, помимо функции GPIO, могут использоваться в качестве линий одного из поддерживаемых ПЛИС интерфейсов (JTAG, SSPI, MSPI, I^2C) или для какой-либо служебной функции (READY, DONE, RECONFIG_N).

Наконец, выбираем позицию BitStream и задаём параметры генерации файла битового потока, как показано на рис. 12. К ним относятся проверка контрольной суммы CRC, сжатие файла битового потока, шифрование файла битового потока, защита файла битового потока от считывания конфигурации, интерфейс фонового программирования, скорость загрузки конфигурации ПЛИС и формат файла битового потока. Под фоновым программированием здесь понимается возможность повторно программировать встроенную Flash-память конфигурации ПЛИС без прерывания текущего управления FPGA. Значение в поле фонового программирования (Background Programming) задаётся в зависимости от модели целевой ПЛИС в соответствии с табл. 1.

Значение OFF в поле Background Programming задаётся в случае, если фоновое программирование отключено. После задания значения OFF в качестве интерфейса фонового программирования для ПЛИС GW2AN-18X или GW2AN-9X задание опции «Use MSPI as regular IO» на вкладке Dual-Purpose Pin становится неактивным и невозможным.

Значение JTAG в поле Background Programming предполагает фоновое

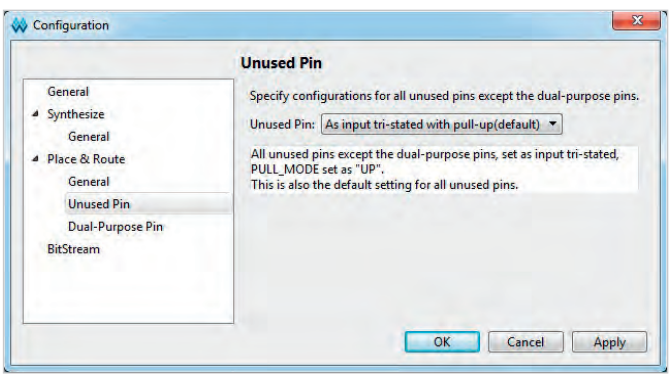

Рис. 10. Подтяжка не используемых в проекте линий GPIO ПЛИС

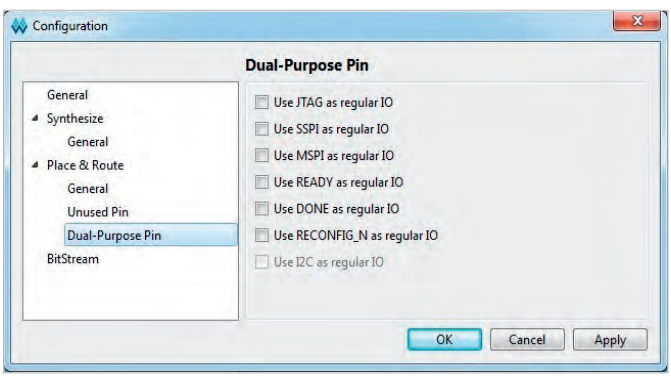

Рис. 11. Использование выводов двойного назначения ПЛИС

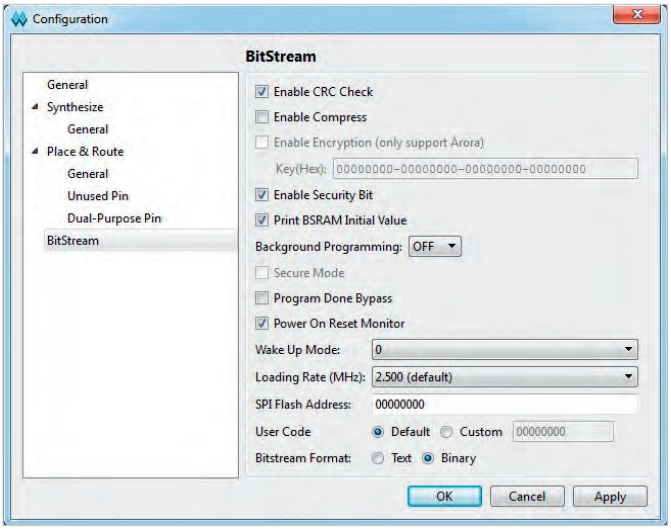

Рис. 12. Параметры генерации файла битового потока

Таблица 1. Значение в поле Background Programming параметров битового потока

Модель ПЛИС GOWIN	Возможные значения поля Background Programming; значение по умолчанию
GW1N-1P5/GW1N-2/GW1NR-2, GW1N-4/GW1N-4B, GW1NR-4B/GW1NR-4D/GW1NRF-4B, GW1NS-4/GW1NSR-4, GW1N-9/GW1N-9C, GW1NR-9/GW1NR-9C, GW1NZ-1/GW1NZ-1C	OFF, JTAG; OFF по умолчанию
GW1N-1P5B/GW1N-2B/GW1NR-2B	OFF, JTAG, I2C; OFF по умолчанию
GW2AN-18X/GW2AN-9X	OFF, Internal, I2C/JTAG/SSPI/QSSPI; OFF по умолчанию

программирование через JTAG при условии, что для линий JTAG в окне Dual-Purpose Pin не задано использование в качестве линий GPIO. Значение I^2C в поле Background Programming предполагает фоновое программирование через I^2C, причём при выборе этого значения на вкладке BitStream появляется дополнительное поле выбора адреса ведомого устройства (ПЛИС) на шине I^2C в диапазоне от 0x00 до 0x7F. После задания I^2C в качестве интерфейса фонового программирования выбор опции «Use JTAG as regular IO» на вкладке Dual-Purpose Pin становится неактивным и невозможным.

Значение Internal в поле Background Programming предполагает фоновое программирование через внутреннюю логику ПЛИС. После задания значения Internal в качестве интерфейса фонового программирования задание опции «Use MSPI as regular IO» на вкладке Dual-Purpose Pin становится неактивным и невозможным.

После задания значения I^2C/JTAG/SSPI/QSSPI в качестве интерфейсов фонового программирования выбор опции «Use MSPI as regular IO» на вкладке Dual-Purpose Pin становится неактивным и невозможным. Также при выборе этого значения на вкладке BitStream появляется дополнительное поле «HOTBOOT», детали использования которого описаны в [2].

Необходимо заметить, что фоновое программирование не может быть осуществлено в случае, если в составе ПЛИС имеется пользовательская Flash-память (User Flash).

Опция защиты файла битового потока от считывания «Enable Secure Bit» позволяет осуществить только однократное программирование ПЛИС. Эта опция поддерживается лишь в ПЛИС GW1NSE-2C и GW1NSER-4C, для которых она по умолчанию отключена.

Опция «Program Done Bypass» устанавливает возможность начать новое программирование ПЛИС до завершения текущего программирования.

Опция монитора сброса при включении питания «Power On Reset Monitor» позволяет произвести автоматический сброс ПЛИС при включении питания. Этот сброс подразумевает обнуление всех ячеек встроенной памяти RAM, установку линий IO в состояние с высоким выходным сопротивлением со слабой внутренней подтяжкой к плюсу питания до окончания конфигурирования и инициализации ПЛИС. По умолчанию эта опция отключена.

Опция задания способа «пробуждения» (Wake Up) ПЛИС позволяет задать один из двух вариантов: 0 и 1. В случае задания варианта 0 ножка DONE должна быть для «пробуждения» ПЛИС подтянута к плюсу питания или к общему проводу. Когда задан вариант 1 и ножка DONE подтянута к плюсу питания, то загрузка и выполнение конфигурации ПЛИС производятся в нормальном режиме. Если же для варианта 1 ножка DONE подтянута к общему проводу, то загрузка конфигурации ПЛИС производится в нормальном режиме, а для «пробуждения» ПЛИС ножка DONE должна быть подтянута к плюсу питания и соединена с линией TCK, на которой действует импульсный сигнал, который и «пробуждает» ПЛИС. По умолчанию задан вариант «пробуждения» 0.

Опция «Loading Rate» задаёт скорость загрузки битового потока из Flash в SRAM в режимах AutoBoot и MSPI. По умолчанию эта скорость соответствует частоте 2,5 МГц. Для уточнения подробностей по заданию скорости загрузки битового потока для различных семейств и линеек ПЛИС GOWIN рекомендуется обратиться к источнику [2].

Опция «SPI Flash Address» позволяет задать адрес во внешней памяти SPI, с которого будет осуществляться загрузка битового потока конфигурации ПЛИС. По умолчанию этот адрес равен 00000000. Для уточнения подробностей по загрузке битового потока конфигурации ПЛИС из внешней памяти рекомендуется обратиться к руководству по программированию ПЛИС GOWIN [3].

Опция «User Code» позволяет задать для пользовательского кода битового потока конфигурации ПЛИС некое константное значение, которое будет проверено программатором при загрузке в ПЛИС файла битового потока. В случае включения указанной опции это константное значение по умолчанию равно 00000000.

Опция «Bitstream Format» позволяет задать формат выходного файла битового потока: текстовый или двоичный. По умолчанию задан двоичный формат. При выборе текстового формата генерируется файл с расширением *.fs в обычном текстовом формате, при выборе двоичного формата генерируются файлы с расширениями *.fs, *.bin и *.binx. Файлы с расширениями *.bin и *.binx представляют собой файлы битового потока в двоичном формате, причём файл с расширением *.binx содержит информацию об аннотации заголовка. Файл с расширением *.bin

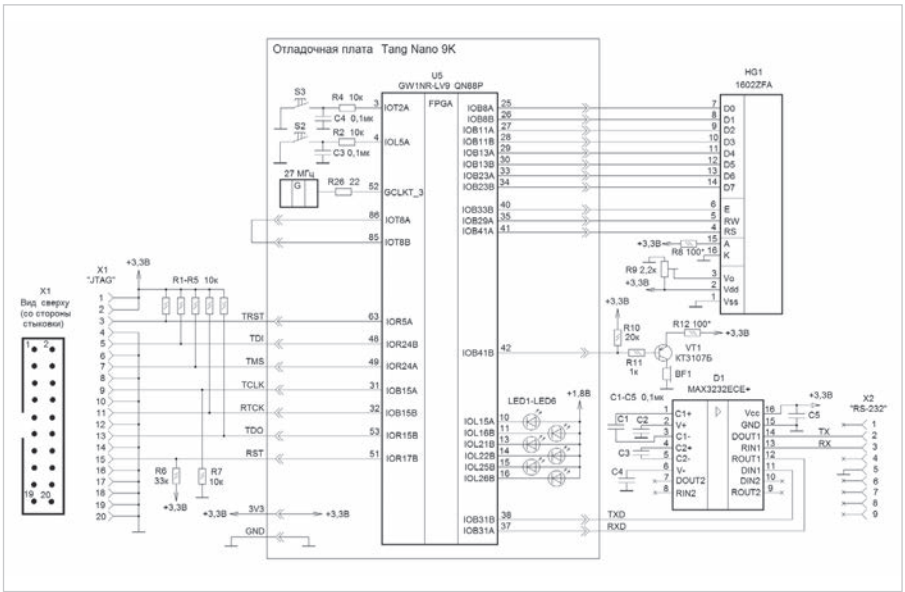

Рис. 13. Принципиальная схема макета с платой Tang Nano 9K

не содержит информации об аннотации заголовка.

Задаём параметры генерации файла битового потока, как показано на рис. 12, и кликаем по кнопке «OK». Наш проект настроен.

Теперь заполним содержимое файла модуля верхнего уровня нашего проекта FPGA_modul.v каким-либо исходным кодом на языке Verilog HDL. Для примера в исходном коде нашего проекта реализуем несколько контроллеров, обслуживающих вывод данных в символьный двухстрочный жидкокристаллический индикатор (ЖКИ), совместимый со стандартом HD44780. Принципиальная схема макета с подключением такого ЖКИ к ПЛИС платы Tang Nano 9K показана на рис. 13. Помимо символьного ЖКИ (на схеме – HG1), на рисунке показано подключение к ПЛИС штатных кнопок S2, S3, светодиодов LED1–LED6 и генератора синхросигнала 27 МГц платы Tang Nano 9K, а также внешние для платы Tang Nano 9K узлы: преобразователь уровней «UART-RS-232» для обеспечения обмена ПЛИС с внешними устройствами по интерфейсу RS-232 (на схеме – ИМС D1, разъём X2 типа DB-9), узел генерации звуковых сигналов (на схеме – транзистор VT1, звукоизлучатель BF1 электромагнитного типа). Также на схеме несколько выводов GPIO ПЛИС подтянуты к плюсу питания +3,3 В и к общему проводу с помощью внешних резисторов R1–R7 и соединены с контактами внешнего по отношению к плате Tang Nano 9K разъёма форм-фактора интерфейса JTAG (на схеме – разъём X1). Этот разъём планируется использовать в целях отладки встроенного ПО программного процессорного IP-ядра при его добавлении к проекту ПЛИС. Прочие штатные элементы платы Tang Nano 9K, в том числе штатные источники питания, на схеме условно не показаны. Линии GPIO ПЛИС выводов 85, 86 соединены между собой внешней короткозамкнутой перемычкой, назначение которой будет объяснено позднее. Кнопка S2 используется в проекте ПЛИС в качестве кнопки сброса.

Внешний вид макета показан на рис. 14.

Теперь создадим файл физической привязки линий ввода-вывода модуля верхнего уровня к ножкам нашей ПЛИС согласно схеме рис. 13. Снова открываем окно создания новых исходных файлов, показанное на рис. 6, выбрав в этом окне позицию Physical Constraints File. Аналогичным образом задаём имя (FPGA_constr) и путь к этому файлу (каталог/src нашего проекта), кликаем на кнопке «OK», после чего заполняем содержимое нашего файла описанием линий ввода-вывода. На первоначальном этапе разработки проекта этот файл создаётся вручную. Указываем для всех цепей входов и выходов нашего проекта ПЛИС номера выводов ПЛИС и прочие параметры: наличие подтяжки к плюсу питания или к общему проводу, максимальный ток для выходов, тип логики для входов в приведённом ниже формате. В каждой такой записи за словом «IO_LOC» следует имя цепи проекта ПЛИС в кавычках, а затем задаваемый для этой цепи номер ножки ПЛИС. За словом «IO_PORT» следует имя цепи

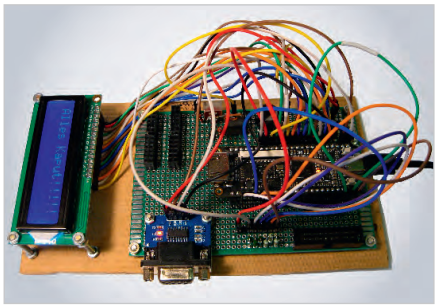

Рис. 14. Внешний вид макета с платой Tang Nano 9K

проекта ПЛИС в кавычках, а затем задаваемые для этой цепи параметры (тип логики, наличие и тип внутренних подтяжек, максимальный ток в мА для выхода), например:

```
// Вход синхросигнала, номер вывода GPIO ПЛИС - 52
IO_LOC “sys_clk” 52;
IO_PORT “sys_clk” IO_TYPE=LVCMOS33 PULL_MODE=UP;

// Выход подключения светодиода, номер вывода GPIO ПЛИС - 10
IO_LOC “led_0” 10;
IO_PORT “led_0” IO_TYPE=LVCMOS33 PULL_MODE=UP DRIVE=8;
```

Отдельные параметры в каждой такой записи разделяются пробелами, в конце каждой записи ставится точка с запятой, строка с комментариями предваряется символами «//». После заполнения сохраняем файл кликом по иконке «дискета».

Теперь добавим в наш проект ПЛИС аппаратную систему PLL в расчёте на перспективное использование в проекте процессорного IP-ядра. Для этого в главном меню выберем Tools > IP Core Generator. В открывшемся окне генератора IP-ядер в разделе Hard Module выберем подраздел CLOCK, а в нём позицию аппаратного IP-ядра PLL – «rPLL» (рис. 15). После клика по ней откроется окно настройки параметров PLL, которое следует заполнить, как показано на рис. 16. Входную частоту для нашей PLL задаём равной 27 МГц, поскольку мы будем тактировать PLL от нашего системного синхросигнала 27 МГц. Выходную частоту PLL зададим в два раза больше входной (54 МГц) в расчёте на тактирование программного процессорного IP-ядра. После клика на кнопке «OK» нам будет предложено включить PLL в наш проект. Соглашаемся. После этого в нашем проекте будут

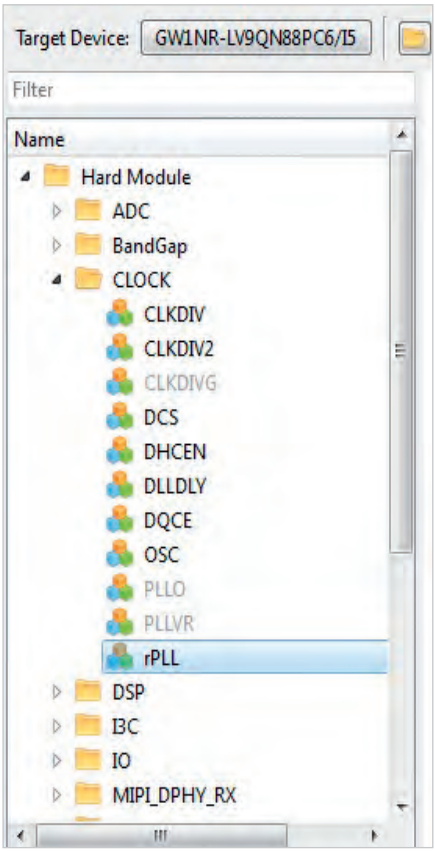

Рис. 15. Дерево генератора IP-ядер

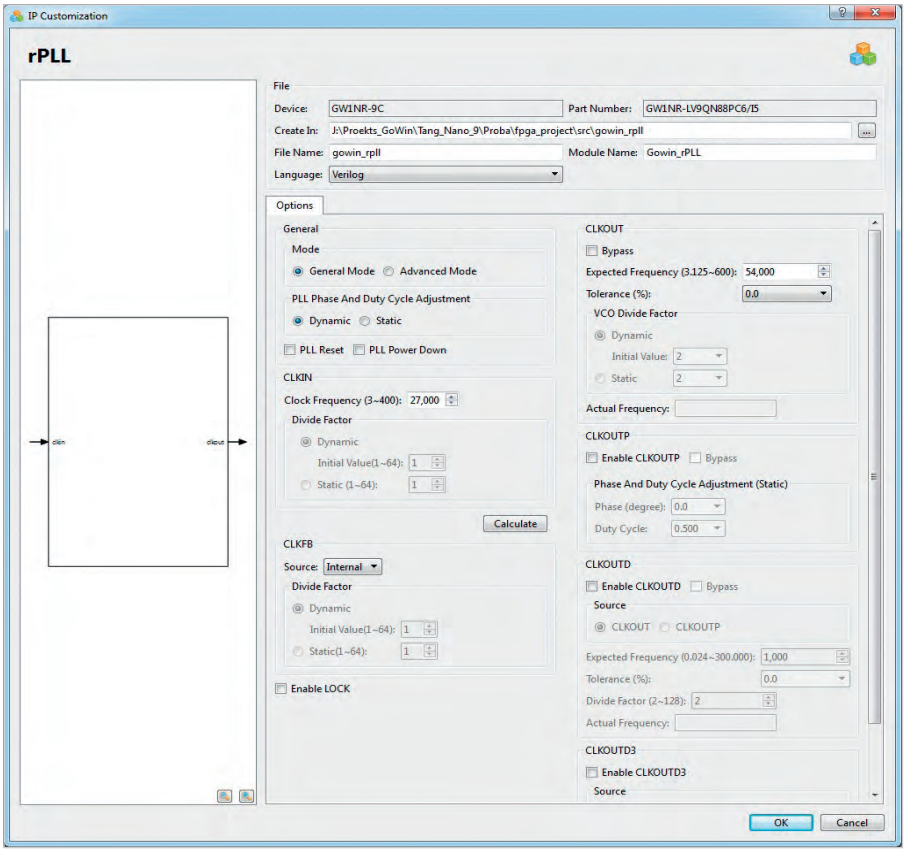

Рис. 16. Окно настройки параметров PLL

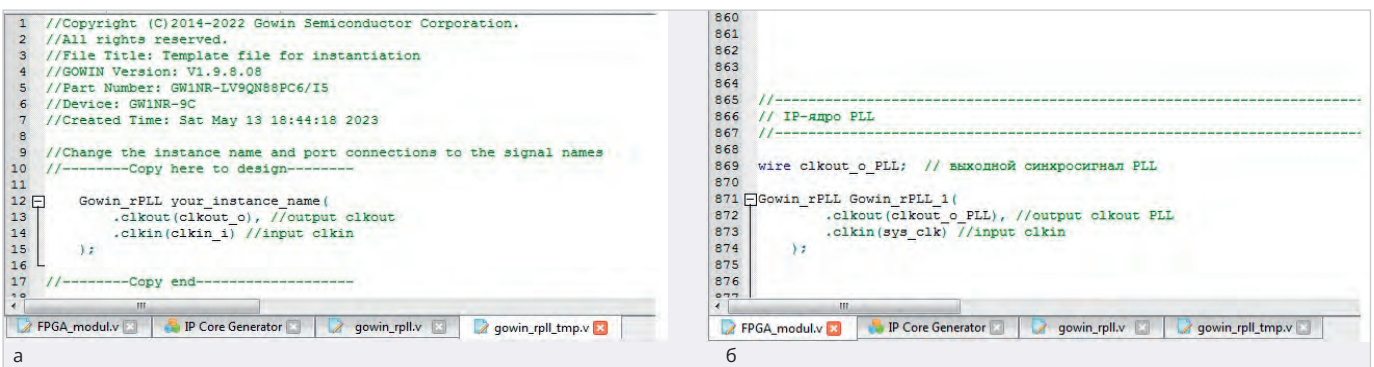

Рис. 17. а) файл шаблона модуля PLL; б) модуль PLL в качестве модуля нижнего уровня в проекте

автоматически созданы два файла: файл исходного кода PLL gowin_rpll.v и файл шаблона модуля PLL govin_rpll_tmp.v, показанный на рис. 17 (а). Из последнего необходимо скопировать исходный код и вставить его в наш файл модуля верхнего уровня проекта FPGA_modul.v, задав ему при этом уникальное имя, а также входные и выходные сигналы, как показано на рис. 17 (б).

Запускаем общую сборку проекта ПЛИС (синтез и компиляцию) кликом на иконке «Run All» на панели инструментов. Сборка проекта займёт некоторое время. В случае успешного завершения сборки все узлы на странице Process будут отмечены зелёными кружками с галочками, как показано на рис. 18. В случае наличия ошибок в проекте один или несколько узлов на странице Process будут отмечены жёлтыми кружками со знаком вопроса, а в окне Console появятся сообщения об ошибках (рис. 19). Щелчком на позиции сообщения об ошибке можно переместиться на ошибку в исходном коде.

В случае безошибочного завершения сборки можно отредактировать распределение сигналов проекта по выводам целевой ПЛИС уже не вручную, а с помощью специального инструмента – планировщика FloorPlaner, для чего необходимо выбрать его позицию на странице Process (рис. 18) и кликнуть по ней. В открывшемся окне FloorPlaner нужно выбрать внизу страницу I/O Constraints, как показано на рис. 20, в которой можно отредактировать параметры всех цепей ПЛИС: номера выводов ПЛИС, наличие подтяжки к плюсу питания или общему проводу, максимальный ток для выходов, тип логики для входов и прочее. Ранее созданный нами файл с расширением FPGA_constr.cst при этом будет автоматически изменён, если он в текущий момент не открыт.

После безошибочного завершения сборки проекта ПЛИС можно переходить к загрузке файла битового потока проекта в конфигурационную память нашей целевой ПЛИС.

3.3. Загрузка проекта в целевую ПЛИС

Перед началом загрузки необходимо физически подключить отладочную

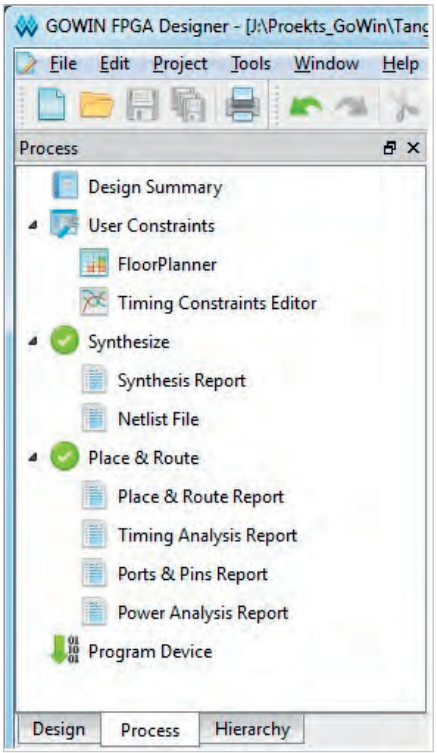

Рис. 18. Успешное завершение сборки проекта ПЛИС

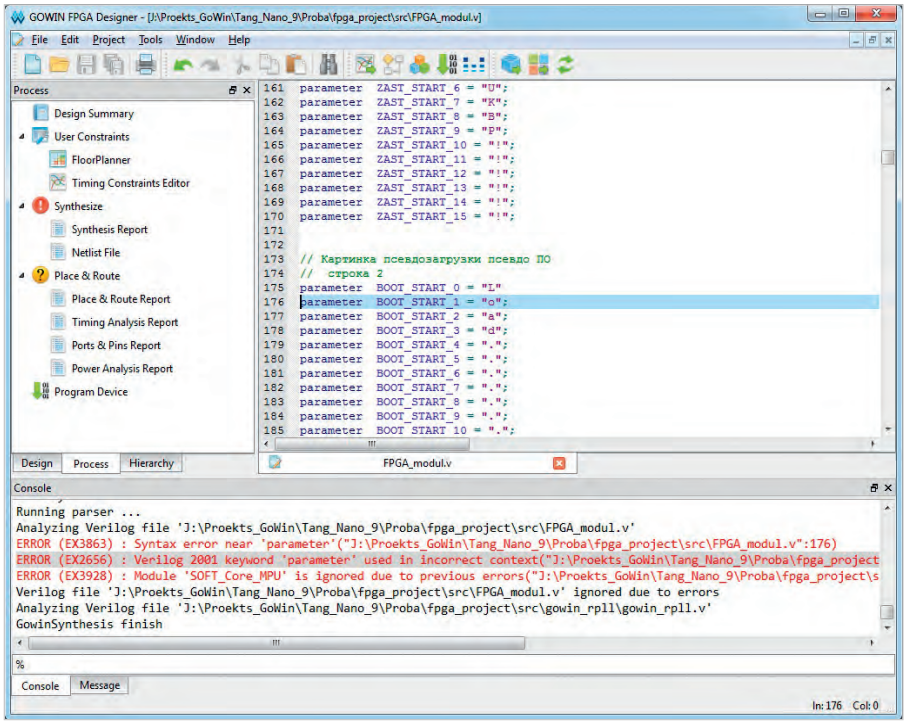

Рис. 19. Завершение сборки проекта ПЛИС с ошибкой

плату Tang Nano 9K к ПК через USB и удостовериться, что операционная система ПК обнаружила новое подключение. В случае использования платы TangNano 9 получаем составное устройство, включающее контроллер USB и COM-порт, как показано на рис. 21.

Для запуска программатора ПЛИС необходимо сделать двойной клик на позиции Program Device на странице Process (рис. 18). Откроется окно программатора Gowin Programmer и поверх него – окно настройки загрузочного кабеля Cable Setting. Если оно будет иметь вид, показанный на рис. 22, а в поле Output окна программатора Gowin Programmer появится сообщение «Cable found:..», значит, программатор Gowin Programmer обнаружил подключение отладочной платы, корректно опознал её и готов загружать в её ПЛИС данные. Если подключение отладочной платы к ПК программатором не обнаружится, то откроется окно с сообщением «No USB Cable Connection». В этом случае необходимо переподключить плату к ПК и вручную протестировать наличие подключения с помощью кнопки «Query/Detect Cable» до получения требуемого результата. Также в окне Cable Setting можно задать скорость (частоту) загрузки данных в ПЛИС.

Добившись корректного обнаружения подключённой отладочной платы, необходимо задать параметры загрузки в ПЛИС. Двойным кликом на позиции нашей ПЛИС в графе Operation открываем окно конфигурирования целевой ПЛИС (рис. 23). В поле «Access Mode» этого окна выбираем из выпадающего меню тип памяти ПЛИС для доступа, в нашем случае – «Embedded Flash Mode», то есть встроенную Flash-память, как показано на рисунке. Путь к загружаемому в ПЛИС файлу задаётся в графе FSFile. По умолчанию там уже задан путь к файлу битового потока нашего проекта fpga_project.fs. Для сохранения заданных параметров загрузки кликаем по кнопке Save. Затем запускаем загрузку в ПЛИС с помощью иконки «Program/Configure» из главного меню окна программатора Gowin Programmer. Ход загрузки конфигурации в ПЛИС отображается нарастающей шкалой, показанной на рис. 24. По завершении в поле Output окна

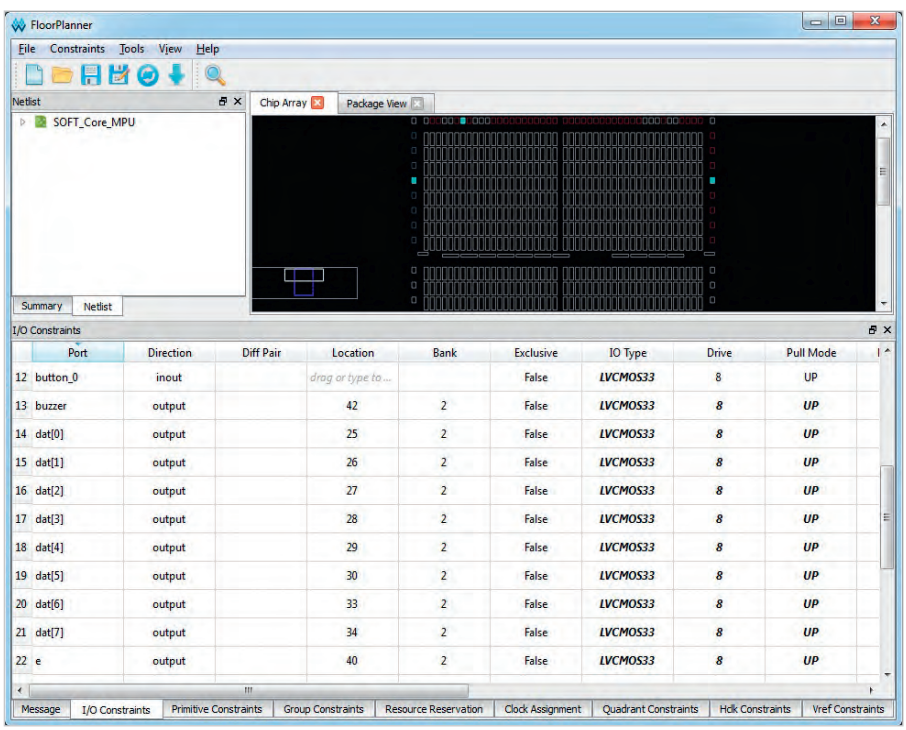

Рис. 20. Страница I/O Constraints окна «FloorPlaner»

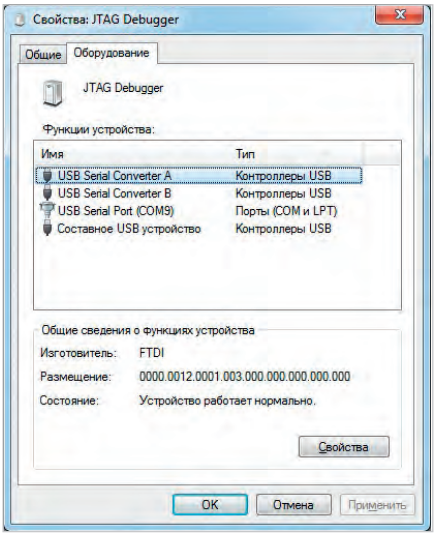

Рис. 21. Так видит подключённую плату Tang Nano 9K операционная система ПК

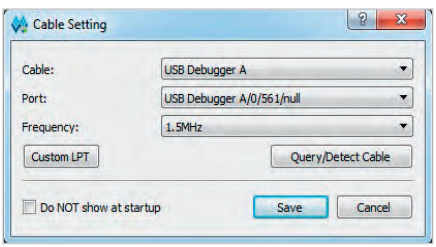

Рис. 22. Окно настройки загрузочного кабеля «Cable Setting»

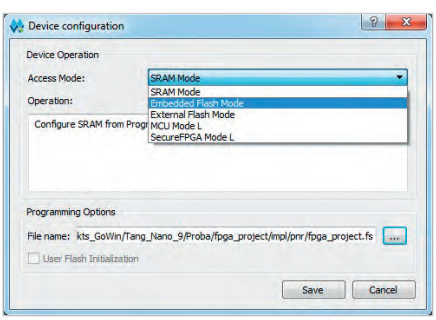

Рис. 23. Окно конфигурирования целевой ПЛИС

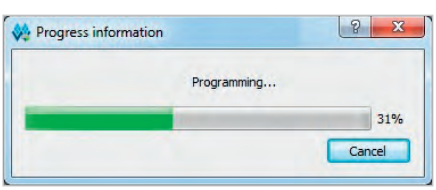

Рис. 24. Ход загрузки конфигурации в ПЛИС

программатора Gowin Programmer выдаётся сообщение о завершении загрузки «Program Finished!». После завершения загрузки проект ПЛИС сразу же стартует: на ЖКИ макета выводится начальная заставка – фраза «Hello, Friend!», затем производится «псевдозагрузка псевдо-ПО»: выводится слово «Load». Далее поле ЖКИ заполняется точками с соответствующим звуковым сопровождением. При подключении вывода 85 ПЛИС к цепи питания +3,3 В на ЖКИ выводится фраза «Things are good», а при его подключении к общему проводу – фраза «Things are bad». Внешнюю короткозамкнутую перемычку между выводами 85, 86 ПЛИС предварительно необходимо разомкнуть.

Данный проект не несёт никакой функциональной нагрузки, а просто демонстрирует работу ПЛИС. Архив проекта в виде файла Proba.zip доступен для загрузки с сайта журнала.

В следующей части статьи будут описаны порядок включения в данный проект программного процессорного IP-ядра и разработки для него встроенного ПО.

Литература

1. URL: https://vostok-electronics.ru/.
2. Gowin Software User Guide. SUG100-2.6E, 11/02/2021.
3. Gowin Programmer User Guide. SUG502-1.4E, 06/01/2022.