Продолжение. Начало в № 2'2015

Разработка программного обеспечения встраиваемых микропроцессорных систем, проектируемых на базе расширяемых вычислительных платформ семейства Zynq-7000 AP SoC фирмы Xilinx. Часть 2

Валерий ЗОТОВ
walerry@km.ru

Состав сформированного комплекта исходных модулей пакета поддержки платы BSP для проектируемой микропроцессорной системы

В процессе автоматического формирования исходных модулей создаваемого пакета поддержки платы для разрабатываемой встраиваемой системы формируется комплект файлов, в состав которого входят:

- спецификация программной платформы проектируемой микропроцессорной системы Microprocessor Software Specification (MSS);
- файл опций для программы генерации библиотек Library Generator (Libgen);
- командный файл управления процессом генерации пакета BSP.

Все перечисленные файлы записываются в папку, название которой совпадает с идентификатором формируемого пакета поддержки платы. Эта папка автоматически создается в рабочем каталоге проекта программного обеспечения для разрабатываемой встраиваемой системы. Для получения доступа к сформированному комплекту исходных модулей пакета BSP следует развернуть одноименный раздел, представленный во встроенной панели **Project Explorer** основного окна средств Xilinx SDK, как демонстрирует рис. 20 [16]. Кроме перечисленных выше файлов, в этом разделе содержится подраздел с названием **BSP Documentation**, предоставляющий возможность

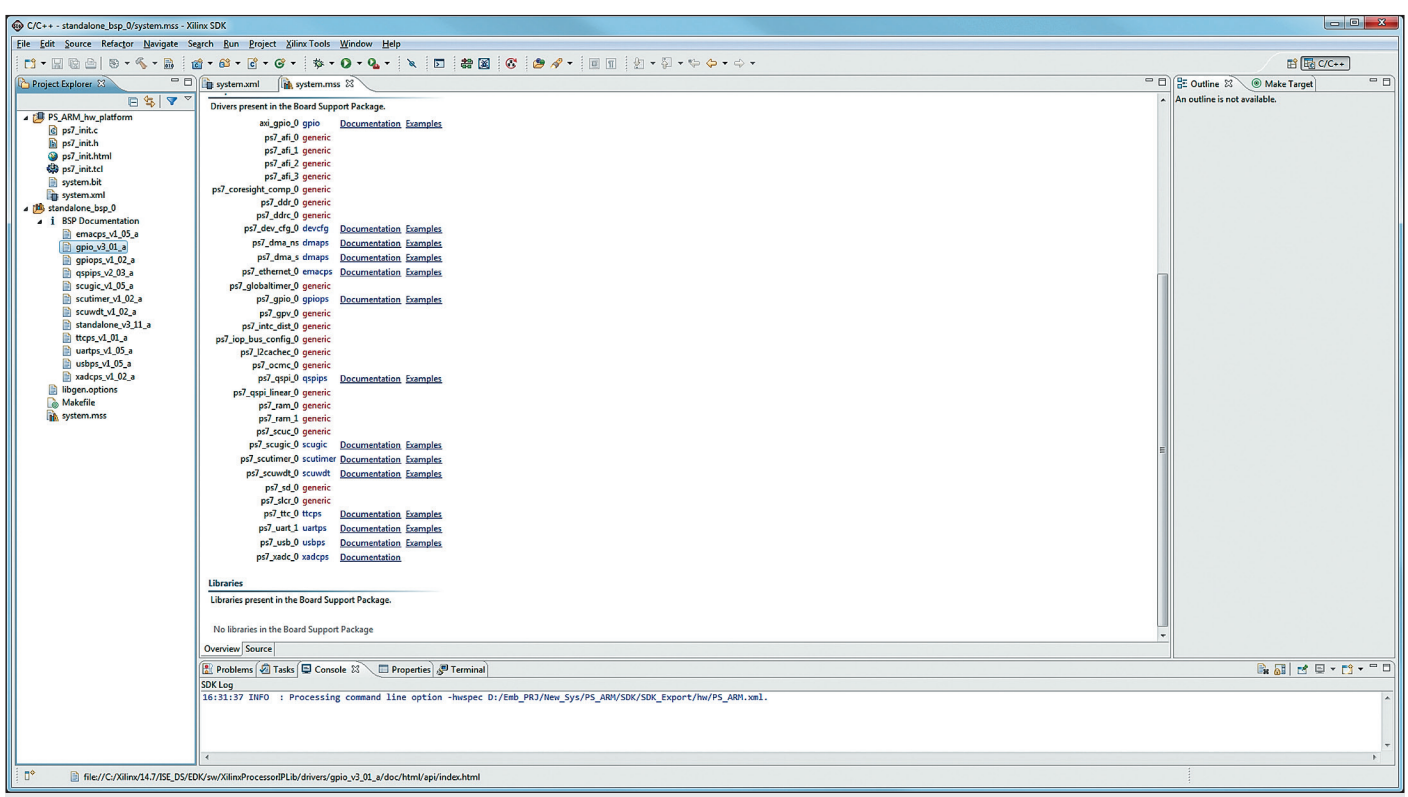

Рис. 1. Получение доступа к документации драйверов IP-компонентов аппаратной части разрабатываемой системы

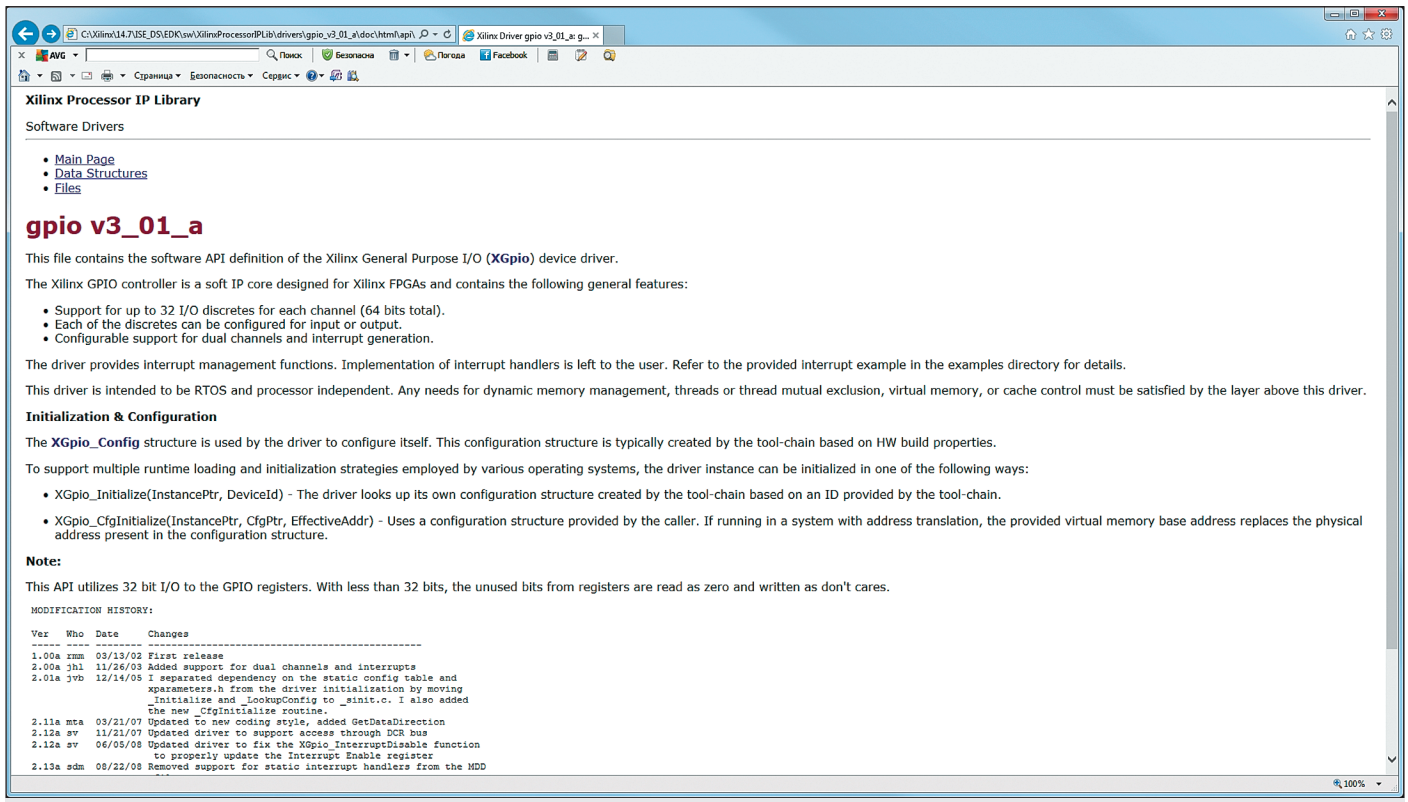

Рис. 2. Вид основной страницы документации, предоставляемой для драйверов IP-компонентов аппаратной части проектируемой системы

быстро ознакомиться с документацией драйверов библиотечных IP-компонентов аппаратной части проектируемой микропроцессорной системы, которые были включены в состав создаваемого пакета поддержки платы. Чтобы получить доступ к этой документации, следует развернуть подраздел **BSP Documentation**, как показано на рис. 1.

Для просмотра справочной информации, относящейся к какому-либо из драйверов, перечисленных в указанном подразделе, нужно расположить курсор в строке с идентификатором интересующего драйвера и дважды щелкнуть левой кнопкой мыши. После этого

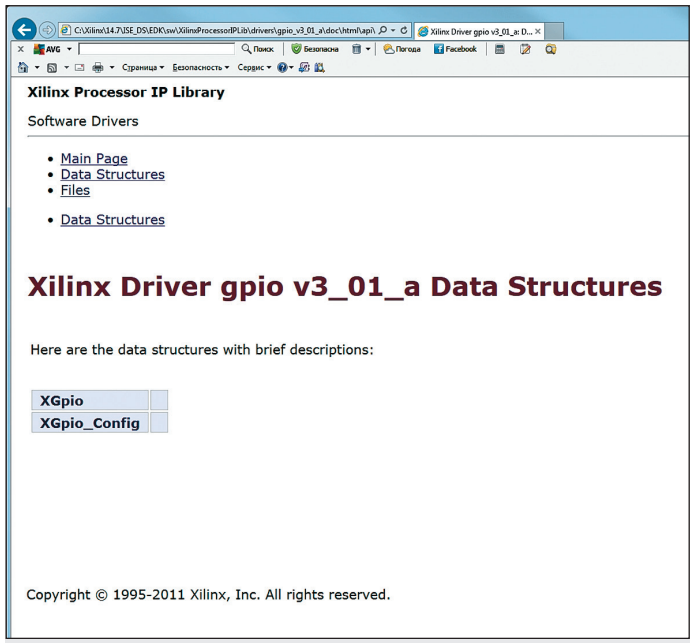

Рис. 3. Отображение информации о структурах данных, используемых в драйвере IP-компонента

автоматически открывается окно веб-обозревателя, используемого по умолчанию на компьютере разработчика (в частности, Internet Explorer компании Microsoft), в котором отображается основная страница документации, предоставляемой фирмой Xilinx для драйвера выбранного библиотечного IP-компонента. Структура данной страницы документации представлена на рис. 2 на примере драйвера, предназначенного для контроллера интерфейса портов ввода/вывода общего назначения, включенного в состав аппаратной части разрабатываемой простейшей системы сбора и обработки данных.

Основная страница документации драйверов библиотечных IP-компонентов, как правило, содержит справочную информацию об особенностях и возможностях предоставляемого драйвера, а также о поддерживаемых функциях интерфейса прикладного программирования API. Здесь же приводятся сведения обо всех существующих версиях выбранного драйвера с указанием даты их создания и наиболее значительных отличий. Кроме того, в верхней части основной станицы документации представлены гиперссылки, позволяющие перейти к более подробной информации, необходимой для практического использования соответствующего драйвера. В частности, гиперссылка **Data Structures** предоставляет возможность перехода к странице документации, в которой перечислены все структуры данных, применяемые в используемых драйверах библиотечных IP-компонентов. В качестве примера на рис. 3 приведен вид этой страницы для драйвера контроллера интерфейса портов ввода/вывода общего назначения. Чтобы получить более подробные сведения о каждой из представленных структур, достаточно расположить курсор в строке с ее идентификатором и щелкнуть левой кнопкой мыши.

На основной странице документации драйверов библиотечных IP-компонентов присутствует также гиперссылка **Files**, которая позволяет перейти к странице, содержащей список всех исходных файлов для выбранного драйвера. Типовой вид этой страницы демонстрирует рис. 4 на примере драйвера контроллера интерфейса портов ввода/вывода общего назначения.

Для того чтобы ознакомиться с детализированной информацией, относящейся к каждому исходному файлу драйвера, нужно указать

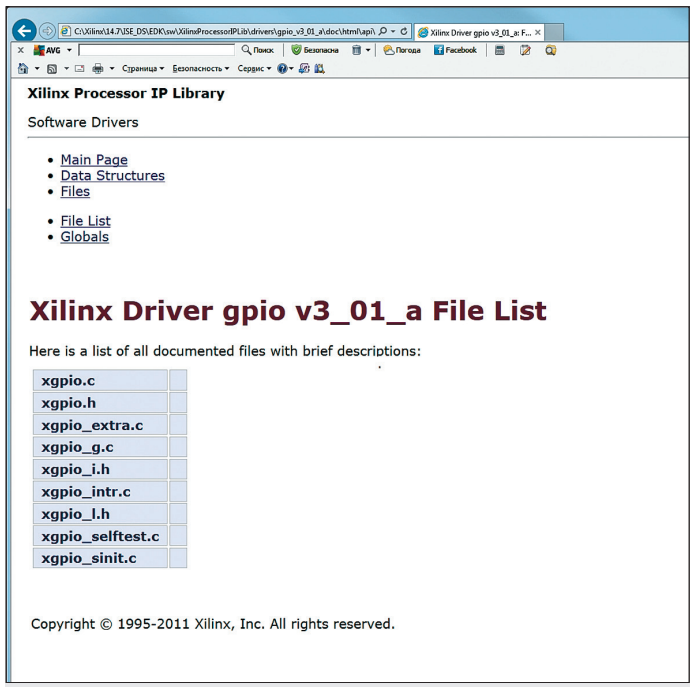

Рис. 4. Отображение списка всех исходных файлов драйвера IP-компонента

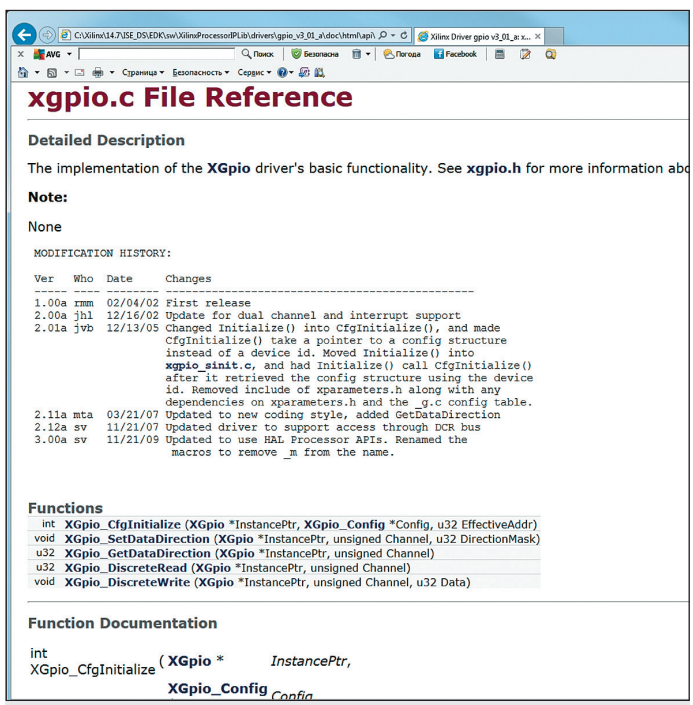

Рис. 5. Отображение подробной информации об исходном файле драйвера IP-компонента

курсором строку с идентификатором интересующего файла и щелкнуть левой кнопкой мыши. В результате открывается страница, чей вид представлен на рис. 5. Эта страница содержит сведения обо всех имеющихся версиях выбранного файла, включающие дату создания и основные отличия от предыдущих. Здесь же перечислены все функции, которые определены или объявлены в выбранном файле. Для каждой из функций приведены справочные данные о назначении и типе ее формальных параметров и возвращаемого значения.

На страницах, содержащих список исходных файлов драйвера библиотечного IP-компонента, а также подробную информацию об этих файлах (рис. 4, 5), присутствует гиперссылка Global, кото-

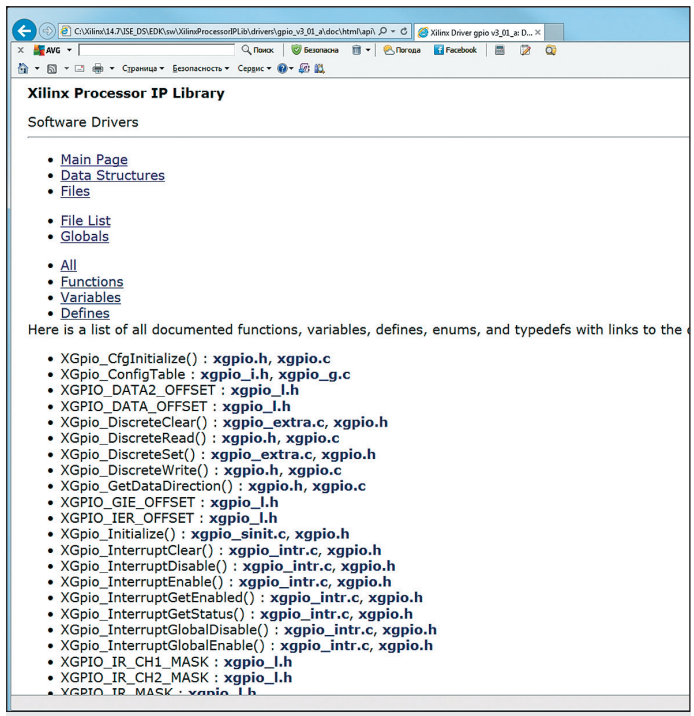

Рис. 6. Отображение списка всех функций, глобальных переменных и определений макросов драйвера IP-компонента

рая предоставляет возможность отображения списка всех функций, глобальных переменных и определений макросов соответствующего драйвера. В каждой строке этого списка приведены идентификаторы исходных файлов, содержащие объявления и определения соответствующей функции, переменной или макроса. При расположении курсора на тексте идентификатора файла и щелчке левой кнопкой мыши выполняется переход к странице, содержащей детализированную информацию о выбранном файле. На рис. 6 показан типовой вид страницы со сведениями обо всех функциях, глобальных переменных и определениях макросов драйвера библиотечного IP-компонента.

В верхней части этой страницы представлены гиперссылки **All**, **Functions**, **Variables** и **Defines**, которые позволяют осуществить фильтрацию отображаемой информации. При выборе гиперссылки **All** на рассматриваемой странице будет присутствовать список всех перечисленных выше программных элементов. Для отображения списка, содержащего только имена функций, нужно воспользоваться гиперссылкой **Functions**. Возможность просмотра только списка идентификаторов всех глобальных переменных предоставляет гиперссылка **Variables**. Переход к списку всех глобальных определений макросов осуществляется с помощью гиперссылки **Defines**.

Для ознакомления с примерами практического использования драйверов IP-компонентов, включенных в состав формируемого пакета поддержки платы, которые предлагаются фирмой Xilinx, рекомендуется воспользоваться гиперссылками **Examples**, представленными на вкладке *system.mss* основного окна средств Xilinx SDK в строках с идентификаторами соответствующих драйверов (рис. 1). При расположении курсора на любой из этих гиперссылок и щелчке левой кнопкой мыши открывается страница документации, содержащая список доступных примеров применения выбранного драйвера, типовой вид которой представлен на рис. 7.

В каждой строке этого списка после названия примера использования драйвера в скобках указана гиперссылка **source**, позволяющая ознакомиться с текстом исходных модулей описания предлагаемого проекта прикладной программы. При расположении курсора на соответствующей гиперссылке и щелчке левой кнопкой мыши на экране появляется стандартная панель запроса, предлагающая

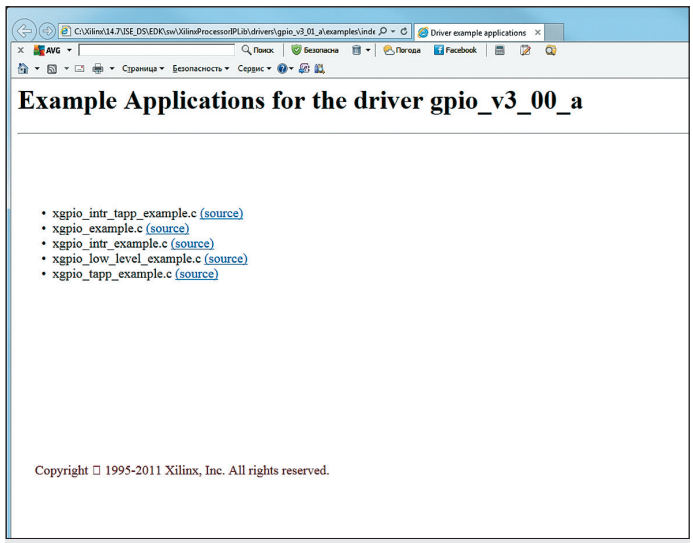

Рис. 7. Отображение списка примеров использования драйвера IP-компонента

открыть или сохранить эти модули в виде файла на диске. В случае выбора первого варианта нужные модули исходного описания будут отрыты непосредственно в окне веб-обозревателя или в окне текстового редактора, применяемого по умолчанию в операционной системе Windows XP/Windows 7 для просмотра и коррекции файлов текстового формата, например в «**Блокноте**». Использование такого текстового редактора для работы с исходным кодом программ является неэффективным. Поэтому в открывшейся панели запроса рекомендуется выбрать вариант записи исходных модулей описания в виде файла на диске, затем открыть его в интегрированной среде разработки программного обеспечения, воспользовавшись командой **Open File** из всплывающего меню **File** (рис. 8). Отдельные фрагменты текста отображаемых модулей предоставляемых примеров целесообразно скопировать в буфер обмена и впоследствии использовать их как шаблоны при подготовке исходного кода разрабатываемых прикладных программ для проектируемой микропроцессорной системы.

Файл опций для генератора библиотек, представленный в составе сформированного комплекта исходных модулей пакета BSP, содержит параметры командной строки запуска программы Libgen. Чтобы ознакомиться с содержимым этого файла и при необходимости внести требуемые изменения, следует расположить курсор в строке с его названием (libgen.options), которая представлена во встроенной панели **Project Explorer**, и дважды щелкнуть левой кнопкой мыши. При этом открывается новая вкладка встроенного текстового редактора интегрированной среды разработки программного обеспечения, где отображается текст автоматически созданного файла опций для генератора библиотек Library Generator. В качестве примера на рис. 9 приведено содержание файла параметров запуска программы Libgen, сгенерированного в процессе создания исходных модулей пакета BSP для простейшей системы сбора и обработки данных.

Справочная информация обо всех возможных параметрах командной строки запуска генератора библиотек содержится в [15]. Чтобы добавить новые опции или изменить параметры, установленные автоматически, следует воспользоваться клавиатурой. При этом необходимо учитывать, что все внесенные изменения в файле опций для генератора библиотек вступят в силу только после его сохранения на диске. Запись файла осуществляется с помощью команды **Save** из всплывающего меню **File** или кнопки быстрого доступа, расположенной на оперативной панели управления основного окна средств Xilinx SDK.

Красным цветом в тексте выделены названия файлов пиктограмм, которые нужно заменить изображениями из этих файлов.

Командный файл (Makefile), входящий в состав сформированного комплекта исходных модулей пакета BSP, содержит все необходимые директивы управления процессом автоматической генерации пакета поддержки платы для разрабатываемой встраиваемой системы. Для контроля и возможной коррекции заданных директив нужно открыть указанный файл, поместив курсор на строку **Makefile**, которая отображается во встроенной панели **Project Explorer**, и дважды щелкнуть левой кнопкой мыши. После этого в основном окне среды

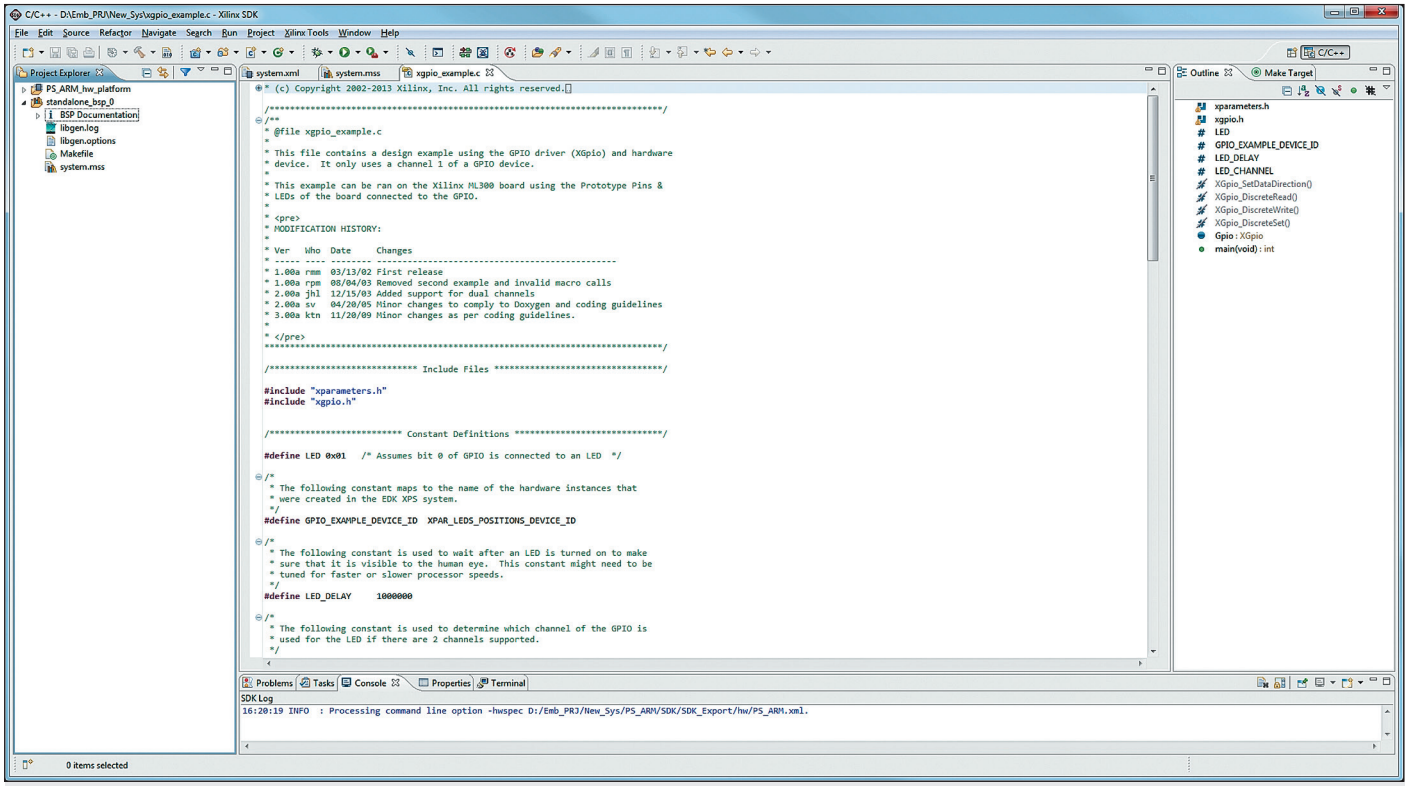

Рис. 8. Отображение исходного текста примера использования драйвера IP-компонента

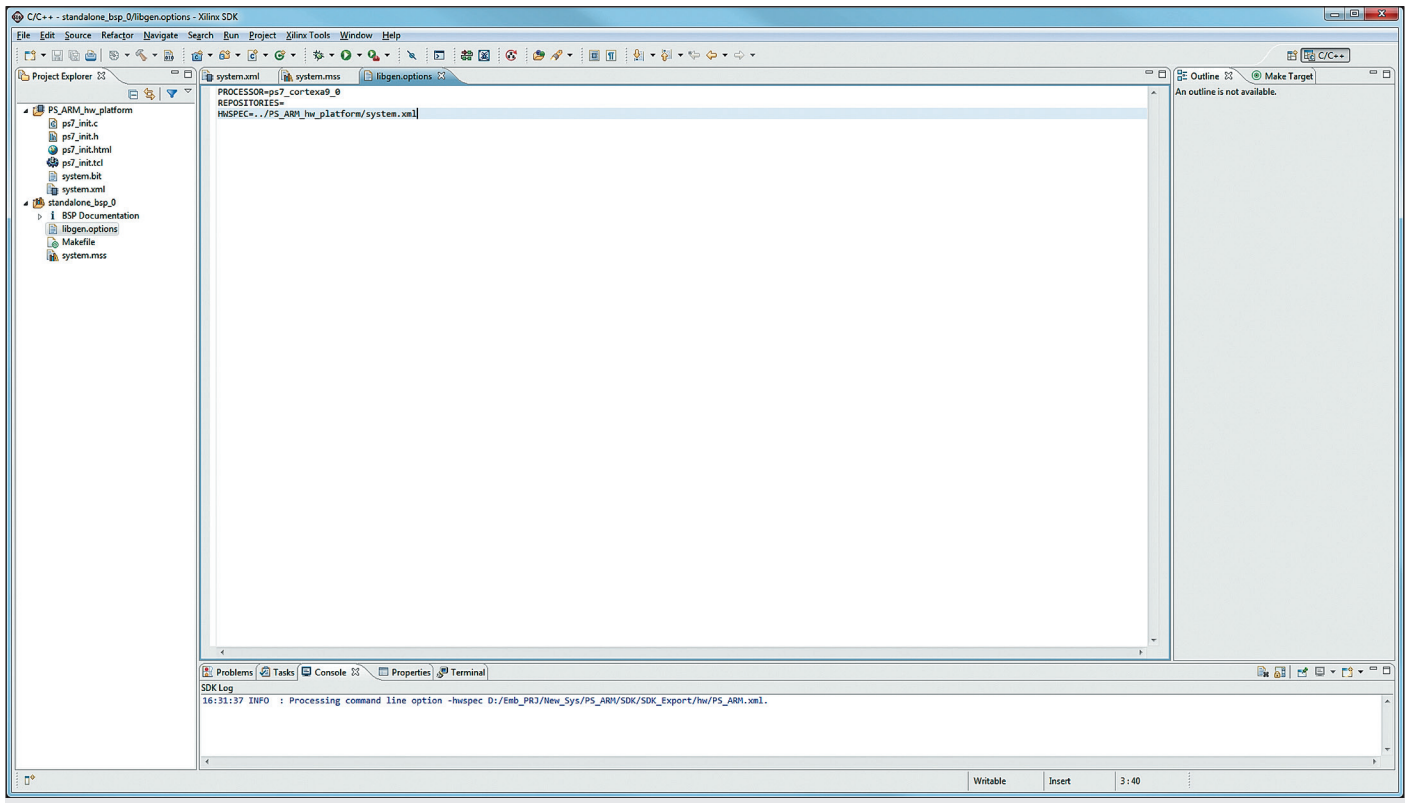

Рис. 9. Просмотр и редактирование файла опций для генератора библиотек Library Generator

разработки добавляется новая вкладка с названием **Makefile**, чей вид демонстрирует рис. 10. При редактировании отображаемого текста файла директив управления необходимо сохранить все выполненные изменения тем же способом, что и в файле опций для генератора библиотек.

Основным исходным модулем при генерации пакета поддержки платы является спецификация программной платформы проектируемой микропроцессорной системы MSS. Поэтому в следующем разделе подробно рассматривается синтаксис и структура соответствующего файла.

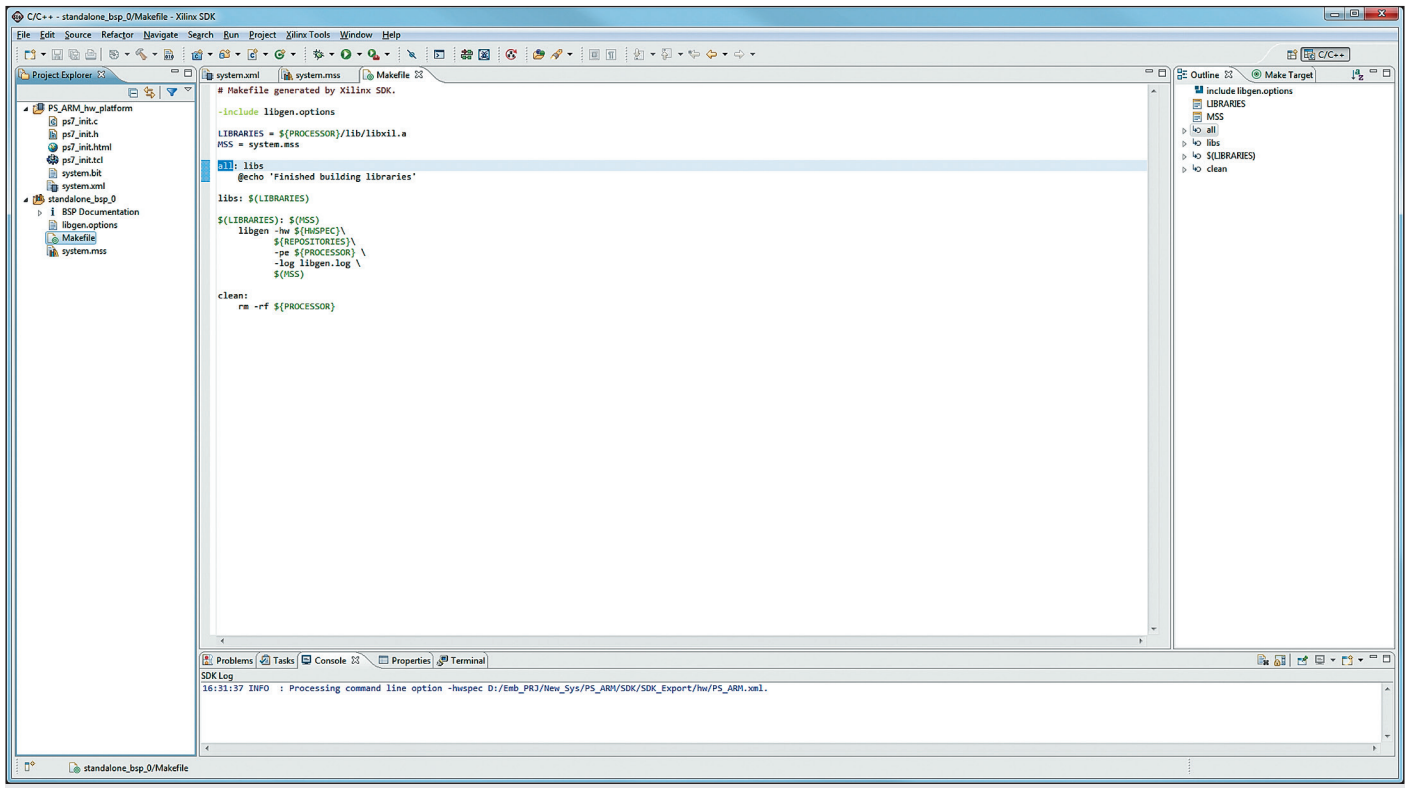

Рис. 10. Просмотр и редактирование файла управления процессом генерации пакета BSP

Синтаксис и структура файла спецификации программной платформы разрабатываемой микропроцессорной системы

Спецификация программной платформы проектируемой микропроцессорной системы представляет собой файл с расширением mss. Он имеет текстовый формат и содержит описание конфигурации и параметров программных компонентов разрабатываемой встраиваемой системы (операционной системы, библиотек и драйверов). Спецификация MSS определяет драйверы для каждого периферийного модуля, стандартные устройства ввода/вывода и программы обработки прерываний. Спецификация программных средств микропроцессорной системы является исходной информацией для генератора библиотек Library Generator (Libgen).

Язык и структура описания программных средств проектируемой микропроцессорной системы в общих чертах напоминает язык описания спецификации аппаратной платформы Microprocessor Hardware Specification (MHS), который был рассмотрен в [11].

Каждый компонент, включаемый в состав спецификации программных средств разрабатываемой встраиваемой системы MSS, описан с помощью конструкции, которая начинается с ключевого слова BEGIN и заканчивается ключевым словом END. Формат этой конструкции выглядит следующим образом:

```
BEGIN <тип_компонента_спецификации_MSS>
parameter <идентификатор_параметра1> = <значение_ параметра1>
[,<идентификатор_параметра1_1> = <значение_ параметра1_1>]
parameter < идентификатор_параметра2> = <значение_ параметра2>
[,<идентификатор_параметра2_1> = <значение_ параметра2_1>]
…
parameter < идентификатор_параметраN> = <значение_ параметраN>
[,<идентификатор_параметраN_1> = <значение_ параметраN_1>]
END
```

В приведенной конструкции квадратными скобками выделены параметры команды ***parameter***, наличие которых зависит от типа команды и не является обязательным.

Блок описания каждого компонента в спецификации программной платформы разрабатываемой встраиваемой системы открывает строка, содержащая ключевое слово BEGIN. В этой же строке указан тип описываемого компонента, который соответствует одному из видов программных средств или процессора. Тип компонента спецификации MSS определяется с помощью соответствующего ключевого слова. В рассматриваемой версии спецификации программных средств для этого зарезервированы четыре варианта ключевого слова: ***OS (Operating System)***, ***DRIVER***, ***LIBRARY***, ***PROCESSOR***. Ключевое слово ***OS*** используется в том случае, если блок описывает параметры операционной системы. Например, следующий блок определяет такие параметры, как тип и версия применяемой операционной системы, идентификатор используемого микропроцессорного ядра, идентификаторы стандартных устройств ввода/вывода.

```
BEGIN OS
   PARAMETER OS_NAME = standalone
   PARAMETER OS_VER = 3.11.a
   PARAMETER PROC_INSTANCE = ps7_cortexa9_0
   PARAMETER STDIN = ps7_uart_0
   PARAMETER STDOUT = ps7_uart_0
END
```

При описании драйвера некоторого IP-компонента проектируемой микропроцессорной системы указывается ключевое слово ***DRIVER***. В качестве примера ниже приводится блок описания параметров драйвера для контроллера интерфейса Ethernet процессорной системы PS кристаллов расширяемых вычислительных платформ семейства Zynq-7000 AP SoC.

```
BEGIN DRIVER
   PARAMETER DRIVER_NAME = emacps
   PARAMETER DRIVER_VER = 1.05.a
   PARAMETER HW_INSTANCE = ps7_ethernet_0
END
```

Если описываемый компонент спецификации MSS представляет собой библиотеку, то используется ключевое слово ***LIBRARY***. Так, в следующем блоке указаны параметры используемой библиотеки xilflash.

```
BEGIN LIBRARY
   PARAMETER LIBRARY_NAME = xilflash
   PARAMETER LIBRARY_VER = 3.04.a
   PARAMETER PROC_INSTANCE = ps7_cortexa9_0
END
```

Для описания параметров используемого микропроцессорного ядра в заголовке блока указывается ключевое слово ***PROCESSOR***. К примеру, в приведенном ниже блоке определяются следующие параметры процессора: условное обозначение типа микропроцессорного ядра, идентификатор и версия соответствующего драйвера, идентификатор экземпляра компонента микропроцессорного ядра в составе аппаратной платформы разрабатываемой встраиваемой системы.

```
BEGIN PROCESSOR
   PARAMETER DRIVER_NAME = cpu_cortexa9
   PARAMETER DRIVER_VER = 1.01.a
   PARAMETER HW_INSTANCE = ps7_cortexa9_0
END
```

В строках, следующих после ключевого слова ***BEGIN***, расположена совокупность команд ***parameter***, с помощью которых указываются параметры программного компонента, включаемого в состав спецификации MSS. Общее число команд N в этой конструкции обусловливается в первую очередь типом описываемого программного компонента. Количество параметров, определяемых в каждой команде ***parameter***, зависит от типа этой команды. В большинстве случаев в каждой строке, содержащей команду ***parameter***, определяется значение одного параметра. Но также могут применяться командные строки, в которых задаются значения двух параметров. Например, в команде, описывающей программу обработки прерывания, указано не только ее название, но и идентификатор соответствующего порта (сигнала) прерывания. Описание каждого компонента спецификации программной платформы проектируемой микропроцессорной системы завершается ключевым словом ***END***.

Команда ***parameter*** является основной и наиболее часто используемой в спецификации программных средств, поэтому рассмотрим ее подробнее. Прежде всего, следует обратить внимание на то, что все параметры, определяемые командой ***parameter***, можно разделить на две группы — глобальные и локальные. Глобальные параметры относятся ко всей спецификации программной платформы в целом. Команды ***parameter***, определяющие значения глобальных параметров, располагаются вне блоков описания программных компонентов BEGIN-END. К группе глобальных относятся параметры описания версии спецификации MSS и программ обработки прерываний для глобальных портов проектируемой микропроцессорной системы. Значение глобального параметра ***VERSION*** указывает номер версии формата спецификаций Platform Specification Format (PSF), используемой в проекте разрабатываемой встраиваемой системы. Значение этого параметра обычно задается в начале файла MSS. Формат команды, определяющей номер версии спецификации программной платформы, выглядит следующим образом:

```
parameter VERSION = <номер_версии_MSS>
```

Например, в командной строке ***parameter VERSION = 2.2.0***. указан номер версии формата спецификации MSS, используемый в САПР Xilinx ISE Design Suite 14.7.

Глобальный параметр ***INT_HANDLER*** предназначен для описания программы обслуживания прерывания, ассоциируемого с соответствующим внешним портом проектируемой микропроцессорной системы. Формат выражения, в котором определяется значение этого параметра, имеет следующий вид:

```
parameter INT_HANDLER = <идентификатор_программы_обработки_прерывания>, INT_PORT = <идентификатор_внешнего_сигнала(порта)_прерывания>
```

Например, командная строка

```
parameter INT_HANDLER = int_handl_proc, INT_PORT = Sys_Interrupt
```

устанавливает в качестве процедуры обслуживания прерывания, ассоциируемого с внешним портом Sys_Interrupt, программу int_handl_proc.

Локальные параметры спецификации MSS относятся только к соответствующим компонентам программной платформы разрабатываемой встраиваемой системы. Команды, определяющие значения локальных параметров, располагаются внутри блоков описания программных компонентов BEGIN-END. Рассмотрим подробнее наиболее часто используемые локальные параметры.

С помощью параметра ***HW_INSTANCE*** указывается позиционное обозначение (идентификатор) экземпляра IP-компонента, с которым ассоциируется описываемый программный компонент. Данный параметр применяется при описании драйверов (DRIVER) и процессора (PROCESSOR). Для определения значения этого параметра следует включить в состав соответствующего блока описания BEGIN-END командную строку, чей формат имеет следующий вид:

```
parameter HW_INSTANCE = <идентификатор_экземпляра_IP-компонента>
```

В этом выражении в качестве значения параметра ***HW_INSTANCE*** должен указываться тот же идентификатор, что задан для соответствующего экземпляра IP-компонента в спецификации аппаратной части проектируемой микропроцессорной системы MHS. В частности, для описания драйвера последовательного асинхронного приемопередатчика, обозначенного в спецификации MHS как **uart_port**, в соответствующий блок BEGIN DRIVER-END спецификации программных средств нужно включить командную строку ***parameter HW_INSTANCE = uart_port***.

Значение параметра ***PROC_INSTANCE*** указывает позиционное обозначение (идентификатор) экземпляра микропроцессорного ядра, с которым ассоциируется описываемая операционная система. Указываемый идентификатор должен совпадать с позиционным обозначением соответствующего экземпляра микропроцессорного ядра в спецификации MHS. Формат выражения, используемого в блоке описания BEGIN OS-END для определения значения этого параметра, выглядит следующим образом:

```
parameter PROC_INSTANCE = <идентификатор_экземпляра_микропроцессорного_ядра>
```

Параметр ***OS_NAME*** предназначен для указания используемой операционной системы. Для того чтобы определить его значение, необходимо в блок описания операционной системы BEGIN OS-END добавить следующее выражение:

```
parameter OS_NAME = <обозначение_используемой_операционной_системы>
```

Если разрабатываемое программное обеспечение проектируемой микропроцессорной системы не предусматривает использование какой-либо операционной системы, то для параметра ***OS_NAME*** следует указать значение **standalone**.

Номер версии применяемой операционной системы задается в форме значения параметра ***OS_VER***. Для определения значения этого параметра в блоке описания BEGIN OS-END должна присутствовать командная строка, формат которой имеет следующий вид:

```
parameter OS_VER = <номер_версии_операционной_системы>
```

Номер версии операционной системы указывается в формате *k.mn.a*, где *k*, *m*, *n* являются цифровыми символами, *a* — алфавитным символом.

Параметр ***DRIVER_NAME*** используется для назначения каждому компоненту спецификации аппаратной части проектируемой системы соответствующего драйвера. Формат команды, определяющей значение этого параметра, выглядит следующим образом:

```
parameter DRIVER_NAME = <идентификатор_драйвера_устройства>
```

В силу того что драйверы некоторых устройств имеют несколько версий, нужно уточнить номер применяемой версии драйвера с помощью параметра ***DRIVER_VER***. Выражение, предназначенное для определения значения параметра ***DRIVER_VER***, имеет следующий формат:

```
parameter DRIVER_VER = <номер_версии_драйвера_устройства>
```

Номер версии драйвера задается в том же формате, что и номер версии операционной системы.

Локальный параметр ***INT_HANDLER*** используется для описания программы обслуживания прерывания, ассоциируемого с соответствующим портом периферийного устройства. Формат командной строки, в которой устанавливается значение этого параметра, имеет следующий вид:

```
parameter INT_HANDLER = <идентификатор_программы_обработки_прерывания>, INT_PORT = <идентификатор_сигнала(порта)_прерывания_периферийного_устройства>
```

В этой командной строке должен быть указан тот же идентификатор порта прерывания периферийного устройства, что и в спецификации аппаратной части разрабатываемой встраиваемой системы для этого компонента.

Для описания включаемой библиотеки необходимо указать ее название и номер версии. Название применяемой библиотеки задается с помощью параметра ***LIBRARY_NAME***. В блок описания используемой библиотеки BEGIN LIBRARY-END должна быть включена командная строка, определяющая значение этого параметра и имеющая следующий формат:

```
parameter LIBRARY_NAME = <название_используемой_библиотеки>
```

Номер версии применяемой библиотеки определяется с помощью параметра ***LIBRARY_VER***. Его значение указывается в том же формате, что и номер версии операционной системы. Для этого используется выражение следующего вида:

```
parameter LIBRARY_VER = <номер_версии_библиотеки>
```

Для идентификации стандартных устройств ввода и вывода в блоке описания опций операционной системы существуют параметры ***STDIN*** и ***STDOUT*** соответственно. В качестве значений этих параметров указываются идентификаторы периферийных IP-компонентов, выполняющих роль стандартных устройств ввода и вывода. Формат командной строки для определения параметров ***STDIN*** и ***STDOUT*** выглядит следующим образом:

```
parameter STDIN = <идентификатор_устройства_ввода>
parameter STDOUT = <идентификатор_устройства_вывода>
```

Например, строки

```
parameter STDIN = uart_RS232
parameter STDOUT = uart_RS232
```

определяют в качестве стандартных устройств ввода и вывода последовательный асинхронный приемопередатчик uart_RS232.

В блоке описания микропроцессорного ядра предусмотрен ряд специфических параметров. Параметр ***XMDSTUB_PERIPHERAL*** позволяет задать идентификатор периферийного устройства, используемого для осуществления процесса отладки. Формат выражения, определяющего значение этого параметра, имеет следующий вид:

```
parameter XMDSTUB_PERIPHERAL = <идентификатор_устройства_отладки>
```

Параметр ***COMPILER*** предназначен для определения компилятора, который должен применяться для компиляции используемых библиотек и драйверов. Его значение должно соответствовать типу применяемого микропроцессорного ядра. Формат соответствующей командной строки выглядит следующим образом:

```
parameter COMPILER = <идентификатор_компилятора>
```

С помощью параметра ***ARCHIVER*** указывается программа архивации объектных

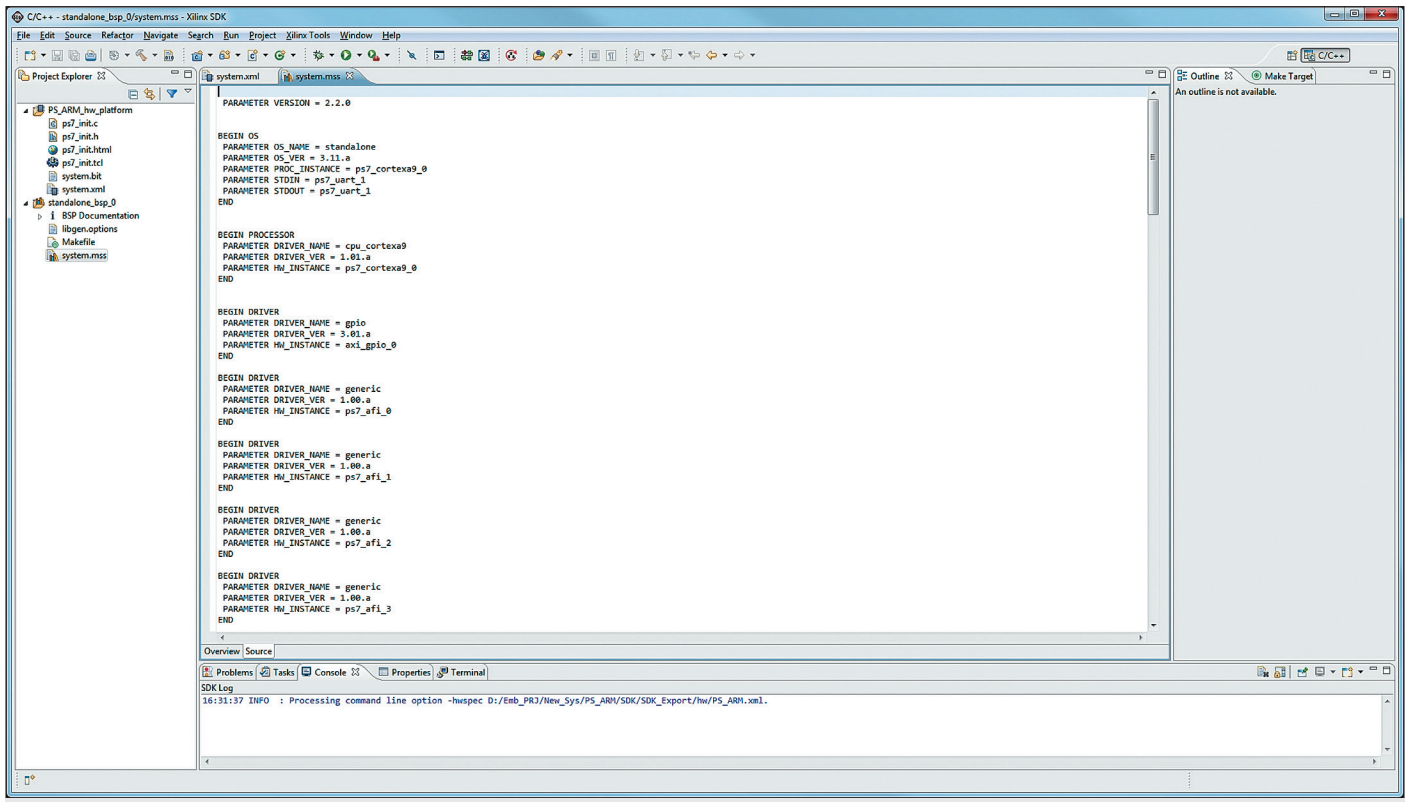

Рис. 11. Отображение текста спецификации программной платформы разрабатываемой системы

файлов в библиотеки. Выбор архиватора зависит от типа используемого микропроцессорного ядра. Чтобы указать требуемый архиватор, нужно включить в состав соответствующего блока спецификации MSS командную строку следующего формата:

```
parameter ARCHIVER = <название_программы_архивации>
```

В блоке описания процессора могут также применяться параметры ***COMPILER_FLAGS*** и ***EXTRA_COMPILER_FLAGS***, которые задают опции компиляции драйверов и библиотек. При использовании стандартного компилятора и архиватора данные параметры не рекомендуется указывать.

В тесте спецификации программных средств, так же как и в спецификации аппаратной платформы разрабатываемой микропроцессорной системы, могут использоваться комментарии. Это делает структуру файла MSS более прозрачной и облегчает поиск параметров, значение которых необходимо скорректировать. Комментарием считается часть любой строки, которая следует после символа #. Комментарий может занимать полную строку, если в начале ее указан символ #. Примеры комментариев различного вида в файле MSS демонстрируют следующие командные строки:

```
# -------------Assign drivers --------------------------------

parameter DRIVER_NAME = uartlite # DRIVER for RS-232
```

Типовая структура файла спецификации программной платформы проектируемой микропроцессорной системы, как правило, содержит следующие разделы:

- команда, указывающая номер версии формата спецификации MSS;
- описание используемой операционной системы;
- определение параметров микропроцессорного ядра;
- блоки описания драйверов IP-компонентов периферийных устройств разрабатываемой встраиваемой системы;
- блоки описания используемых библиотек.

Поскольку файл спецификации MSS имеет текстовый формат, для его подготовки и внесения изменений можно использовать любой текстовый редактор, например, который входит в состав интегрированной среды разработки программного обеспечения Xilinx SDK. Однако процедура создания абсолютно новой спецификации программной платформы в текстовом редакторе занимает достаточно много времени и не исключает появления различных ошибок. Наиболее эффективным является интерактивный способ подготовки спецификации MSS, представленный в предыдущей части данной статьи. Текстовый режим целесообразно использовать в основном для редактирования значений отдельных параметров или в тех случаях, когда новая спецификация программных средств разрабатываемой микропроцессорной системы формируется на основе уже существующего файла MSS.

Для отображения текста спецификации программной платформы необходимо на вкладке *system.mss* основного окна среды разработки программного обеспечения (рис. 1) открыть страницу **Source**, поместив курсор на закладку с ее названием и щелкнув левой кнопкой мыши. После этого основное окно средств Xilinx SDK приобретает вид, приведенный на рис. 11.

Ниже в качестве примера приводится текст спецификации программной платформы для простейшей системы сбора и обработки данных, разработка аппаратной части которой рассмотрена в предыдущей публикации цикла [11].

```
# Номер версии формата спецификации MSS
#
PARAMETER VERSION = 2.2.0
#
# Описание параметров операционной системы
#
BEGIN OS
   PARAMETER OS_NAME = standalone
   PARAMETER OS_VER = 3.11.a
   PARAMETER PROC_INSTANCE = ps7_cortexa9_0
   PARAMETER STDIN = ps7_uart_1
   PARAMETER STDOUT = ps7_uart_1
END
#
# Описание параметров микропроцессорного ядра
#
BEGIN PROCESSOR
   PARAMETER DRIVER_NAME = cpu_cortexa9
   PARAMETER DRIVER_VER = 1.01.a
   PARAMETER HW_INSTANCE = ps7_cortexa9_0
END
#
# Описание драйверов периферийных устройств
#
BEGIN DRIVER
   PARAMETER DRIVER_NAME = generic
   PARAMETER DRIVER_VER = 1.00.a
   PARAMETER HW_INSTANCE = ps7_afi_0
END
#
```

```
BEGIN DRIVER
  PARAMETER DRIVER_NAME = generic
  PARAMETER DRIVER_VER = 1.00.a
  PARAMETER HW_INSTANCE = ps7_afi_1
END
#
BEGIN DRIVER
  PARAMETER DRIVER_NAME = generic
  PARAMETER DRIVER_VER = 1.00.a
  PARAMETER HW_INSTANCE = ps7_afi_2
END
#
BEGIN DRIVER
  PARAMETER DRIVER_NAME = generic
  PARAMETER DRIVER_VER = 1.00.a
  PARAMETER HW_INSTANCE = ps7_afi_3
END
#
BEGIN DRIVER
  PARAMETER DRIVER_NAME = generic
  PARAMETER DRIVER_VER = 1.00.a
  PARAMETER HW_INSTANCE = ps7_coresight_comp_0
END
#
BEGIN DRIVER
  PARAMETER DRIVER_NAME = generic
  PARAMETER DRIVER_VER = 1.00.a
  PARAMETER HW_INSTANCE = ps7_ddr_0
END
#
BEGIN DRIVER
  PARAMETER DRIVER_NAME = generic
  PARAMETER DRIVER_VER = 1.00.a
  PARAMETER HW_INSTANCE = ps7_ddrc_0
END
#
BEGIN DRIVER
  PARAMETER DRIVER_NAME = devcfg
  PARAMETER DRIVER_VER = 2.04.a
  PARAMETER HW_INSTANCE = ps7_dev_cfg_0
END
#
BEGIN DRIVER
  PARAMETER DRIVER_NAME = dmaps
  PARAMETER DRIVER_VER = 1.06.a
  PARAMETER HW_INSTANCE = ps7_dma_ns
END
#
BEGIN DRIVER
  PARAMETER DRIVER_NAME = dmaps
  PARAMETER DRIVER_VER = 1.06.a
  PARAMETER HW_INSTANCE = ps7_dma_s
END
#
BEGIN DRIVER
  PARAMETER DRIVER_NAME = emacps
  PARAMETER DRIVER_VER = 1.05.a
  PARAMETER HW_INSTANCE = ps7_ethernet_0
END
#
BEGIN DRIVER
  PARAMETER DRIVER_NAME = generic
  PARAMETER DRIVER_VER = 1.00.a
  PARAMETER HW_INSTANCE = ps7_globaltimer_0
END
#
BEGIN DRIVER
  PARAMETER DRIVER_NAME = gpiops
  PARAMETER DRIVER_VER = 1.02.a
  PARAMETER HW_INSTANCE = ps7_gpio_0
END
#
BEGIN DRIVER
  PARAMETER DRIVER_NAME = generic
  PARAMETER DRIVER_VER = 1.00.a
  PARAMETER HW_INSTANCE = ps7_gpv_0
END
#
BEGIN DRIVER
  PARAMETER DRIVER_NAME = generic
  PARAMETER DRIVER_VER = 1.00.a
  PARAMETER HW_INSTANCE = ps7_intc_dist_0
END
#
BEGIN DRIVER
  PARAMETER DRIVER_NAME = generic
  PARAMETER DRIVER_VER = 1.00.a
  PARAMETER HW_INSTANCE = ps7_iop_bus_config_0
END
#
BEGIN DRIVER
  PARAMETER DRIVER_NAME = generic
  PARAMETER DRIVER_VER = 1.00.a
  PARAMETER HW_INSTANCE = ps7_l2cachec_0
END
#
BEGIN DRIVER
  PARAMETER DRIVER_NAME = generic
  PARAMETER DRIVER_VER = 1.00.a
  PARAMETER HW_INSTANCE = ps7_ocmc_0
END
#
BEGIN DRIVER
  PARAMETER DRIVER_NAME = qspips
  PARAMETER DRIVER_VER = 2.03.a
  PARAMETER HW_INSTANCE = ps7_qspi_0
END
#
BEGIN DRIVER
  PARAMETER DRIVER_NAME = generic
  PARAMETER DRIVER_VER = 1.00.a
  PARAMETER HW_INSTANCE = ps7_qspi_linear_0
END
#
BEGIN DRIVER
  PARAMETER DRIVER_NAME = generic
  PARAMETER DRIVER_VER = 1.00.a
  PARAMETER HW_INSTANCE = ps7_ram_0
END
#
BEGIN DRIVER
  PARAMETER DRIVER_NAME = generic
  PARAMETER DRIVER_VER = 1.00.a
  PARAMETER HW_INSTANCE = ps7_ram_1
END
#
BEGIN DRIVER
  PARAMETER DRIVER_NAME = generic
  PARAMETER DRIVER_VER = 1.00.a
  PARAMETER HW_INSTANCE = ps7_scuc_0
END
#
BEGIN DRIVER
  PARAMETER DRIVER_NAME = scugic
  PARAMETER DRIVER_VER = 1.05.a
  PARAMETER HW_INSTANCE = ps7_scugic_0
END
#
BEGIN DRIVER
  PARAMETER DRIVER_NAME = scutimer
  PARAMETER DRIVER_VER = 1.02.a
  PARAMETER HW_INSTANCE = ps7_scutimer_0
END
#
BEGIN DRIVER
  PARAMETER DRIVER_NAME = scuwdt
  PARAMETER DRIVER_VER = 1.02.a
  PARAMETER HW_INSTANCE = ps7_scuwdt_0
END
#
BEGIN DRIVER
  PARAMETER DRIVER_NAME = generic
  PARAMETER DRIVER_VER = 1.00.a
  PARAMETER HW_INSTANCE = ps7_sd_0
END
#
BEGIN DRIVER
  PARAMETER DRIVER_NAME = generic
  PARAMETER DRIVER_VER = 1.00.a
  PARAMETER HW_INSTANCE = ps7_slcr_0
END
#
BEGIN DRIVER
  PARAMETER DRIVER_NAME = ttcps
  PARAMETER DRIVER_VER = 1.01.a
  PARAMETER HW_INSTANCE = ps7_ttc_0
END
#
BEGIN DRIVER
  PARAMETER DRIVER_NAME = uartps
  PARAMETER DRIVER_VER = 1.05.a
  PARAMETER HW_INSTANCE = ps7_uart_1
END
#
BEGIN DRIVER
  PARAMETER DRIVER_NAME = usbps
  PARAMETER DRIVER_VER = 1.05.a
  PARAMETER HW_INSTANCE = ps7_usb_0
END
#
BEGIN DRIVER
  PARAMETER DRIVER_NAME = xadcps
  PARAMETER DRIVER_VER = 1.02.a
  PARAMETER HW_INSTANCE = ps7_xadc_0
END
#
# Описание используемых библиотек
#
BEGIN LIBRARY
  PARAMETER LIBRARY_NAME = lwip140
  PARAMETER LIBRARY_VER = 1.06.a
  PARAMETER PROC_INSTANCE = ps7_cortexa9_0
END
#
BEGIN LIBRARY
  PARAMETER LIBRARY_NAME = xilmfs
  PARAMETER LIBRARY_VER = 1.00.a
  PARAMETER PROC_INSTANCE = ps7_cortexa9_0
END
```

На основании сформированной спецификации программных средств далее выполняется генерация пакета поддержки платы BSP, включающая компиляцию соответствующих драйверов и библиотек.

Генерация пакета поддержки платы BSP для разрабатываемой микропроцессорной системы

Процесс генерации пакета поддержки платы для разрабатываемой микропроцессорной системы, реализуемой на базе кристаллов расширяемых вычислительных платформ фирмы Xilinx, осуществляется в автоматическом режиме. В ходе этого процесса выполняется сформированный командный файл (Makefile). Запуск процесса генерации пакета BSP может осуществляться вручную или автоматически. Для запуска вручную нужно в главном меню основного окна среды разработки программного обеспечения выделить пункт **Project**, после чего в открывшемся всплывающем меню выбрать команду **Build Project**, как показано на рис. 12.

Кроме того, с этой же целью можно воспользоваться аналогичной командой из контекстно-зависимого всплывающего меню, открываемого щелчком правой копкой мыши на строке с названием раздела генерируемого пакета поддержки платы, представленной во встроенной панели **Project Explorer**. В результате выполнения указанных операций генератор библиотек Library Generator на основании информации, приведенной в спецификации программной платформы MSS, формирует компоненты пакета поддержки платы для разрабатываемой микропроцессорной системы (драйверы, библиотеки, программы обслуживания прерываний). Весь ход процесса генерации пакета поддержки платы отображается во встроенной панели консольных сообщений **Console** основного окна средств Xilinx SDK (рис. 13).

При этом в разделе генерируемого пакета BSP автоматически создается подраздел, название которого совпадает с идентификатором (позиционным обозначением) используемого микропроцессорного ядра. В этом подразделе автоматически формируются четыре каталога с названиями **code**, **include**, **lib** и **libsrc**, которые предназначены для записи различных компонентов пакета поддержки платы (рис. 14).

Раздел **code** используется для хранения файлов, которые содержат код, применяемый в процессе отладки. В разделе **include** представлены заголовочные файлы для драйверов IP-компонентов периферийных устройств, включенных в состав проектиру-

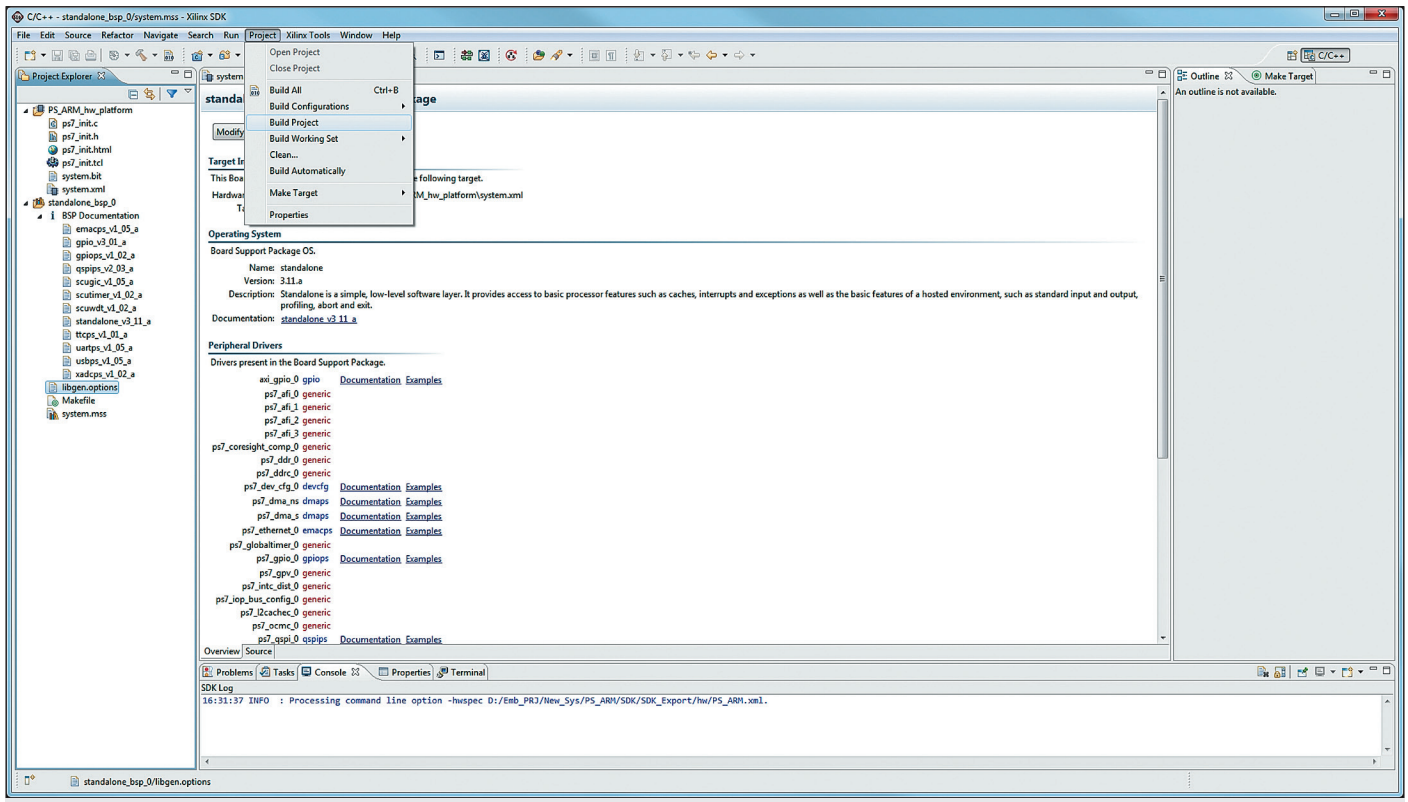

Рис. 12. Запуск процесса генерации пакета поддержки платы

емой системы. Каталог **lib** содержит библиотечные файлы. В раздел **libsrc** записываются исходные и вспомогательные файлы, необходимые в процессе компиляции библиотек и драйверов.

Для автоматического запуска процесса генерации пакета поддержки платы следует во всплывающем меню **Project**, чей вид приведен на рис. 12, щелчком левой кнопки мыши активизировать команду **Build Automatically**. При этом в указанной строке меню появляется маркер, информирующий об активации режима автоматического запуска компиляции и сборки проектов. В данном режиме, в частности, осуществляется автоматический старт процесса генерации пакета

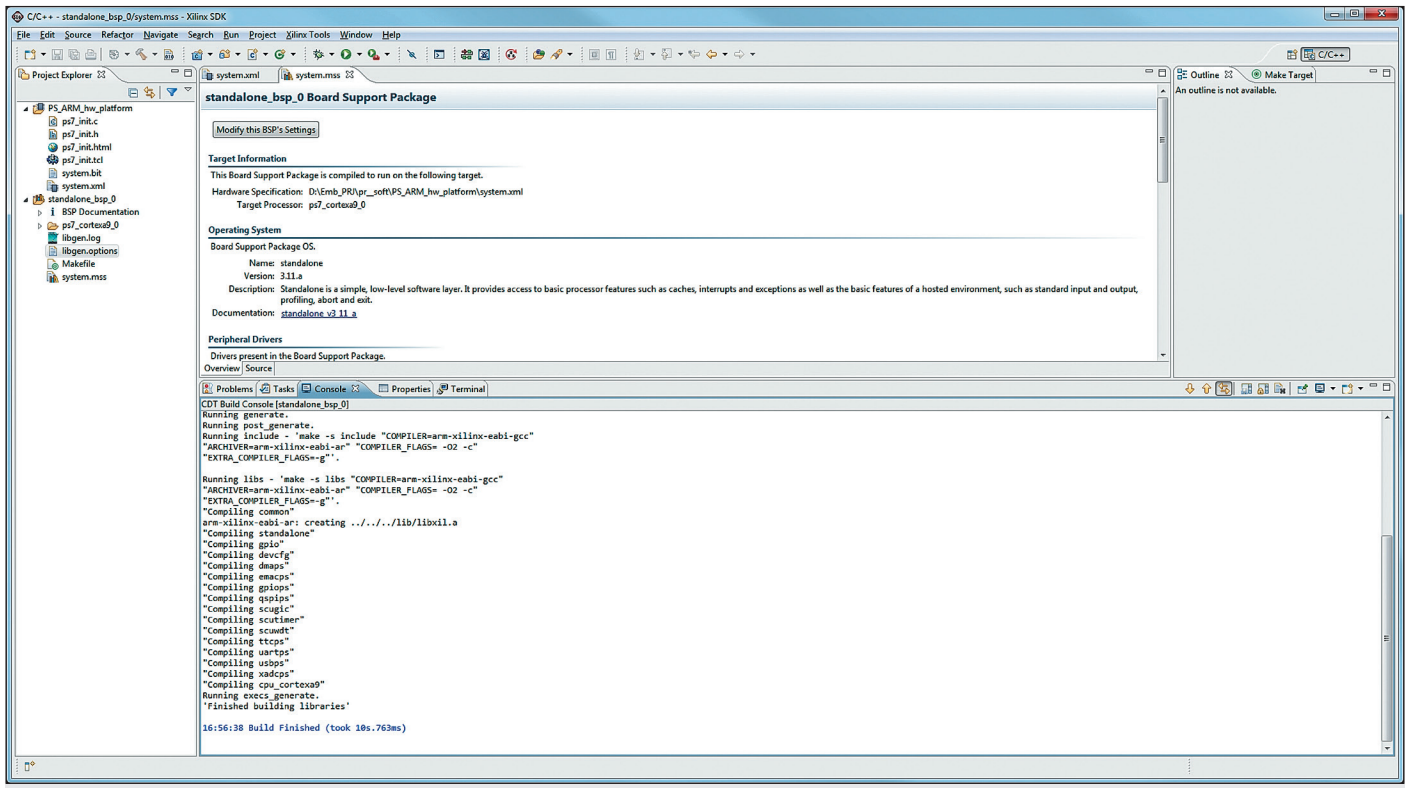

Рис. 13. Отображение хода выполнения процесса генерации пакета BSP

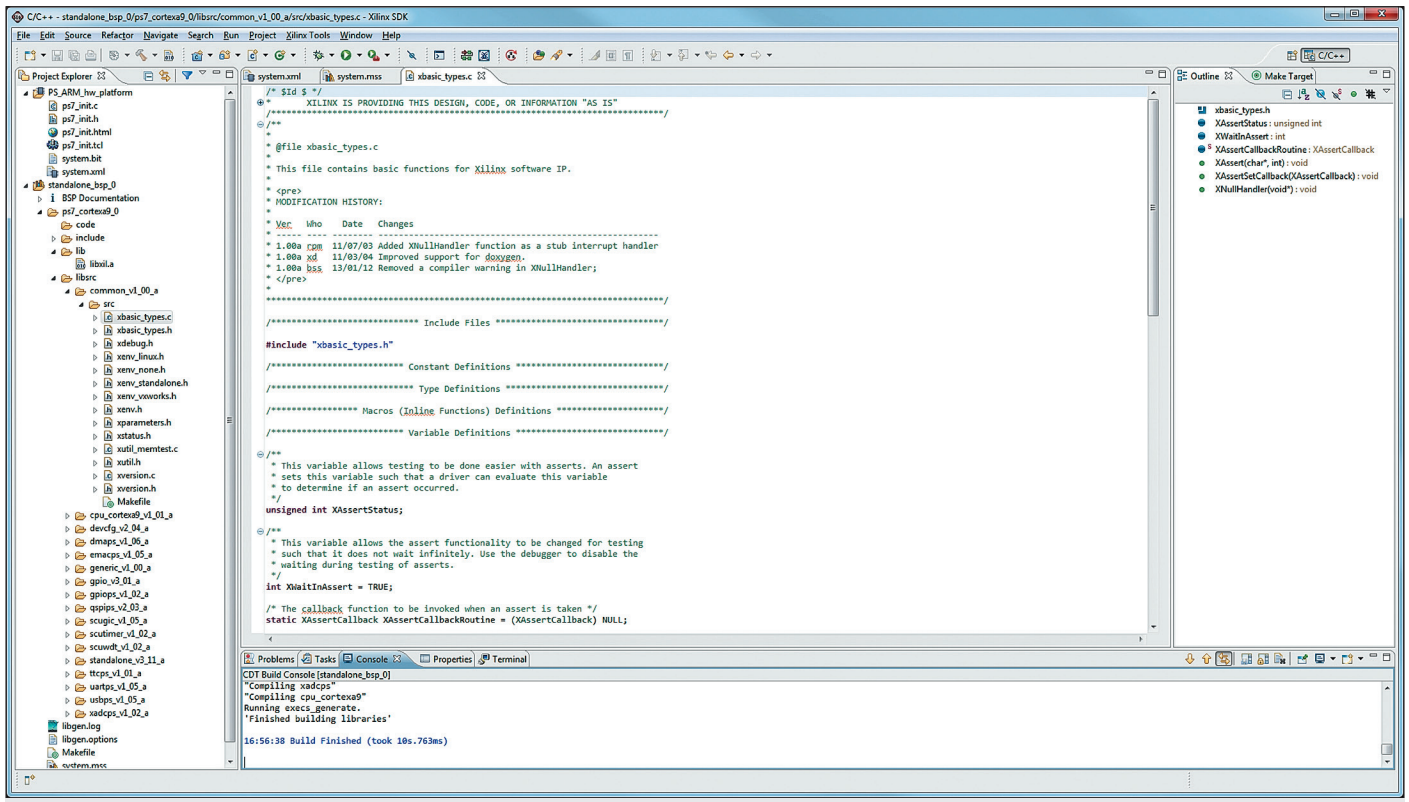

Рис. 14. Отображение структуры и компонентов сгенерированного пакета BSP

BSP сразу после внесения каких-либо изменений в исходные модули этого пакета и их сохранения. Чтобы деактивировать установленный режим автоматического запуска компиляции и сборки проектов, надо повторно выполнить команду **Build Automatically** из всплывающего меню **Project**.

В качестве примера сгенерируем пакет поддержки платы для простейшей системы сбора и обработки данных, спецификация программной платформы которой была представлена выше. При отсутствии ошибок в каталоге пакета BSP должен появиться раздел с названием **ps7_cortexa9_0**, содержащий все необходимые файлы драйверов, библиотек и программ обслуживания прерываний, структура которого показана на рис. 14.

После успешного завершения процесса генерации пакета поддержки платы можно перейти непосредственно к разработке прикладного программного обеспечения для проектируемой микропроцессорной системы.

Продолжение следует

Литература

1. Зотов В. Расширение семейства программируемых систем на кристалле Zynq-7000 AP SoC // Компоненты и технологии. 2013. № 12. 2014. № 1.
2. Zynq-7000 All Programmable SoC Overview. Advance Product Specification. Xilinx, 2013.
3. Zynq-7000 All Programmable SoC (XC7Z010, XC7Z015, and XC7Z020): DC and AC Switching Characteristics. Xilinx, 2013.
4. Zynq-7000 All Programmable SoC (XC7Z030, XC7Z045, and XC7Z100): DC and AC Switching Characteristics. Xilinx, 2013.
5. Zynq-7000 All Programmable SoC Technical Reference Manual. Xilinx, 2013.
6. Zynq-7000 All Programmable SoC: Concepts, Tools, and Techniques (CTT) A Hands-On Guide to Effective Embedded System Design. Xilinx, 2013.
7. Zynq-7000 All Programmable SoC Software Developers Guide. Xilinx, 2013.
8. Zynq-7000 All Programmable SoC PCB Design and Pin Planning Guide. Xilinx, 2013.
9. 7 Series FPGAs and Zynq-7000 All Programmable SoC XADC Dual 12-Bit 1 MSPS Analog-to-Digital Converter User Guide. Xilinx, 2012.
10. Зотов В. Средства автоматизированного проектирования и этапы разработки встраиваемых микропроцессорных систем на базе расширяемых процессорных платформ семейства Zynq-7000 AP SoC // Компоненты и технологии. 2014. № 2–3.
11. Зотов В. Проектирование встраиваемых микропроцессорных систем на базе расширяемых процессорных платформ семейства Zynq-7000 AP SoC в САПР Xilinx ISE Design Suite // Компоненты и технологии. 2014. № 4–12. 2015. № 1.
12. Зотов В. Аппаратные средства разработки и отладки встраиваемых микропроцессорных систем, проектируемых на основе расширяемых вычислительных платформ фирмы Xilinx семейства Zynq-7000 AP SoC // Компоненты и технологии. 2013. № 1.
13. Зотов В. ZedBoard. Эффективный инструмент разработки и отладки встраиваемых микропроцессорных систем, проектируемых на основе расширяемых вычислительных платформ фирмы Xilinx семейства Zynq-7000 AP SoC // Компоненты и технологии. 2013. № 6.
14. Зотов В. MicroZed — семейство унифицированных модулей для отладки и реализации встраиваемых микропроцессорных систем, проектируемых на основе платформ фирмы Xilinx серии Zynq-7000 AP SoC // Компоненты и технологии. 2013. № 11.
15. Embedded System Tools Reference Manual. Xilinx, 2013.
16. Зотов В. Разработка программного обеспечения встраиваемых микропроцессорных систем, проектируемых на базе расширяемых вычислительных платформ семейства Zynq-7000 AP SoC фирмы Xilinx. Часть 1 // Компоненты и технологии. 2015. № 2.