

# **SF2-Junior-KIT**

**Обработка прерываний**

# **Введение**

В данном руководстве рассматриваются особенности архитектуры СнК SmartFusion2 в части касающейся реализации обработки прерываний, а также описан пример создания проекта СнК и БПО с реализацией процедур обработки аппаратных прерываний.

## **Необходимое программное обеспечение**

Для изучения материала, изложенного в данном руководстве необходимо следующее программное обеспечение:

- среда разработки [Microsemi Libero SoC v11.8](#);
- утилита для программирования микросхем СнК и ПЛИС FlashPro v11.8 или более поздняя версия, которая может быть установлена как часть пакета программ Microsemi Libero SoC и может быть запущена внутри Libero SoC или отдельно;
- среда разработки встраиваемого программного обеспечения SoftConsole v3.4 или более поздняя, которая может быть установлена как часть пакета программ Microsemi Libero SoC или отдельно;
- программа-оболочка HyperTerminal или аналогичное программное обеспечение (PuTTy или TeraTerm).

## **Необходимое аппаратное обеспечение**

Вам понадобится отладочный набор [SF2-Junior-KIT](#), включающий следующие компоненты:

- 1) Модуль SF2-Junior-KIT;
- 2) Жидкокристаллический дисплей 320x240 с интерфейсом SPI и сенсорной панелью (touchscreen);
- 3) Программатор FlashPro4;
- 4) USB – Bluetooth донгл;
- 5) Модуль приемопередатчика Bluetooth – UART;
- 6) Преобразователь напряжения AC-DC 9В 1A;
- 7) Кабель USB 2.0 A-male to mini-B.

## Особенности СнК SmirtFusion2 касающиеся обработки прерываний

Для согласования различных скоростей работы внешних устройств объединенных в систему наряду с режимом «программного обмена», используются режимы работы «по прерыванию» и «прямого доступа к памяти». В данном руководстве рассмотрен процесс конфигурации микроконтроллерной подсистемы СнК SmartFusion для работы в режиме обработки прерываний.

В информационно-управляющей системе, реализуемой на SmartFusion2 по способу реализации механизма обработки прерываний можно выделить три типа источников прерываний:

1) блоки микроконтроллерной подсистемы реализованные аппаратно. Примерами таких блоков являются MMUART\_0, I2C\_0, Timer\_0 и т.п. Отличительной особенностью блоков данного типа является полная поддержка механизма обработки прерываний как со стороны аппаратной, так и в программной части. То есть к контроллеру прерываний процессора аппаратно подключены «именные» линии прерываний от указанных блоков, в комплект прилагаемого Microsemi программного обеспечения входят программные драйверы блоков рассматриваемого типа. С функциями интерфейса прикладных программирования (API) для обслуживания механизма прерываний можно ознакомиться в руководстве программиста по данному блоку доступному на сайте Microsemi;

2) IP-ядра из стандартного каталога Libero SoC, например coreI2C, corePWM. IP-ядра «именных» прерываний не имеют, но в состав выходного интерфейса ядер включены сигналы «Int», которые могут быть подключены к контроллеру прерываний MSS Interrupt Management микроконтроллерной подсистемы SmartFusion2 при сборке проекта в мастере SmartDesign. Microsemi предлагает программные драйверы с необходимым API для обработки прерываний;

3) пользовательские IP-ядра и блоки архитектуры разрабатываемого устройства, реализуемые в FPGA Fabric и внешние по отношению к SoC SmartFusion2 источники сигналов. Примеры таких блоков – пользовательский счетчик импульсов на n двоичных разрядов написанный на языке Verilog или VHDL или кнопочный микропереключатель, подключенный к контакту ввода-вывода SoC SmartFusion2. Блоки этого типа специальной поддержки драйверами не имеют, разработчику предлагается использовать драйверы блока менеджера прерываний (MSS Interrupt Management) и блока контактов общего назначения микроконтроллерной подсистемы (MSS GPIO), входные сигналы которого в подобных случаях необходимо настроить как источники прерываний.

Компонент микроконтроллерной подсистемы MSS Interrupt Management по сути является частью стандартного контроллера прерываний процессора архитектуры ARM, доступной для подключения сигналов функциональных блоков, создаваемых в проектах пользователя. К его входам рекомендуется подключать выходы «Int» IP-ядер стандартного каталога Libero SOC. Обработчики прерываний связанные с сигналами, подаваемыми на входы этого компонента, выполняются с минимальной задержкой, однако простой и «прямолинейный» механизм этого компонента не допускает дополнительных настроек. Входы данного компонента чувствительны к только к положительному уровню входного сигнала (к логической «1»). Драйверы IP-ядер стандартного каталога Libero SOC имеют функции освобождения прерывания (ClearIRQ) которые выполняются в конце обработчика прерывания и гарантируют однократное вхождение в процедуру обработчика при наступлении события.

Для подключения сигналов прерываний от IP-ядер и компонентов пользователя целесообразно использовать блок MSS GPIO микроконтроллерной подсистемы. После включения выбранных контактов GPIO\_x и конфигурирования их в качестве линий прерываний можно определить тип чувствительности контакта GPIO – к «1», «0», к переднему, заднему или обоим фронтам сигнала.

## Описание проекта

В качестве примера создадим проект для СнК SmartFusin2 в котором будет осуществляться обработка сигналов и событий в режиме работы «по прерыванию» от следующих аппаратно-программных блоков:

- 1) интерфейс MMUART\_0 микроконтроллерной подсистемы;
- 2) аппаратно реализованный в микроконтроллерной подсистеме таймер MSS Timer2;
- 3) пользовательский 28-разрядный делитель частоты, реализованный в матрице ПЛИС;
- 4) источник короткого импульсного сигнала, формируемый 29-разрядным счетчиком, реализованным в матрице ПЛИС;
- 5) три кнопки установленные на плате отладочного набора, причем в случае первой кнопки активным уровнем прерывания будет сигнал логического «0»; для второй кнопки - отрицательный фронт сигнала; для третьей - оба фронта генерируемого кнопкой сигнала.

# Разработка проекта системы на кристалле

Запустите приложение Libero SoC 11.8, дважды кликнув на ярлычок на рабочем столе или на аналогичный в меню Пуск > Все программы > Microsemi > Libero SoC v11.8.

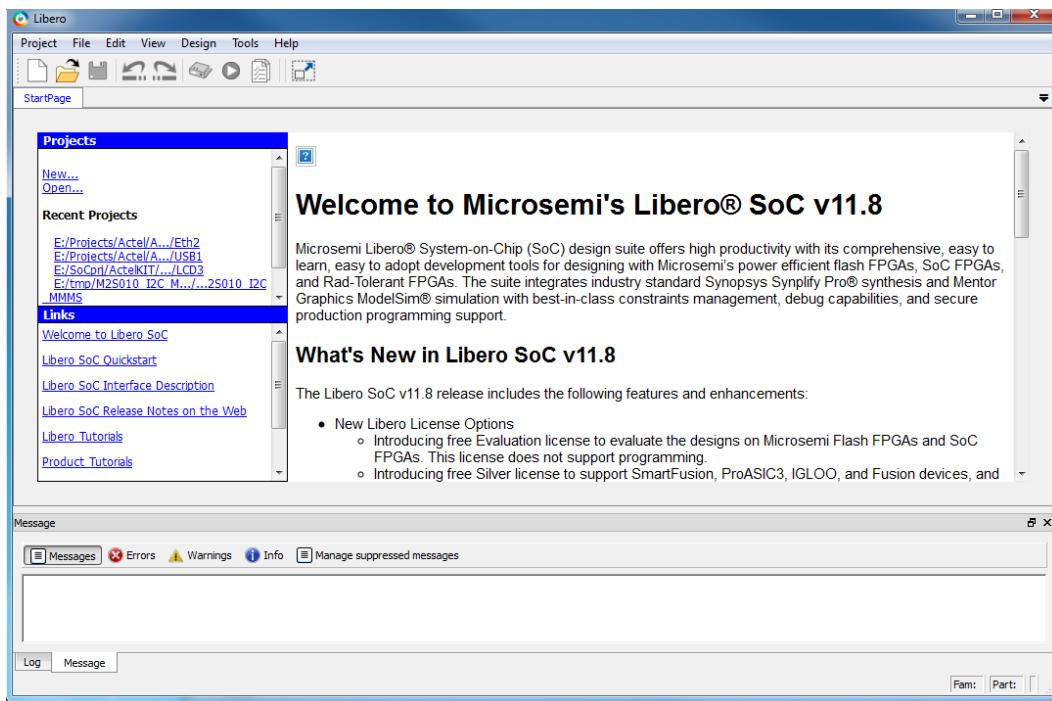


Рис. 1.

В главном меню Libero SoC (рис. 1) выполните команду **Project/New Project**, запустится мастер создания нового проекта. В появившемся окне укажите название проекта, например M2S010\_int, место расположения нашего проекта на диске и предпочтаемый язык проектирования аппаратуры – Verilog или VHDL. Опцию Enable block creation устанавливать не нужно. Нажмите кнопку «Next» (рис. 2).

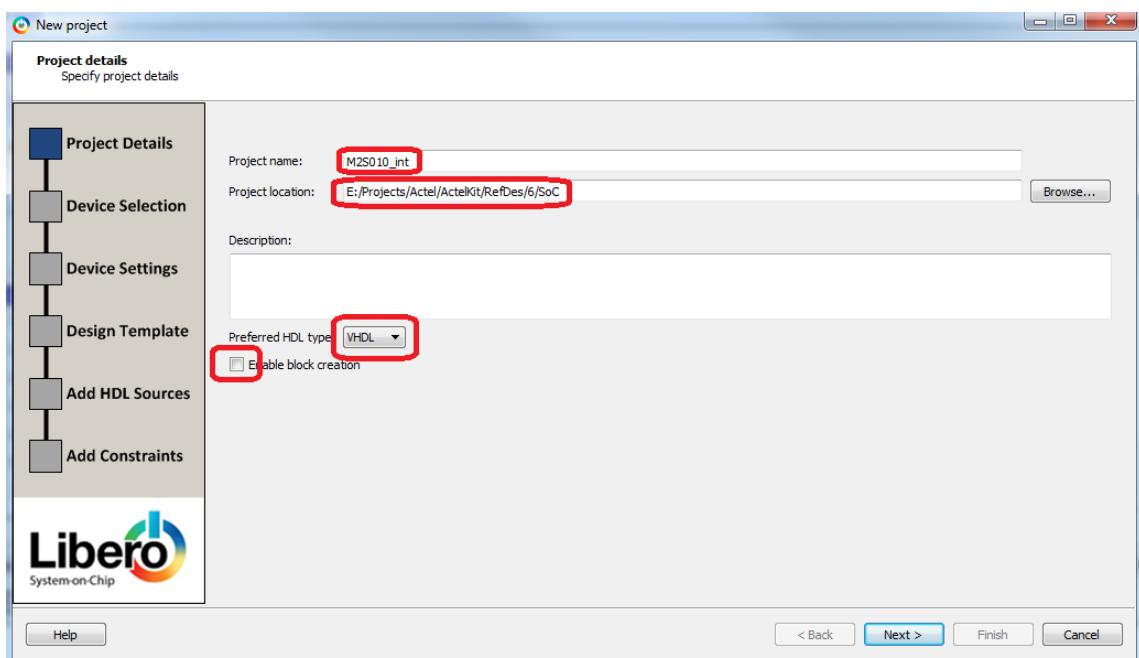


Рис. 2.

В появившемся окне «Device selection» выбором желаемых параметров в выпадающих списках укажите PartNumber микросхемы, с которой будем работать. При работе с отладочным комплектом SF2-Junior-KIT необходимо выбрать вариант M2S010-TQ144. После чего нажмите «Next».

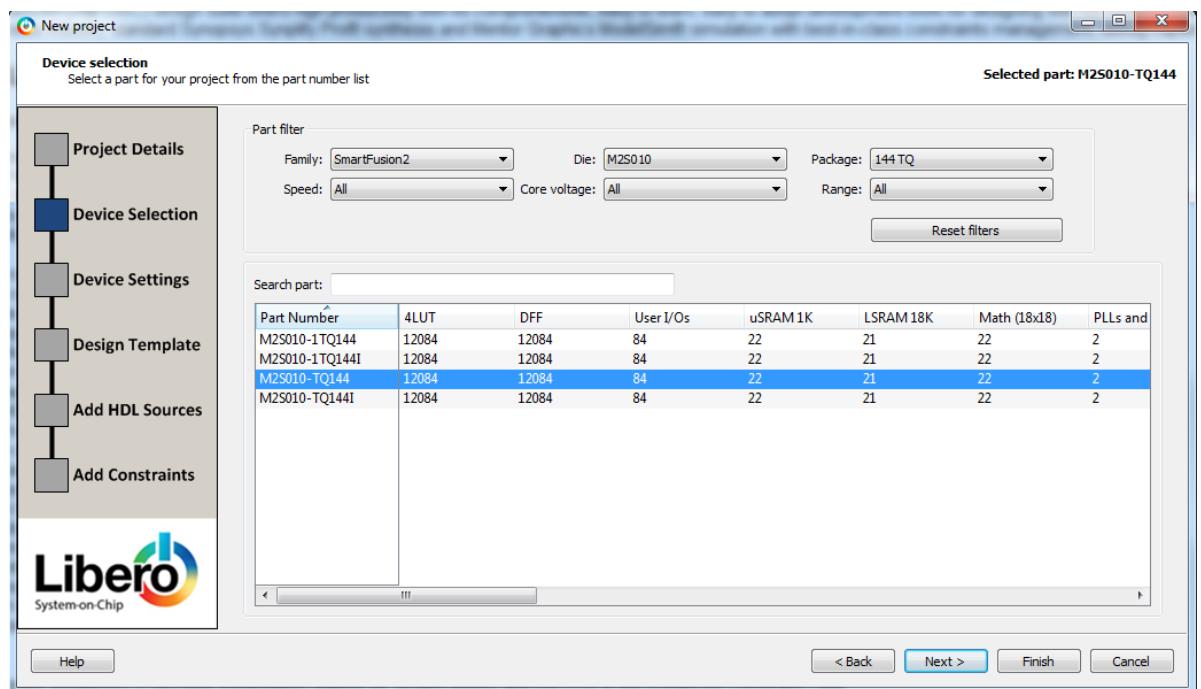


Рис. 3.

В следующем окне выберите настройки стандарта ввода-вывода по умолчанию LVCMOS 2.5V, напряжение питания PLL 2.5 V и задержку старта микросхемы после сигнала Reset 100 ms. Нажмите кнопку «Next» (рис. 4).

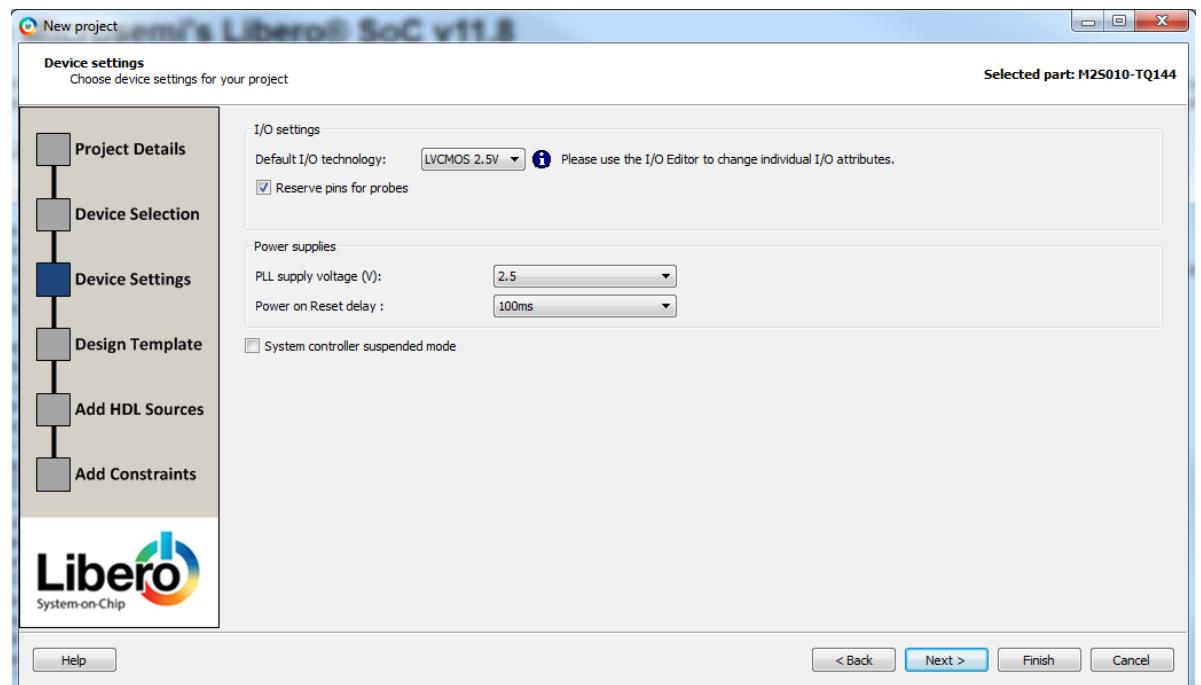


Рис. 4.

В следующем появившемся окне предлагается выбрать мастер, который будет использоваться для настройки микроконтроллерной подсистемы. Выберем пункт «Create a microcontroller (MSS) based design», после чего нажмем «Next» (рис. 5).

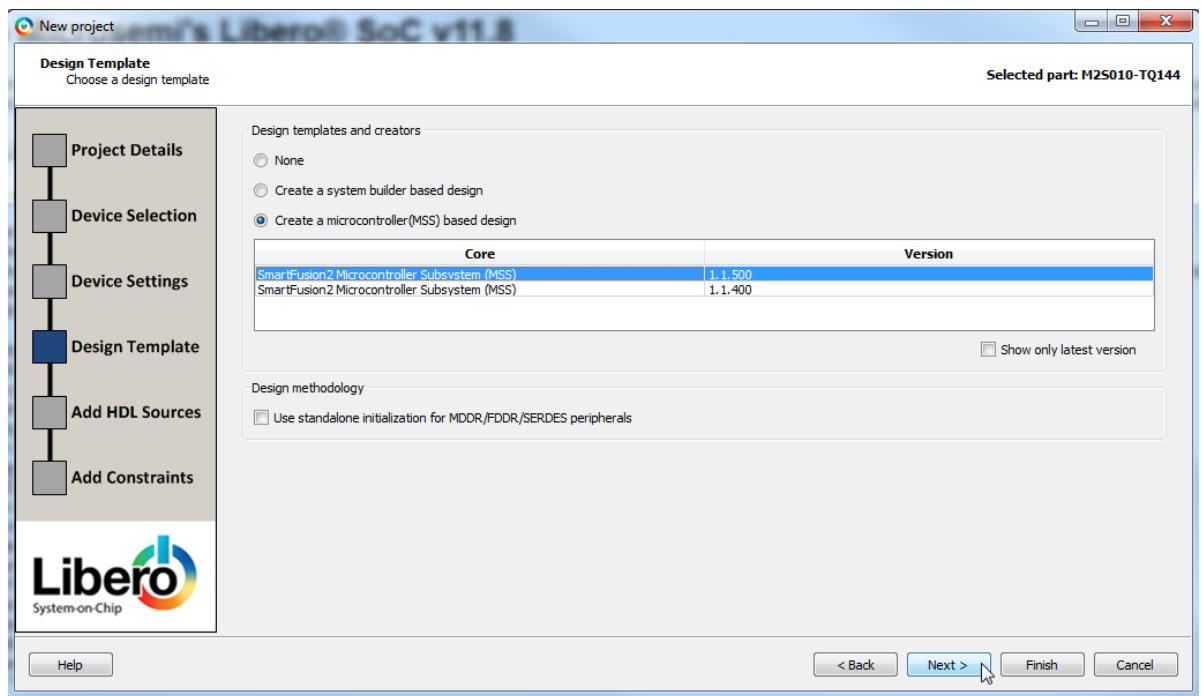


Рис. 5.

В следующих появляющихся окнах ничего не меняем, просто нажимаем «Next» до появления окна показанного на рис. 6. При выборе способа установки проектных ограничений нажмите кнопку «Use Enhanced Constraint Flow».

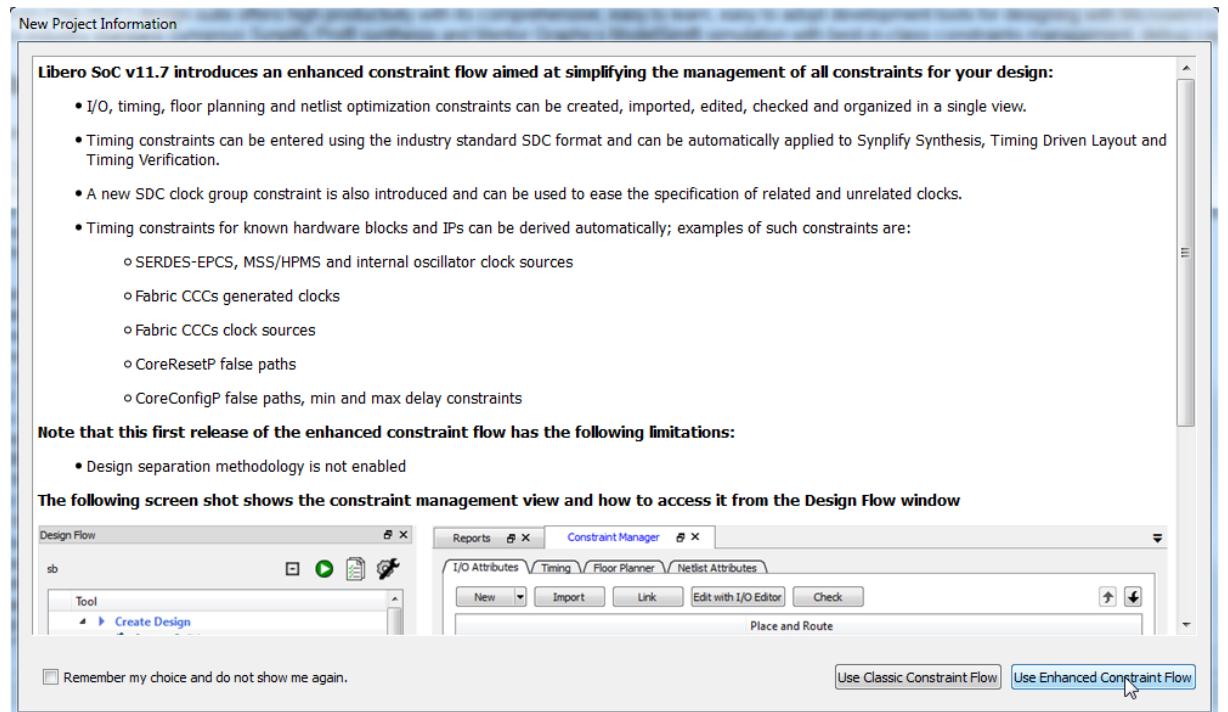


Рис. 6.

В результате выполнения описанных действий появится окно утилиты SmartDesign, в котором будет находиться один компонент M2S010\_int\_MSS\_0 – микроконтроллерная подсистема СнК SmartFusion2 (рис. 7).

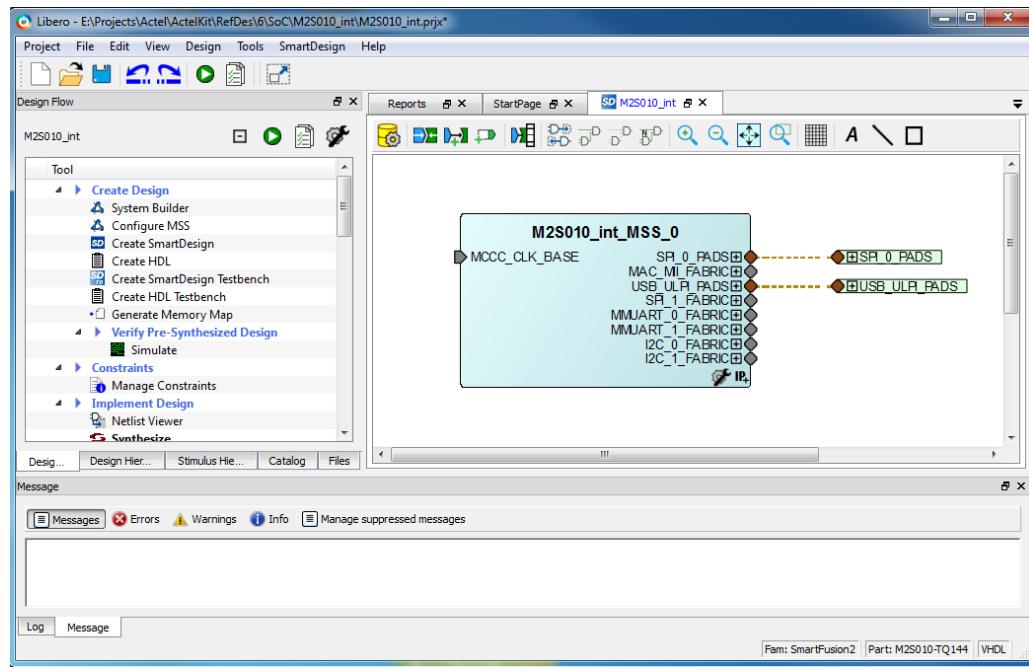


Рис. 7.

Созданный мастером компонент M2S010\_int\_MSS\_0 отражает состояние настроек микроконтроллерной подсистемы «по умолчанию». Необходимо изменить эти настройки в соответствии с задачами, решаемыми нашим приложением.

Настроим блоки архитектуры MSS, для этого дважды щелкнем на компоненте M2S010\_int\_MSS\_0. Откроется окно настроек микроконтроллерной подсистемы (рис. 8).

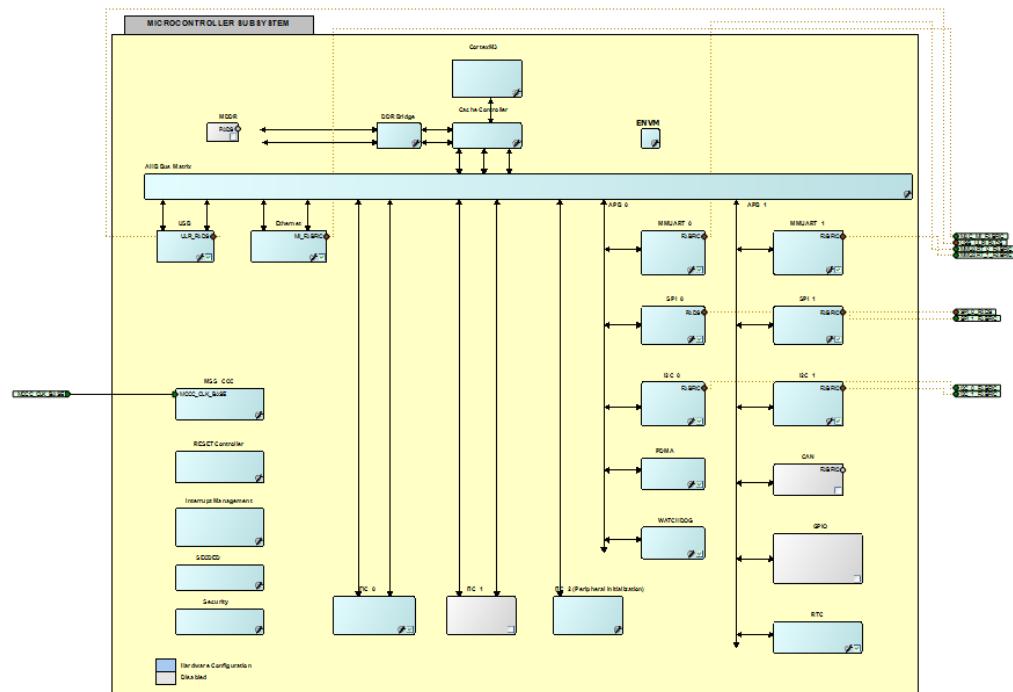


Рис. 8.

Из всего ряда доступных компонентов оставим включенными блоки UART\_0, GPIO. Все остальные компоненты MSS, а именно USB, Ethernet, MMUART\_1, SPI\_0, SPI\_1, I2C\_0, I2C\_1, PDMA, CAN, WatchDog, RTC, FIC\_1 в разрабатываемом проекте задействованы не будут, их необходимо отключить, т. е. снять галочку в правом нижнем углу перечисленных компонентов (рис. 9).

Теперь настроим используемые в нашем проекте блоки микроконтроллерной подсистемы. Начнем с контроллера сброса. Для этого дважды щелкнем на блоке RESET Controller. Выберем опции, представленные на рис. 10.

Настроим контроллер универсального приемопередатчика MMUART\_0. В появившемся окне выберем следующие опции (рис. 11):

- Duplex Mode: Full Duplex.
- Async/Sync Mode: Asynchronous.
- Use Modem Interface: снять галочку
- RHD: Fabric
- TXD: Fabric

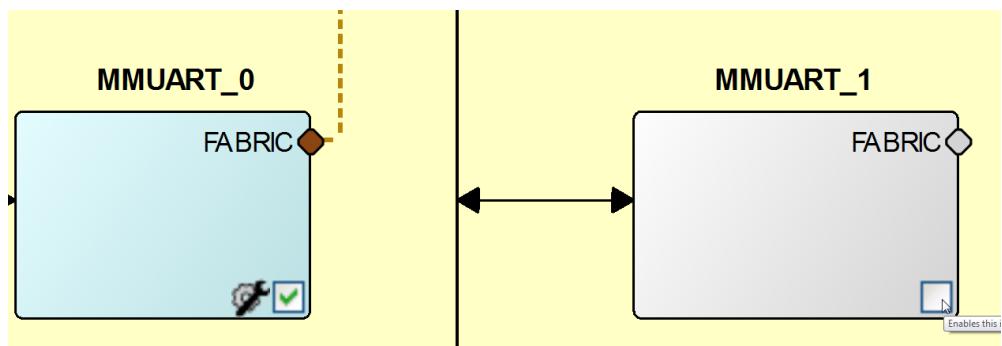


Рис. 9.

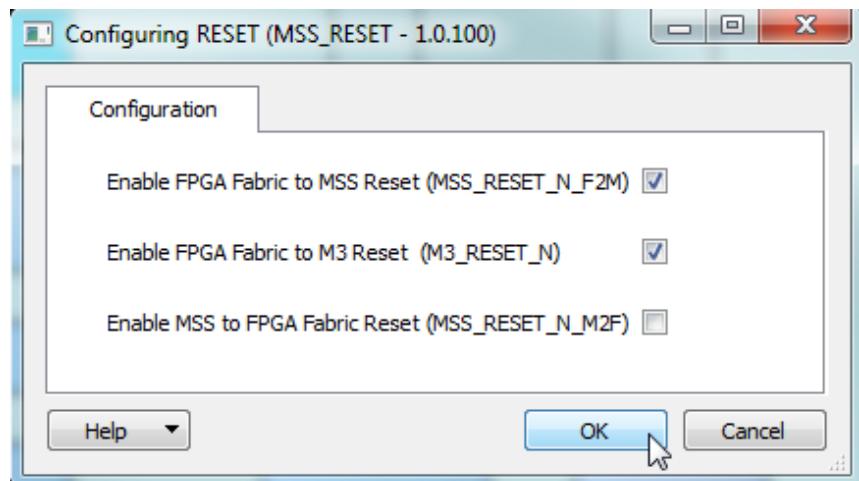


Рис. 10.

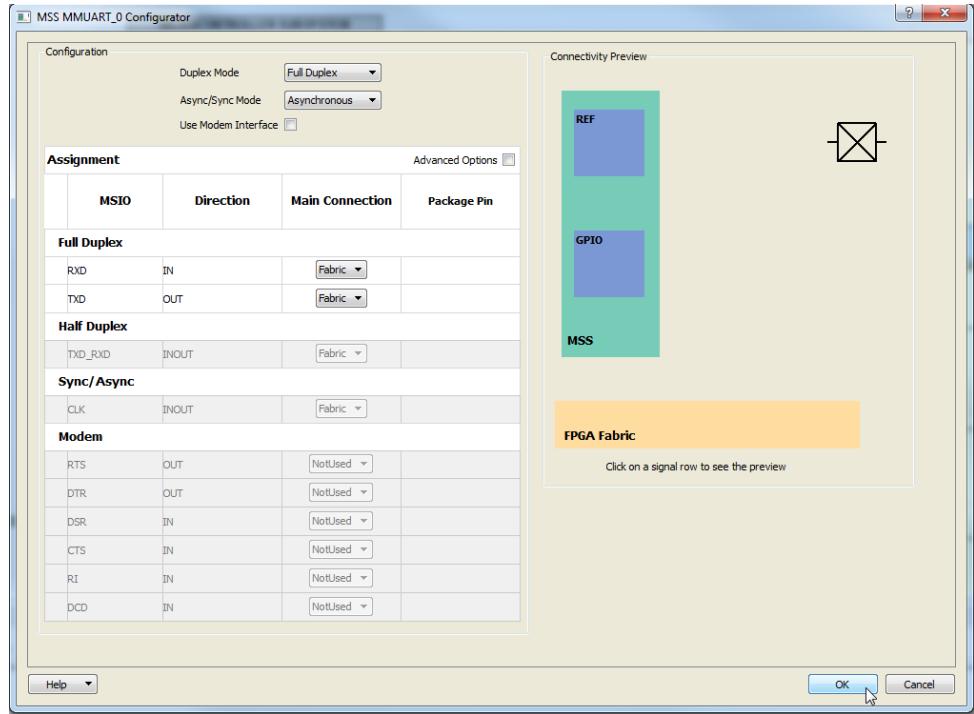


Рис. 11.

Настройки системы формирования сигналов тактирования микроконтроллерной подсистемы СнК SmartFusion2 представлены на рис. 12.

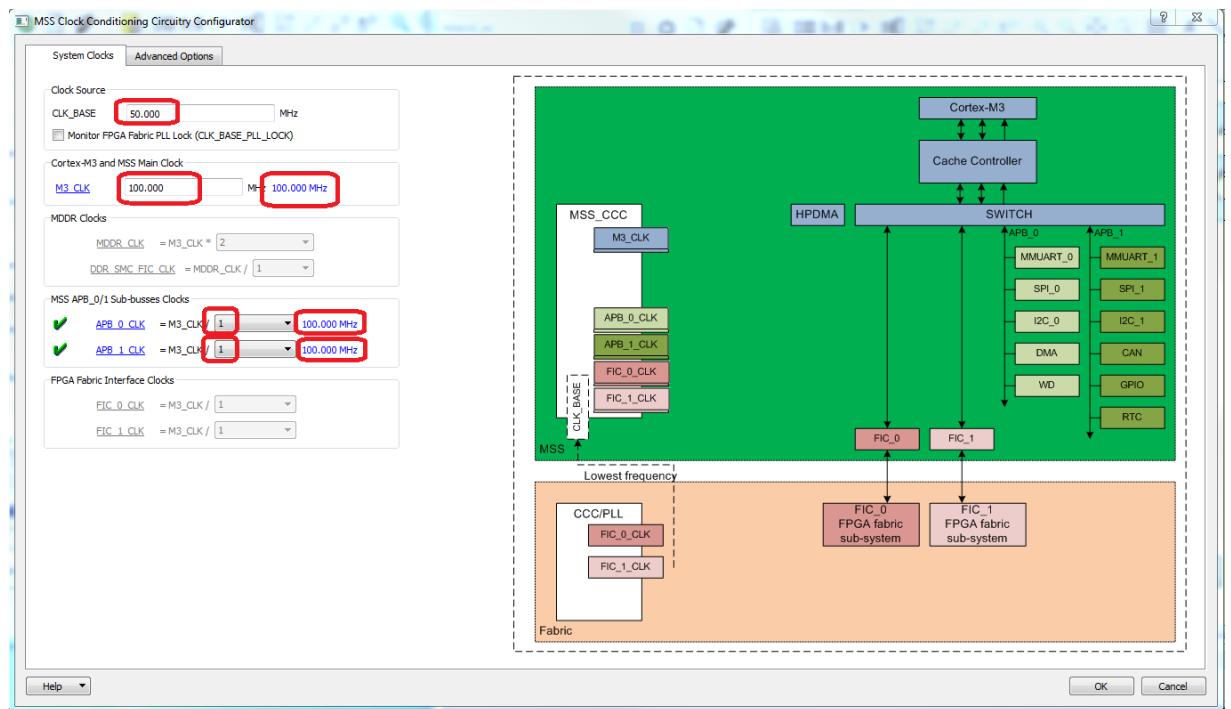


Рис. 12.

Настройки контроллера прерываний MSS SmartFusion2 представлены на рис. 13.

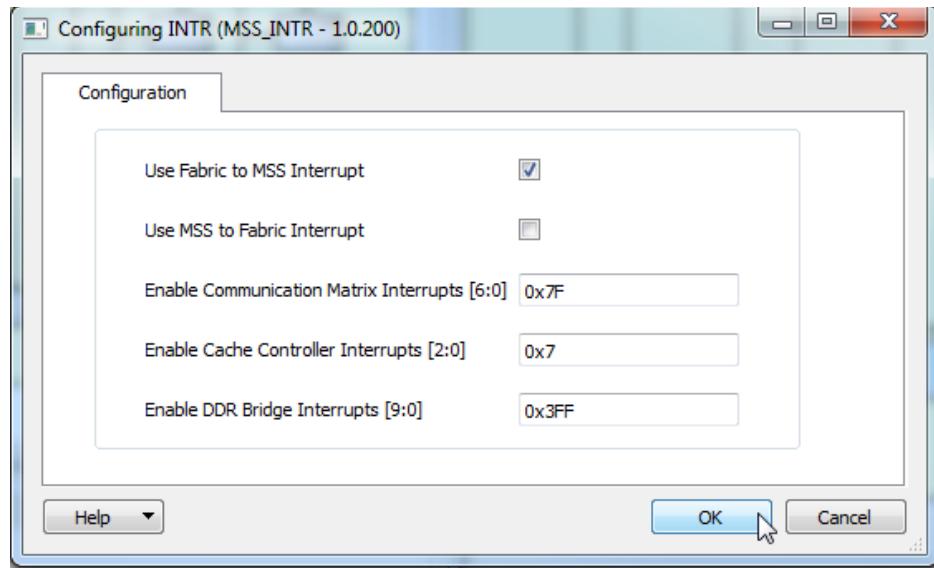


Рис. 13.

Настройки блока контактов ввода-выводов общего назначения микроконтроллерной подсистемы MSS GPIO представлены на рис. 14.

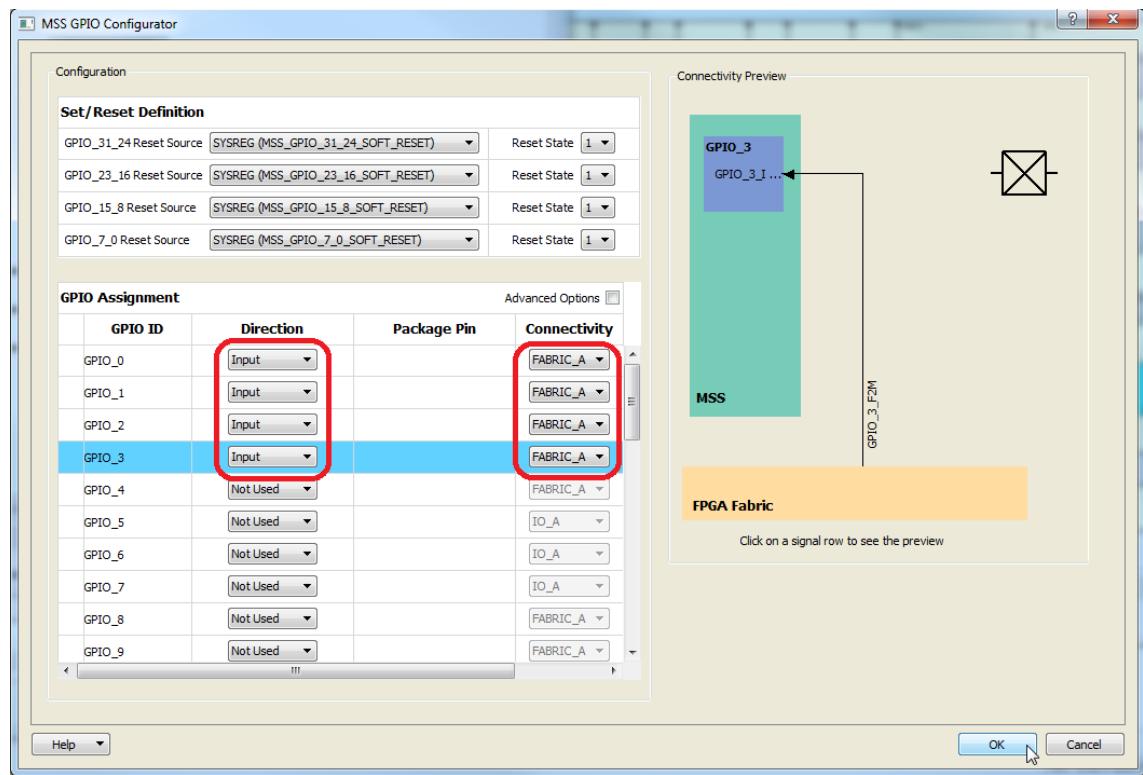


Рис. 14.

В результате описанных действий окно настроек микроконтроллерной подсистемы примет вид, представленный на рис. 15.

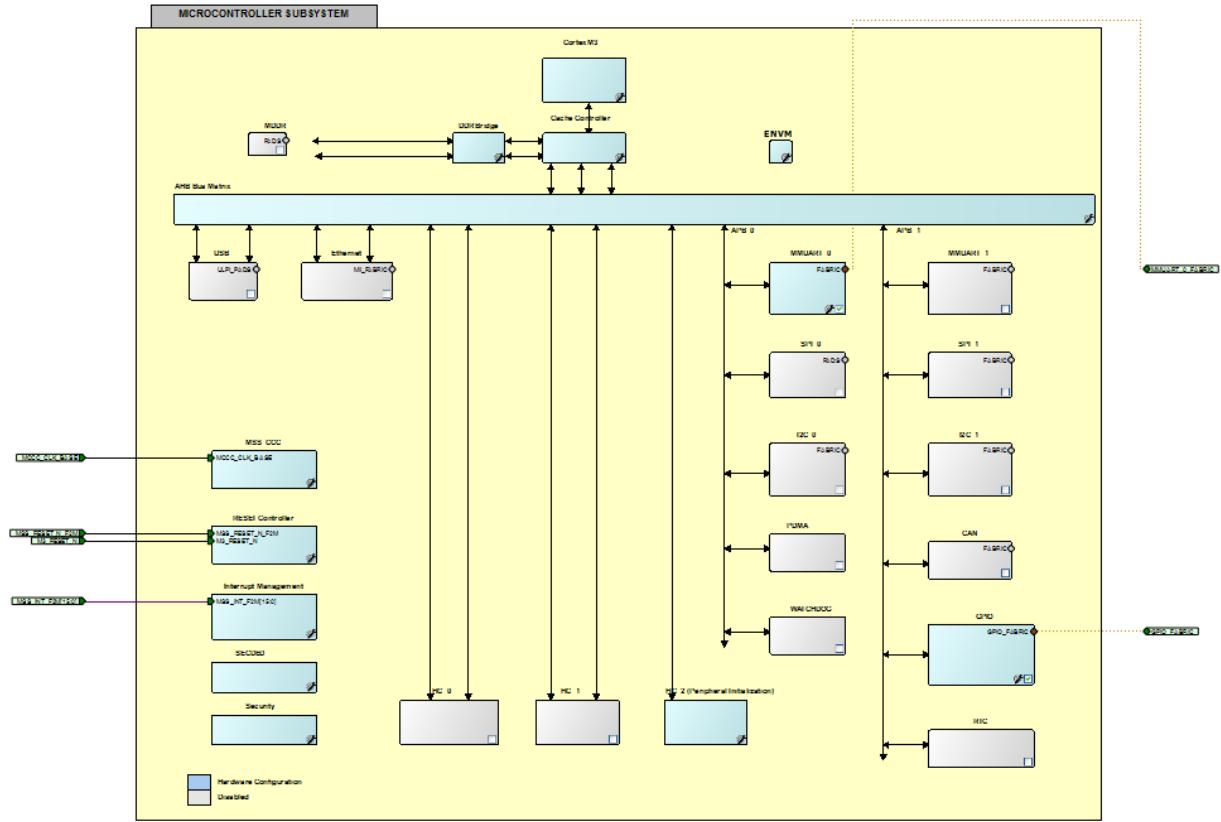


Рис. 15.

Сохраним изменения и вернемся во вкладку M2S010\_int редактора SmartDesign. Внешний вид компонента микроконтроллерной подсистемы изменился – в правом верхнем углу появился восклицательный знак на желтом фоне. Это означает, что свойства компонента изменились и его необходимо обновить. Для обновления необходимо щелкнуть по нему правой кнопкой мыши и в появившемся меню выбрать команду **Update Instance(s) with Latest Component...** (рис. 16).

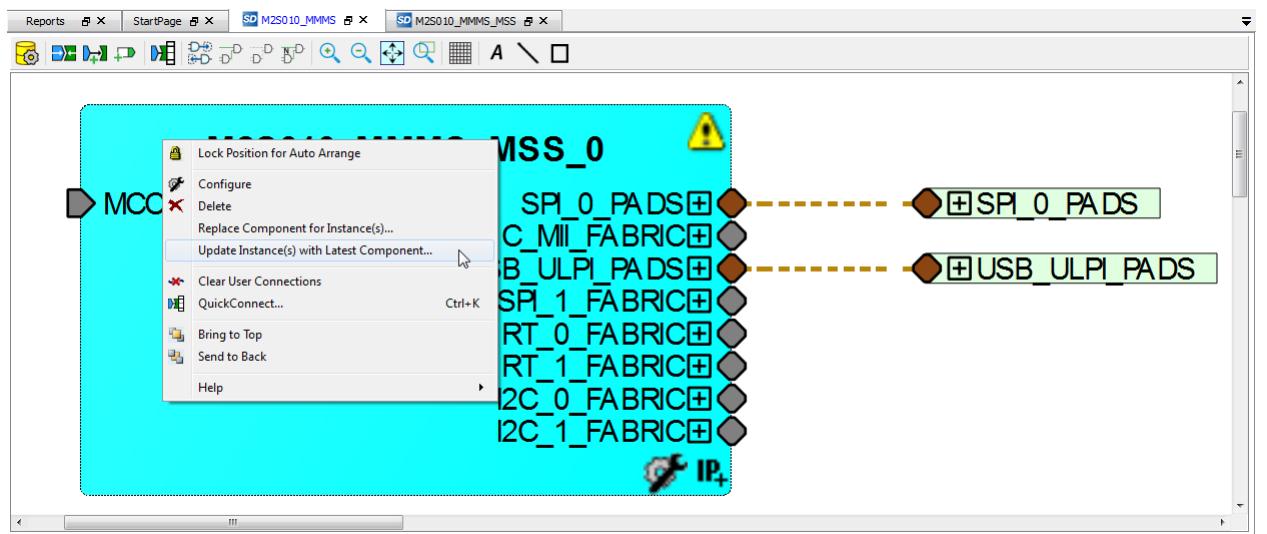


Рис. 16.

После обновления компонент должен принять вид, подобный представленному на рис. 17.

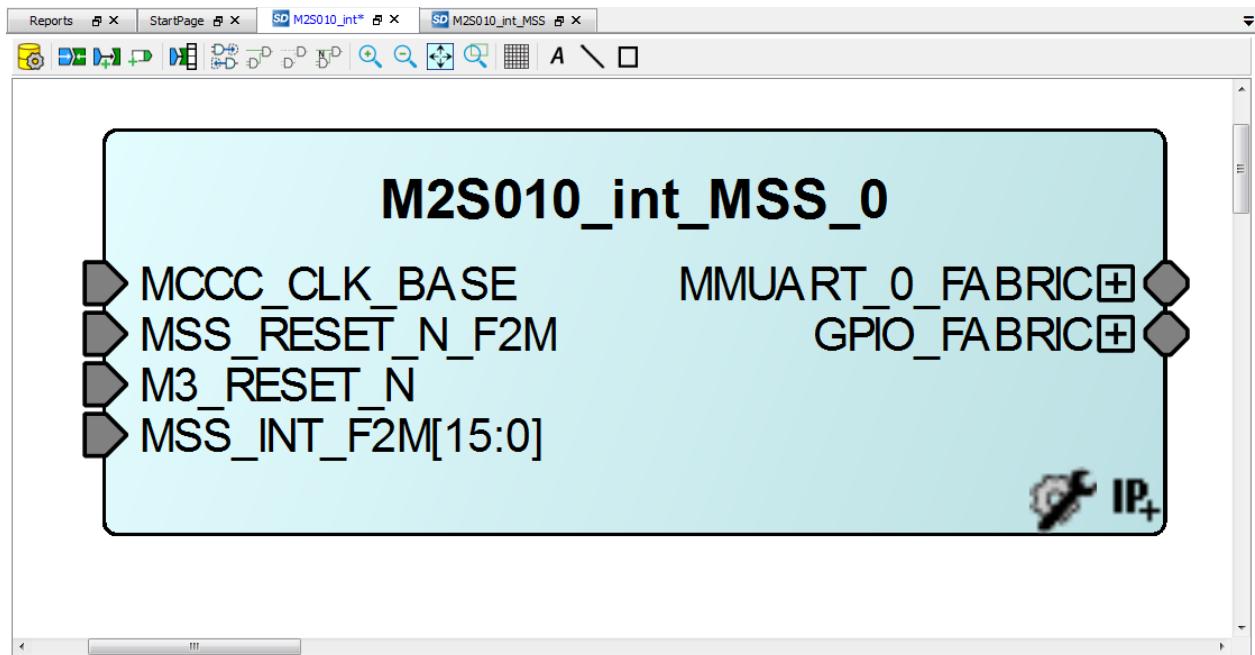


Рис. 17.

Теперь в дополнение к компоненту микроконтроллерной подсистемы из стандартного каталога Libero SoC на рабочее поле проекта нужно добавить IP-ядра и компоненты, отвечающие за тактирование, системный сброс и генерацию событий.

Для реализации описанной выше функциональности проекта необходимы ядра и компоненты стандартной библиотеки Libero SoC, указанные в таблице 1.

Таблица 1.  
Ядра и компоненты стандартного каталога Libero SoC, используемые в проекте.

№ п/п	Раздел стандартного каталога Libero SoC	Название ядра/компонента в каталоге Libero SoC	Название в проекте	Количество в проекте	Назначение
1	Processors	SmartFusion2 Microcontroller Subsystem (MSS)	M2S010_ int_MSS_0	1	Конфигуратор микроконтроллерной подсистемы MSS SmartFusion2
2	Macro Library	SYSRESET	SYSRESET_0	1	Генератор сигнала «Сброс»
3	Clock & Management	Chip Oscillators	OSC_0	1	Источник сигнала тактирования

Для использования нужного компонента в проекте необходимо перейти во вкладку Catalog, раскрыть нужный раздел каталога и мышью перетащить компонент на рабочее поле проекта (рис. 18).

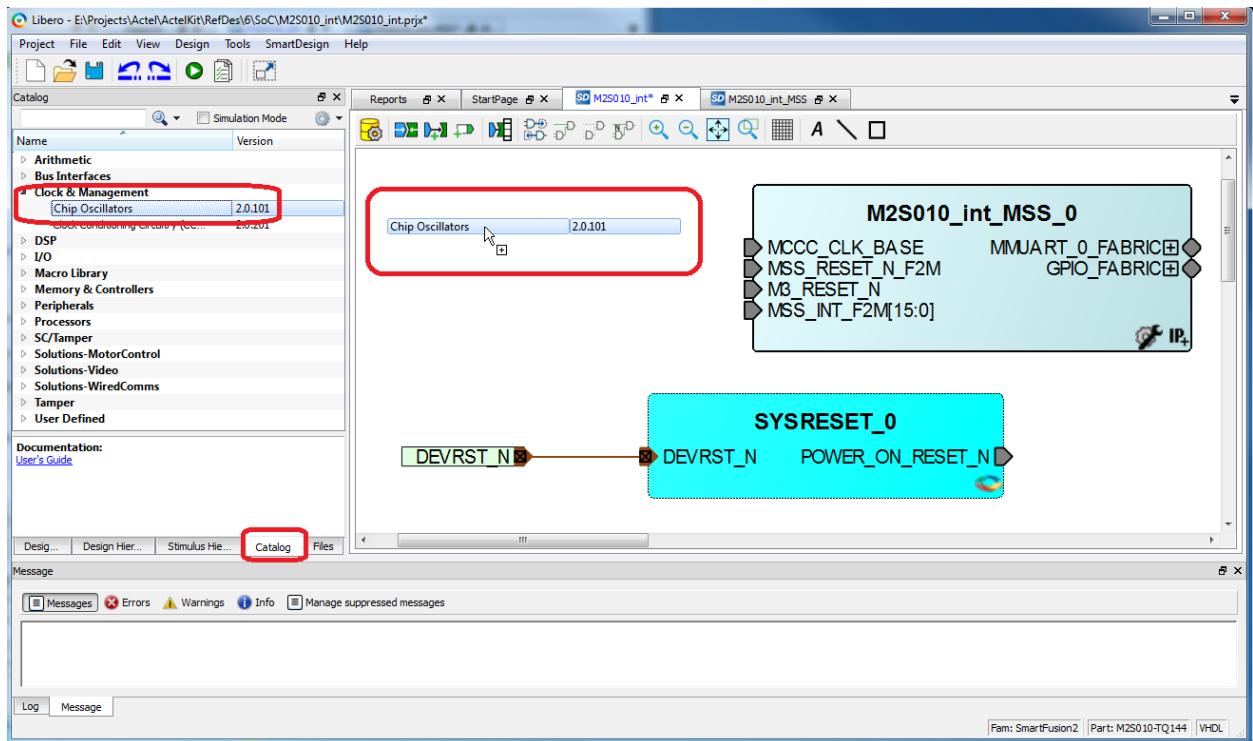


Рис. 18.

Параметры настройки источника сигнала тактирования приведены на рис. 19. Будем использовать внутренний тактовый RC-генератор опорных колебаний на 50 МГц.

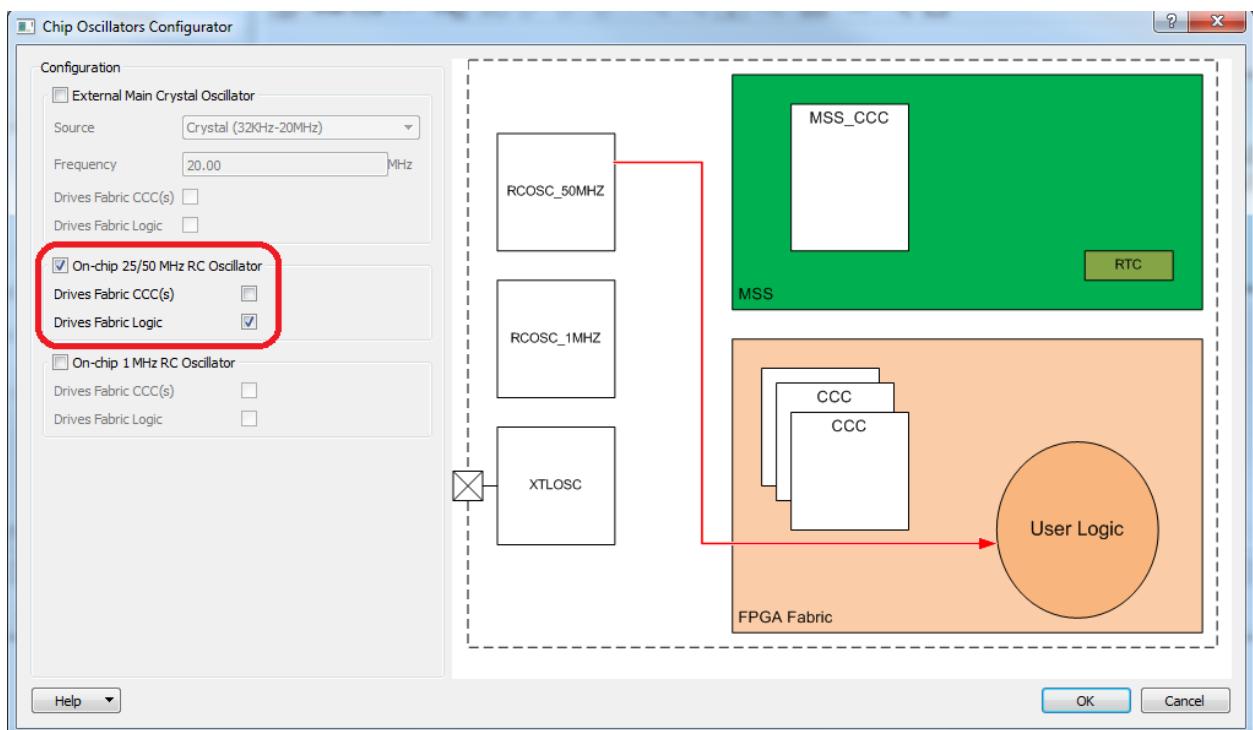


Рис. 19.

Теперь необходимо добавить в проект счетчик и генератор коротких импульсов имплементируемых в матрице FPGA Fabric на основе описания на языке VHDL. Для этого выполним команду основного меню File > New > HDL (рис. 20).

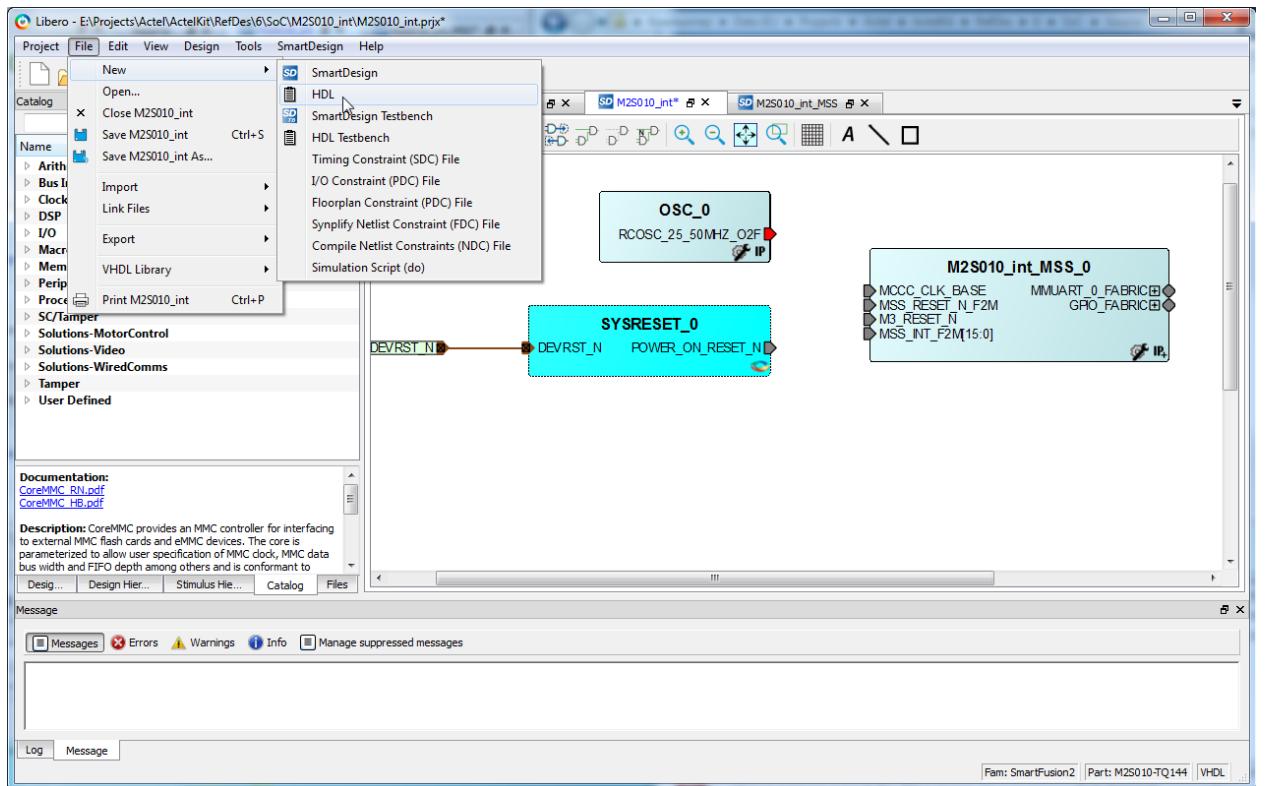


Рис. 20.

В появившемся окне укажем язык проектирования аппаратуры, на котором будем создавать описание и введем имя нового компонента «Pulser» (рис. 21).

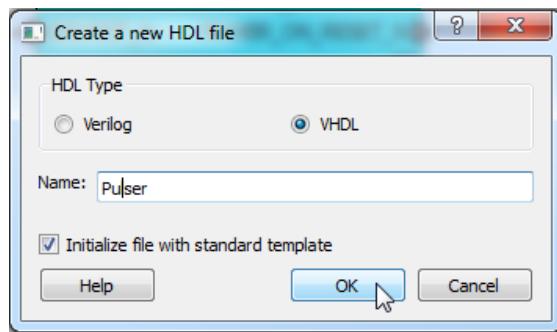


Рис. 21.

В результате описанных действий мастер создаст шаблон требуемого блока.

```

1 ----- Company: <Name>
2 -- File: Pulser.vhd
3 --
4 -- File history:
5 -- <Revision number>: <Date>: <Comments>
6 -- <Revision number>: <Date>: <Comments>
7 -- <Revision number>: <Date>: <Comments>
8 --
9 --
10 -- Description:
11 --
12 -- <Description here>
13 --
14 -- Targeted device: <Family::SmartFusion2> <Die::M2S010> <Package::144 TQ>
15 -- Author: <Name>
16 --
17 -----
18 library IEEE;
19 use IEEE.std_logic_1164.all;
20
21 entity Pulser is
22 port (
23   --<port_name> : <direction> <type>;
24   port_name1 : IN std_logic; -- example
25   port_name2 : OUT std_logic_vector(1 downto 0) -- example
26   --<other_ports>;
27 );
28 end Pulser;
29
30 architecture architecture_Pulser of Pulser is
31 begin
32   -- signal, component etc. declarations
33   signal signal_name1 : std_logic; -- example
34   signal signal_name2 : std_logic_vector(1 downto 0); -- example
35
36 begin
37   -- architecture body
38 end architecture_Pulser;
39
40

```

Рис. 22.

Заполним описание компонента на языке VHDL содержимым файла Pulser.vhd из папки Source архива [прилагаемого](#) к данному описанию. Проверим введенный текст на наличие синтаксических ошибок (рис. 23).

```

1 library CheckHDLfile;
2 use IEEE.std_logic_1164.all;
3 use IEEE.std_logic_arith.all;
4 use IEEE.std_logic_unsigned.all;
5
6 entity pulser is
7 port( Clock: in std_logic;
8       Reset: in std_logic;
9       Output: out std_logic);
10 end pulser;
11
12 architecture Behavioral of pulser is
13 signal temp : std_logic_vector(0 to 28);
14 signal t : std_logic;
15 begin
16 process(Clock, Reset)
17 begin
18 if Reset='0' then
19   temp <= "00000000000000000000000000000000";
20   t <= '0';
21 elsif(rising_edge(Clock)) then
22   if temp = "10111101011110000100000000" then    --111010000100100000
23     temp <= "00000000000000000000000000000000";
24     t <= '1';
25   else
26     temp <= temp + 1;
27     t <= '0';
28   end if;
29 end process;
30 Output <= t;
31 end Behavioral;
32

```

Рис. 23.

Аналогично генератору коротких импульсов создадим компонент счетчика тактовых импульсов.

В результате в палитре компонентов окажутся доступными 2 пользовательских компонента: Pulser и Counter (рис. 24).

The screenshot shows the Libero SoC software interface. In the top menu, 'Project' is selected. The 'Design Hierarchy' tab is active, showing a tree structure with 'work' expanded, containing 'pulser (Pulser.vhd)', 'M2S010\_int' (highlighted with a red box), and 'Counter (Counter.vhd)'. Below the hierarchy, the 'Counter.vhd' file is open in the main editor area, displaying VHDL code for a counter. The code defines an entity 'Counter' with a port for 'Clock' and 'Reset', and an output 'Q' as a 27-bit vector. It includes a behavioral process that increments 'Q' on rising edges of 'Clock' if 'Reset' is low. The code also contains a synthesis loop for the output 'Q'.

```

library IEEE;
use IEEE.std_logic_1164.all;
use IEEE.std_logic_arith.all;
use IEEE.std_logic_unsigned.all;

entity Counter is
    port(Clock : in std_logic;
         Reset : in std_logic;
         Tcnt : out std_logic);
end Counter;

architecture behavioral of Counter is
begin
    process(Clock, Reset)
    begin
        if (Reset = '0') then
            Qaux <= (others => '0');
        elsif (Clock'event and Clock = '1') then
            Qaux <= Qaux + 1;
        end if;
    end process;
    Q <= std_logic_vector(Qaux);
    process(Qaux)
    variable aux : std_logic;
    begin
        aux := '1';
        for I in 0 to 27 loop
            if (Qaux(I) = '0') then
                aux := '0';
            end if;
        end loop;
        Tcnt <= aux;
    end process;
end behavioral;

```

Рис. 24

Вынесем созданные компоненты на рабочее поле верхнего уровня нашего проекта. И выполним соединения. Выполнить соединения контактов компонентов в Libero SoC можно несколькими способами:

- 1) При нажатой клавише **<Ctrl>** левой кнопкой мыши выделить два или более соединяемых общей цепью контактов, щелкнуть правой кнопкой мыши на одном из контактов и в появившемся меню выбрать Connect (рис. 25);
- 2) Используя быструю клавишу Connection Mode (рис. 26);
- 3) Используя мастер соединений QuickConnect (рис. 27.).

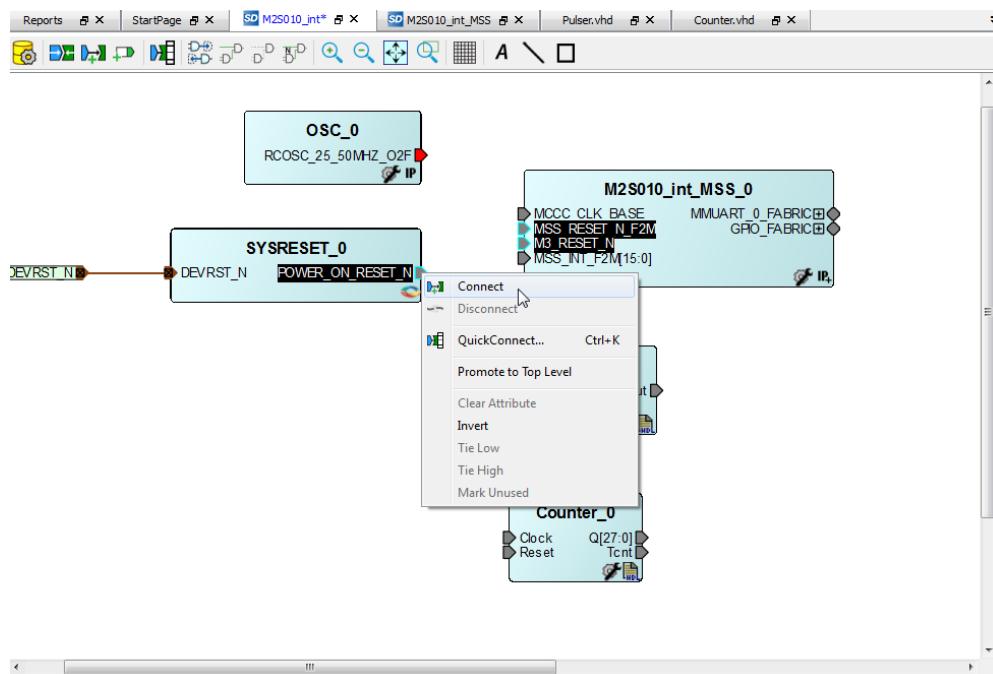


Рис. 25.

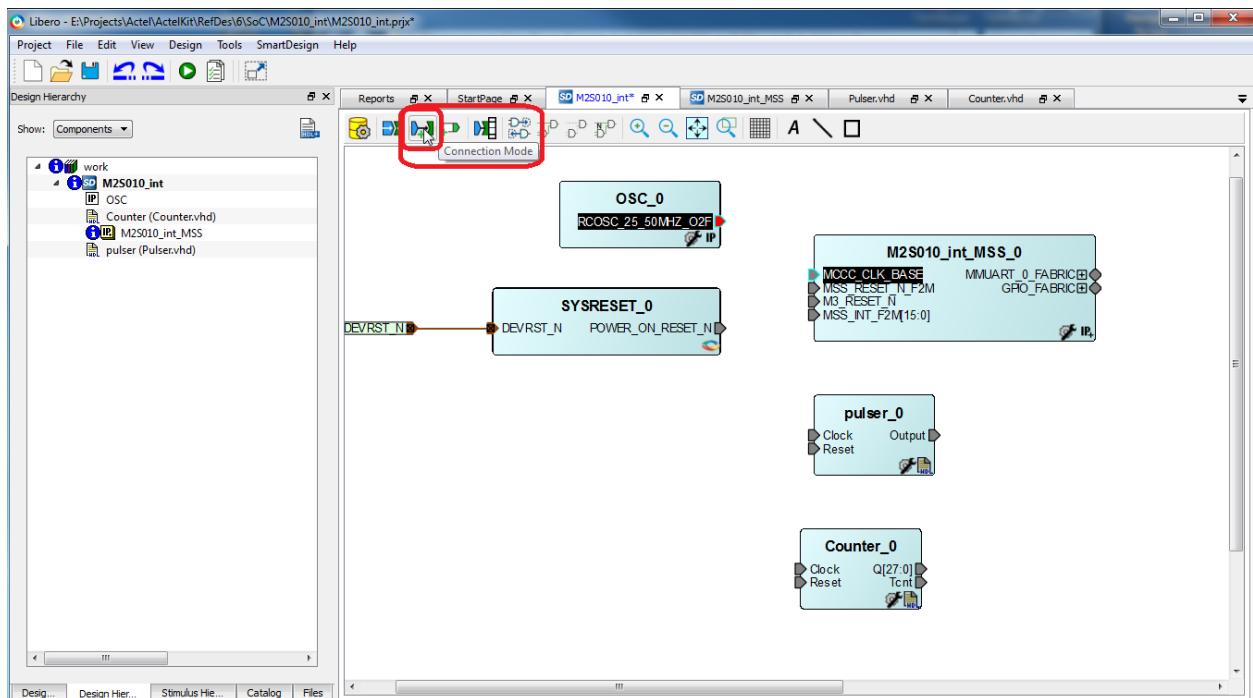


Рис. 26.

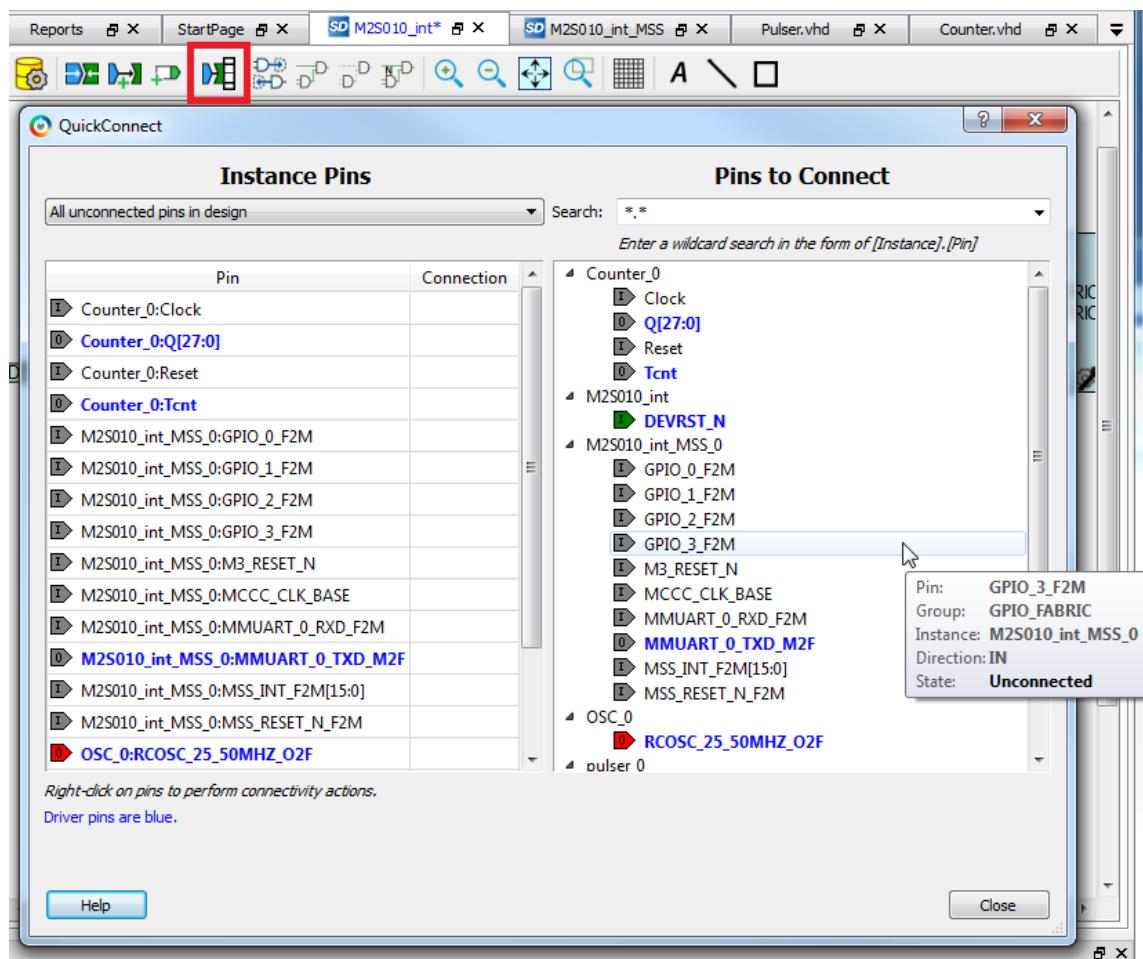


Рис. 27.

Используя описанные выше способы, выполним соединения. Ниже приведен список соединений проекта в формате IP-ядро.Контакт – IP-ядро.Контакт:

1. OSC\_0.RCOSC\_25\_50MHZ\_O2F – M2S010\_int\_MSS\_0.MCCC\_CLK\_BASE.
2. OSC\_0.RCOSC\_25\_50MHZ\_O2F – pulser\_0.Clock.
3. OSC\_0.RCOSC\_25\_50MHZ\_O2F – Counter\_0.Clock.
4. SYSRESET\_0.POWER\_ON\_RESET\_N – M2S010\_int\_MSS\_0.MSS\_RESET\_N\_F2M.
5. SYSRESET\_0.POWER\_ON\_RESET\_N – M2S010\_int\_MSS\_0.M3\_RESET\_N.
6. SYSRESET\_0.POWER\_ON\_RESET\_N – pulser\_0.Reset.
7. SYSRESET\_0.POWER\_ON\_RESET\_N – counter\_0.Reset.
8. Pulser\_0.Output – M2S010\_int\_MSS\_0.MSS\_INT\_F2M[0].
9. counter\_0.Q[27] – M2S010\_int\_MSS\_0.GPIO\_3\_F2M

Для выполнения соединения Pulser\_0.Output – M2S010\_int\_MSS\_0.MSS\_INT\_F2M[0] необходимо предварительно выделить нужный контакт MSS\_INT\_F2M[0] из массива. Для этого щелкнем на контакте M2S010\_int\_MSS\_0.MSS\_INT\_F2M[15:0] правой кнопкой мыши и опишем новое представление массива контактов (рис. 28). Аналогичные действия необходимо выполнить с выходными контактами счетчика Counter\_0.

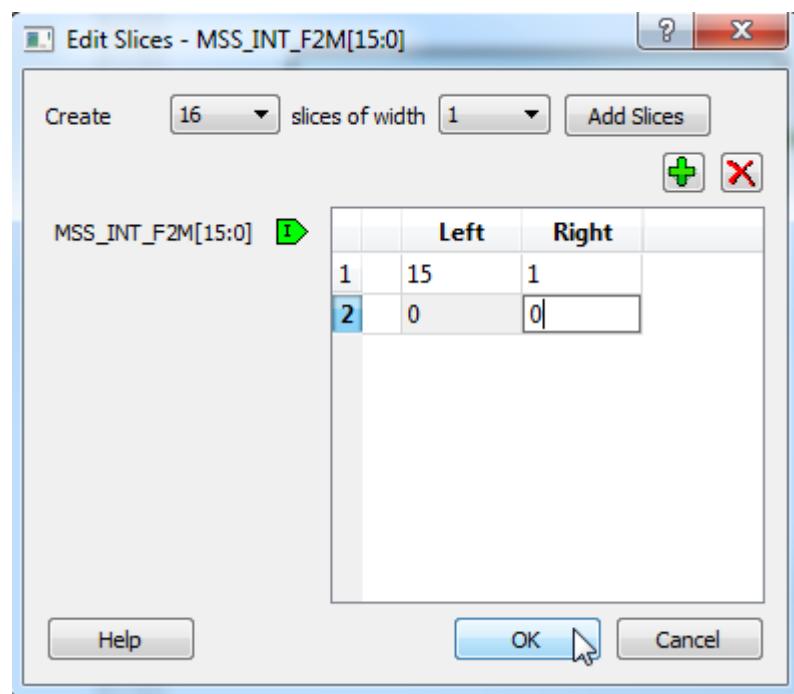


Рис. 28.

Оставшиеся линии прерываний MSS\_INT\_F2M[15:1] подтягиваем «вниз» (рис. 29).

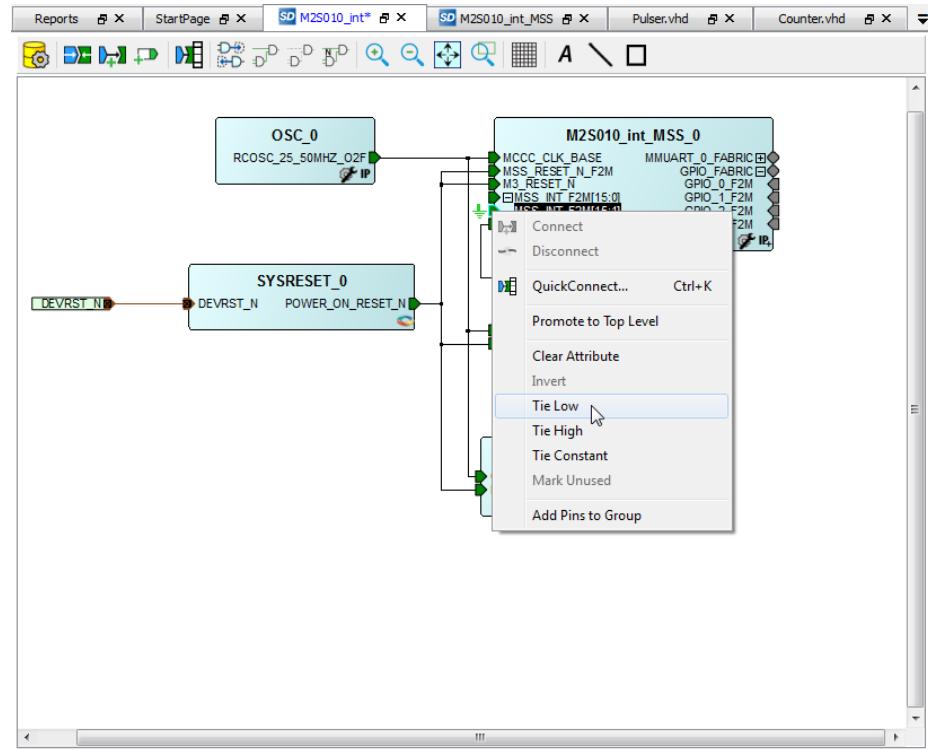


Рис. 29.

Контакты M2S010\_int\_MSS\_0.MMUART\_0\_FABRIC, GPIO\_1 – GPIO3 и Counter\_0.Q[26] выводим на верхний уровень, т.е. щелкаем на контакте правой кнопкой мыши и в появившемся меню выбираем команду **Promote to Top Level**.

Переименуем контакты «GPIO» компонента микроконтроллерной подсистемы и Q[26] счетчика в соответствии с их функциональным назначением для улучшения читабельности схемы. Верхний уровень проекта после выполнения всех подключений представлен на рис. 30.

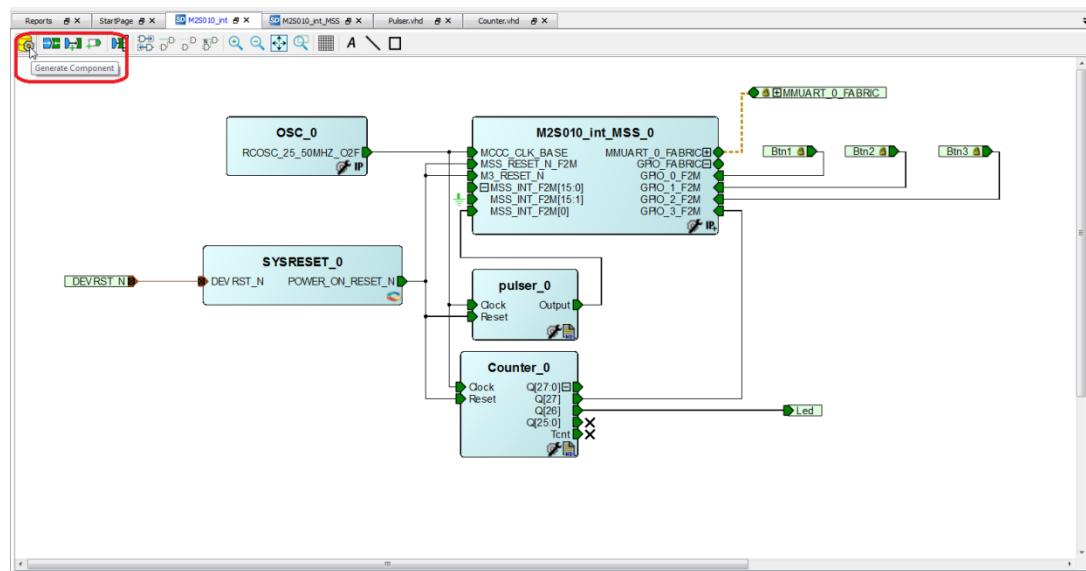


Рис. 30.

Сохраняем изменения и выполняем команду **Generate Component**.

Выполним синтез проекта (рис. 31) и назначим контакты микросхемы входным и выходным сигналам нашего проекта для чего выполним команду **Manage Constraints** (рис. 32), в

появившейся вкладке **Constraints Manager** выполним команду **Edit > Edit with I/O Editor** (рис. 33). В открывшемся окне утилиты I/O Editor назначаем контакты микросхемы M2S010-TQ144 входящим и выходящим сигналам нашего проекта (рис. 34). Сохраним изменения и закроем I/O Editor.

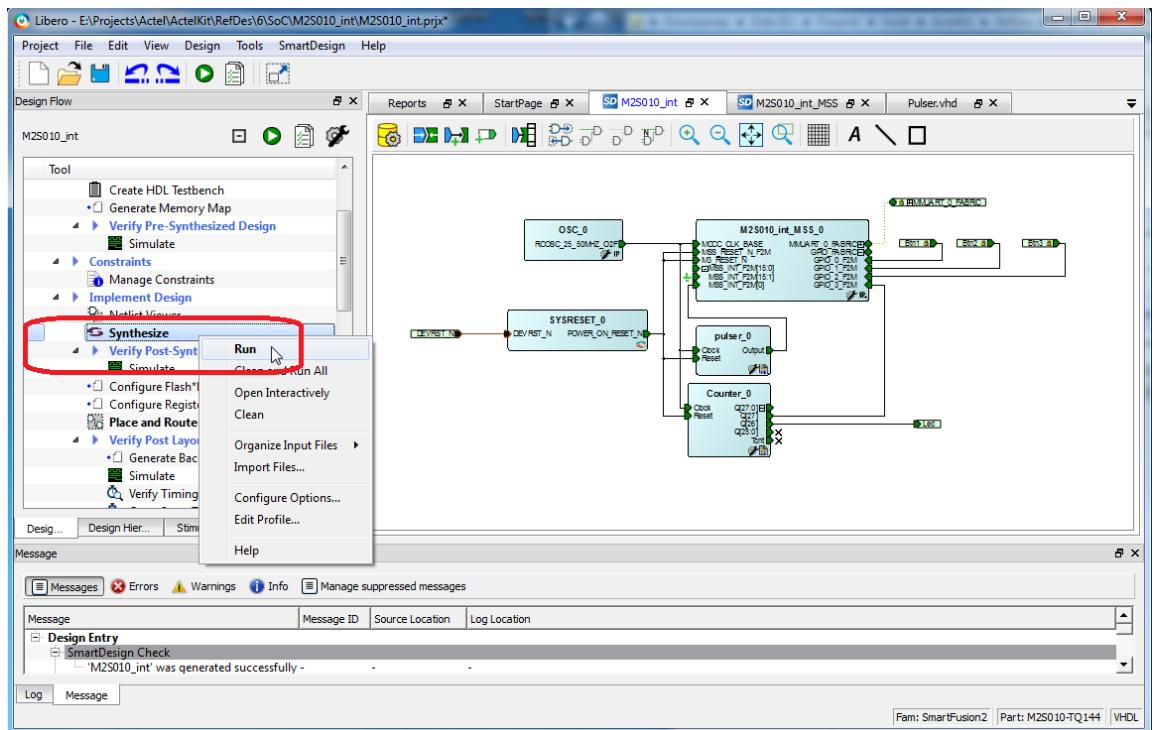


Рис. 31.

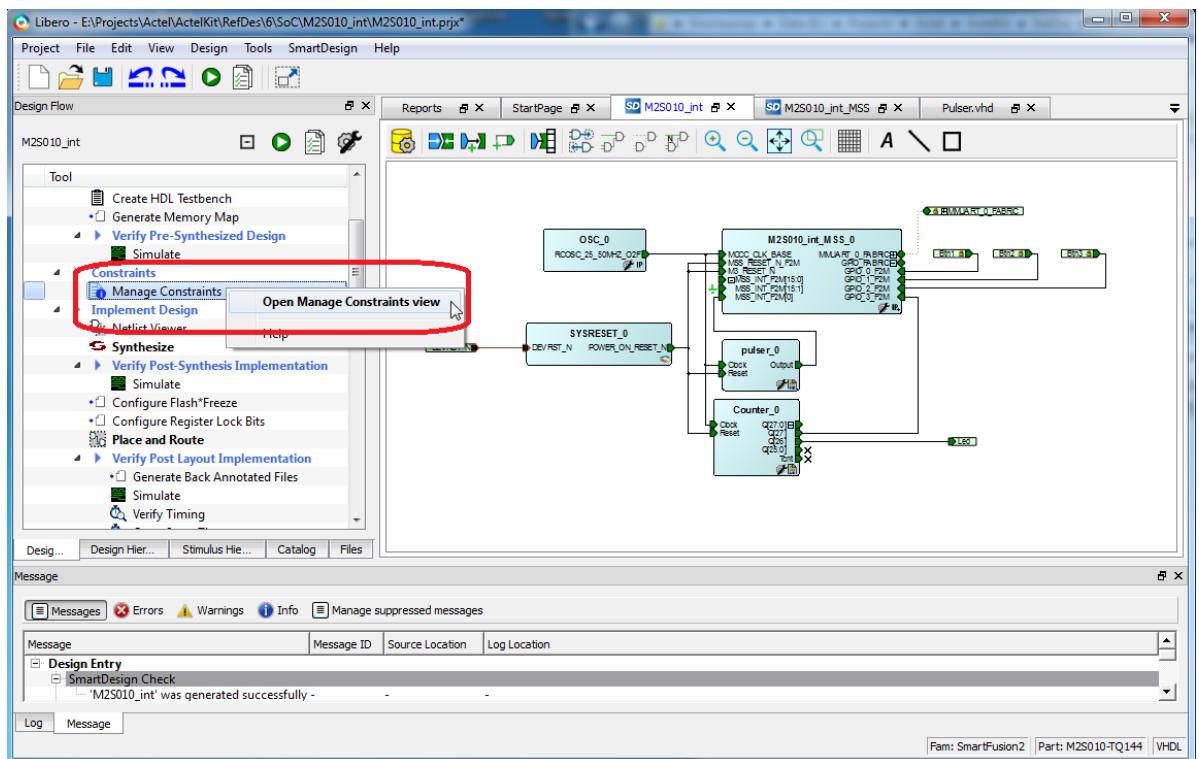


Рис. 32.

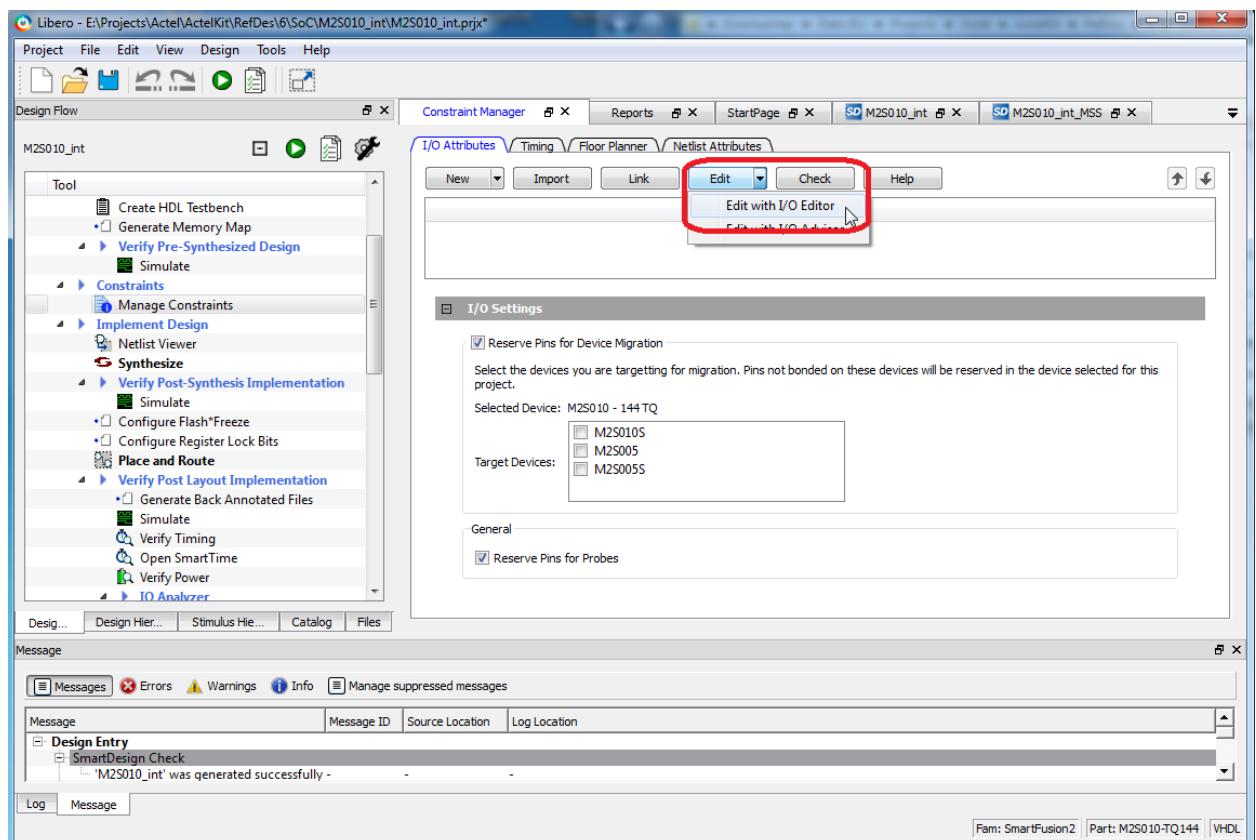


Рис. 33.

The screenshot shows the I/O Editor window titled 'I/O Editor - M2S010\_int'. The window has a menu bar: File, Edit, View, Tools, Help. Below the menu is a toolbar with icons for saving, opening, and editing. The main area contains a table with columns: Port, Port Name, Direction, I/O Standard, Pin Number, Locked, Macro Cell, and Bank Name.

	Port	Port Name	Direction	I/O Standard	Pin Number	Locked	Macro Cell	Bank Name
1		Btn1	Input	LVCMS25	13	<input checked="" type="checkbox"/>	ADLIB:INBUF	Bank7
2		Btn2	Input	LVCMS25	14	<input checked="" type="checkbox"/>	ADLIB:INBUF	Bank7
3		Btn3	Input	LVCMS25	43	<input checked="" type="checkbox"/>	ADLIB:INBUF	Bank4
4		DEVRST_N	Input	--	72	<input checked="" type="checkbox"/>	ADLIB:SYSRESET	--
5		Led	Output	LVCMS25	63	<input checked="" type="checkbox"/>	ADLIB:OUTBUF	Bank4
6		MMUART_0_RXD_F2M	Input	LVCMS25	9	<input checked="" type="checkbox"/>	ADLIB:INBUF	Bank7
7		MMUART_0_TXD_M2F	Output	LVCMS25	10	<input checked="" type="checkbox"/>	ADLIB:OUTBUF	Bank7

Рис. 34.

Выполним команду **Configure Firmware Cores** во вкладке **Design Flow** (рис. 35).

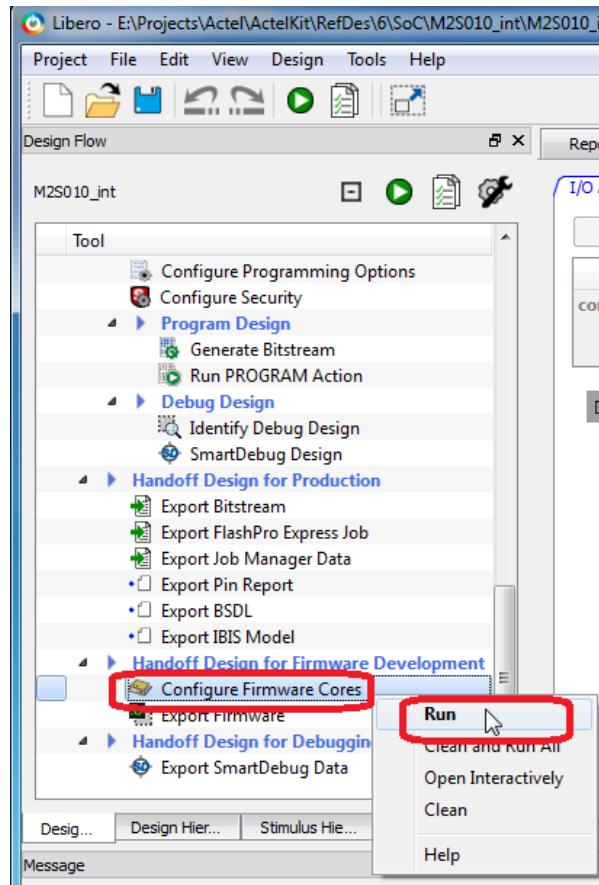


Рис. 35.

В случае если во вкладке **DESIGN\_FIRMWARE** один или несколько драйверов отображены курсивом, необходимо нажать кнопку **Download All Firmware** (рис. 36) для загрузки отсутствующих драйверов по сети Internet с сайта производителя (для успешной загрузки компьютер должен быть подключен к сети Интернет).

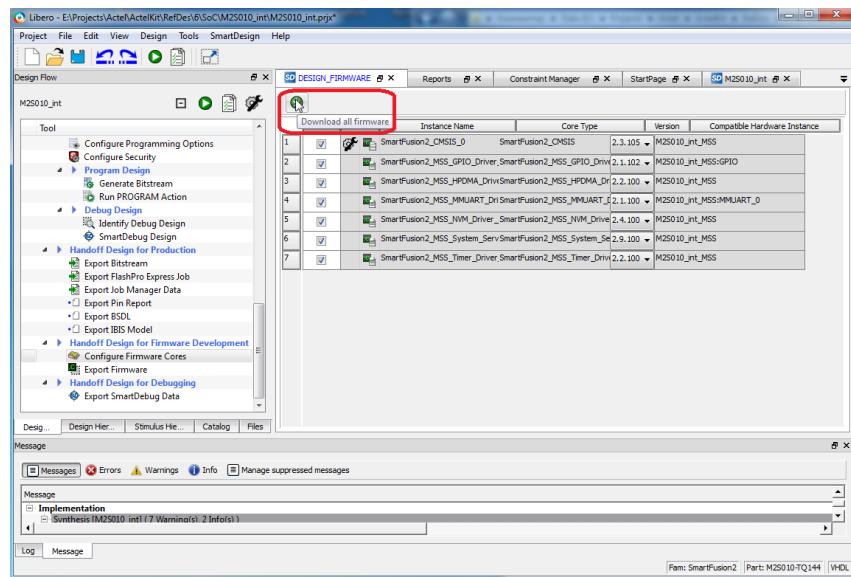


Рис. 36.

Теперь создадим проект встроенного программного обеспечения (ВПО) процессора ARM Cortex-M3 выполнив команду **Export Firmware**. В появившемся окне выберем среду разработки **SoftConsole 3.4** и опцию **Create software project including hardware configuration and firmware drivers** (рис. 37).

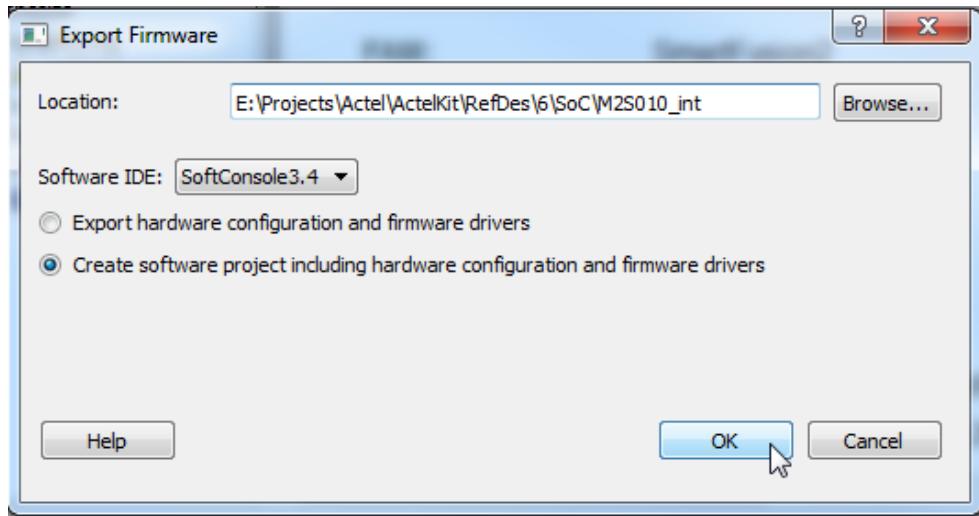


Рис. 37.

# Разработка встроенного программного обеспечения

Переходим к разработке кода встраиваемого приложения. Встроенное программное обеспечение нашего проекта должно выполнить следующие шаги:

- 1) Проинициализировать драйверы всех используемых в проекте аппаратных блоков;
- 2) Назначить процедуры обработчики прерываний;
- 3) Во обработчиках прерываний выполнить отправку в UART информационного сообщения о произошедшем событии;
- 4) Очистить прерывание.

Откроем созданный проект ВПО в среде SoftConsole v.3.4 (рис. 38). Заменим содержимое файла main.c и исходным кодом, содержащимся в одноименном файле из каталога /Source архива с файлами проекта, прилагаемом к данному руководству (можно заменить файл в проекте файлом из архива) (рис. 39).

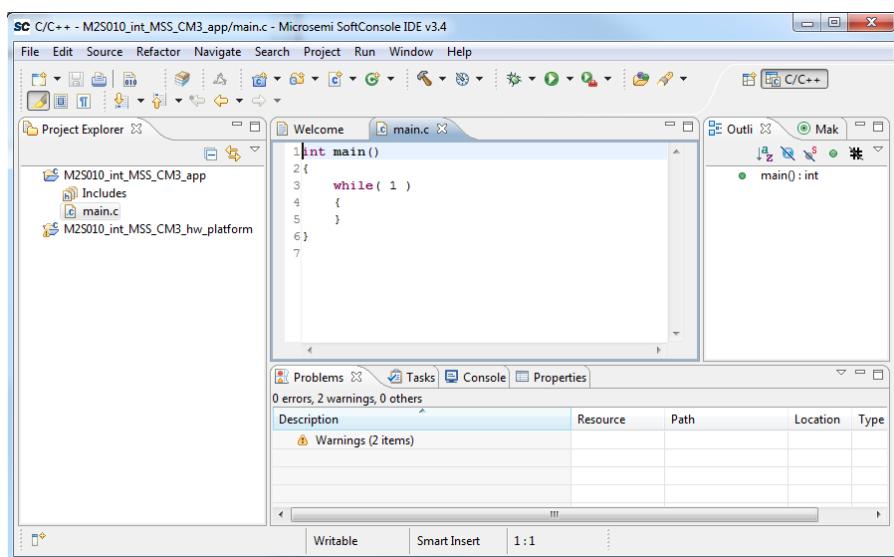


Рис. 38.

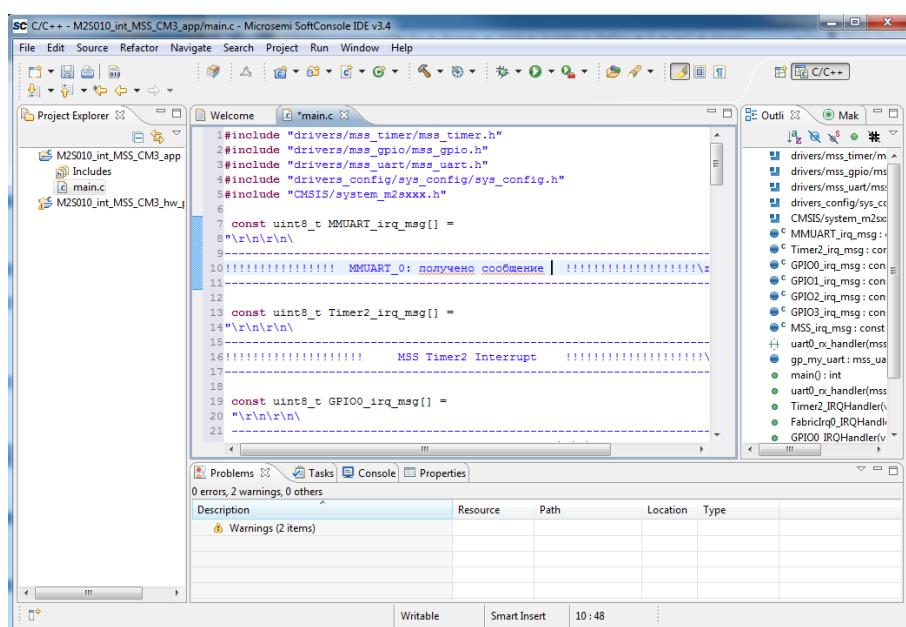


Рис. 39.

Установим активной конфигурацию Release (рис. 40).

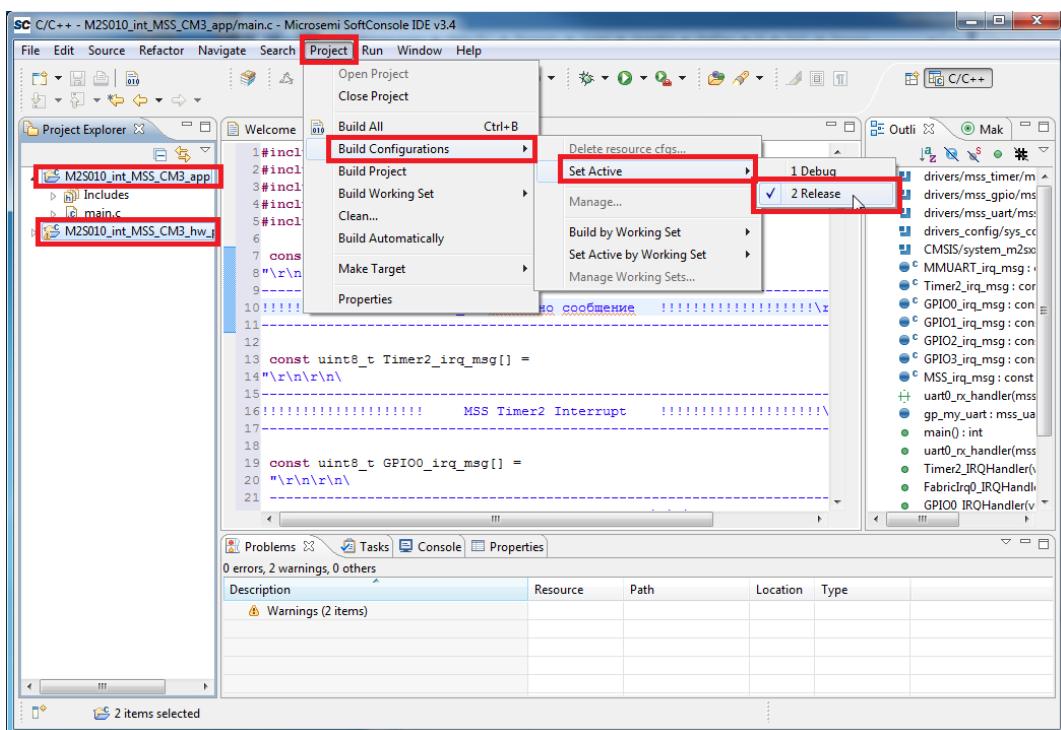


Рис. 40.

Выделив проект M2S010\_int\_MSS\_CM3\_app, отредактируем его свойства (рис. 41). Во вкладке C/C++ Build > Settings перейдем в окно Tool Settings и в настройках GNU C Linker > Miscellaneous (рис. 42) изменим значение параметра Linker flags на

-T\${workspace\_loc:/M2S010\_int\_MSS\_CM3\_hw\_platform/CMSIS/startup\_gcc/production-smartfusion2-execute-in-place.ld}

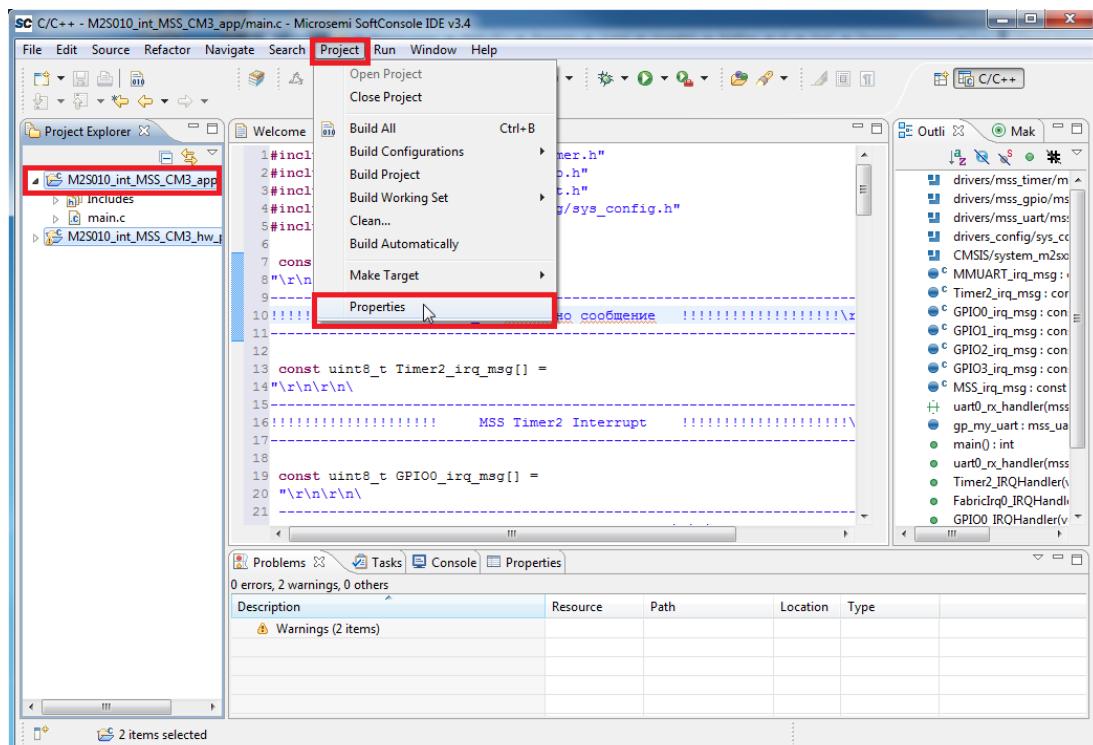


Рис. 41.

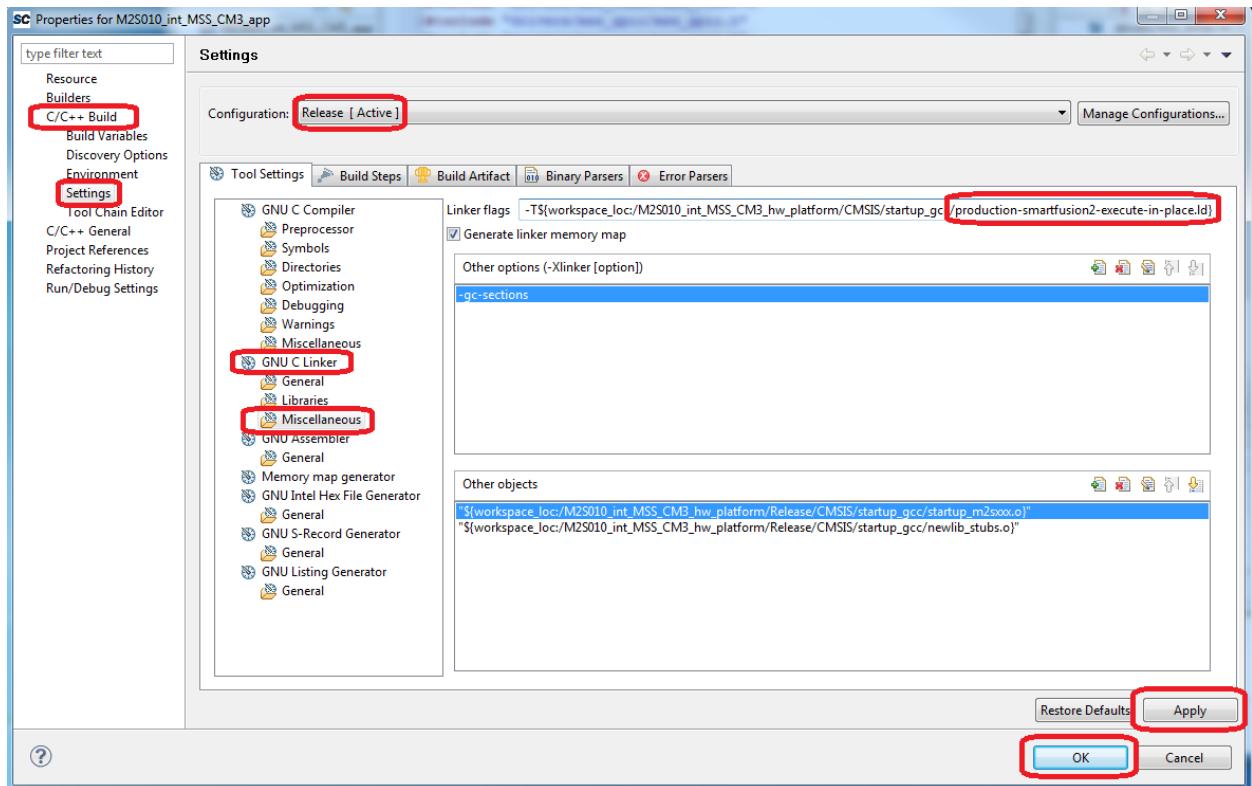


Рис. 42.

Выполним команду основного меню **Project > Clean** и получим файл с исполнимым кодом нашего приложения в паке Release проекта ВПО (рис. 43).

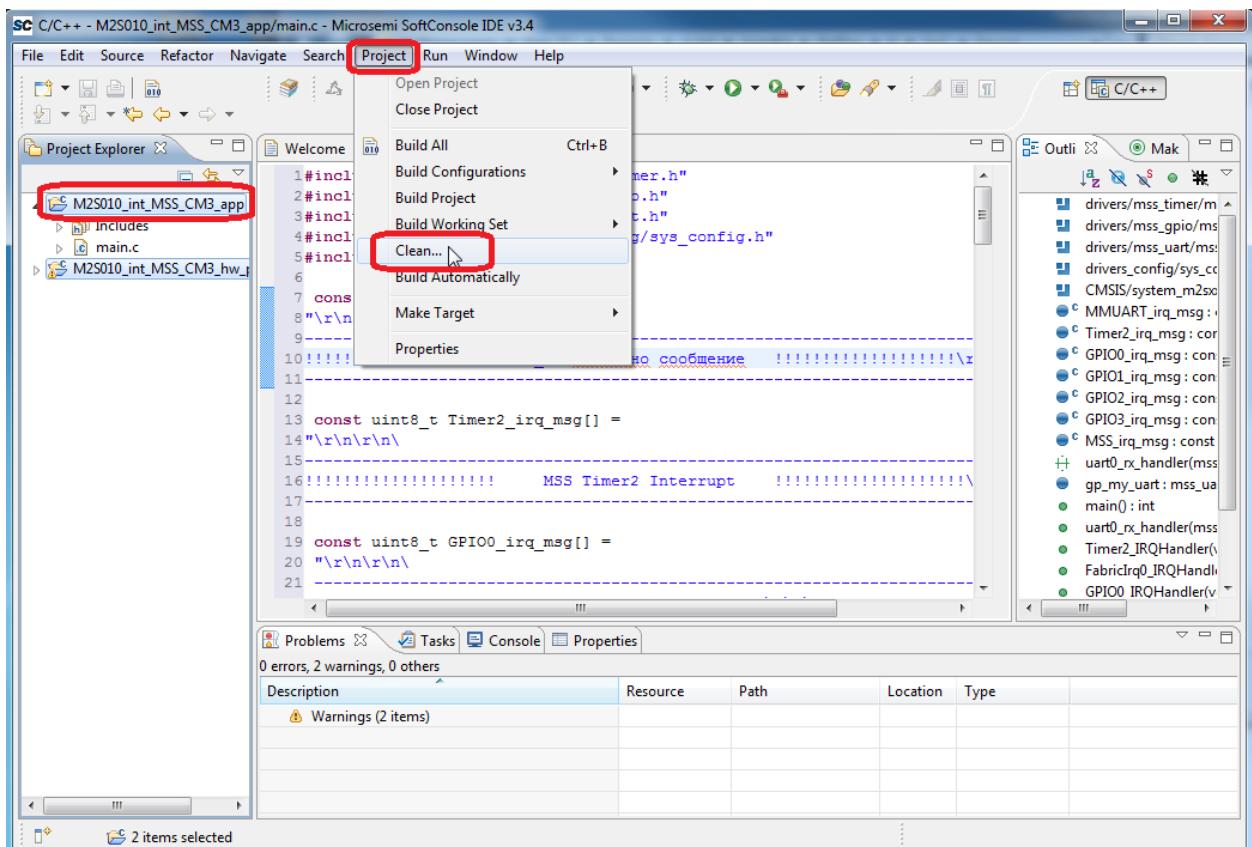


Рис. 43.

В результате выполнения команды в папке проекта ВПО появится каталог Release, а в нем файл образа встраиваемого программного обеспечения процессора ARM Cortex-M3, не содержащий отладочной информации, который может быть использован для создания единого файла прошивки СнК, включающего битстримы FPGA Fabric и ВПО (рис. 44).

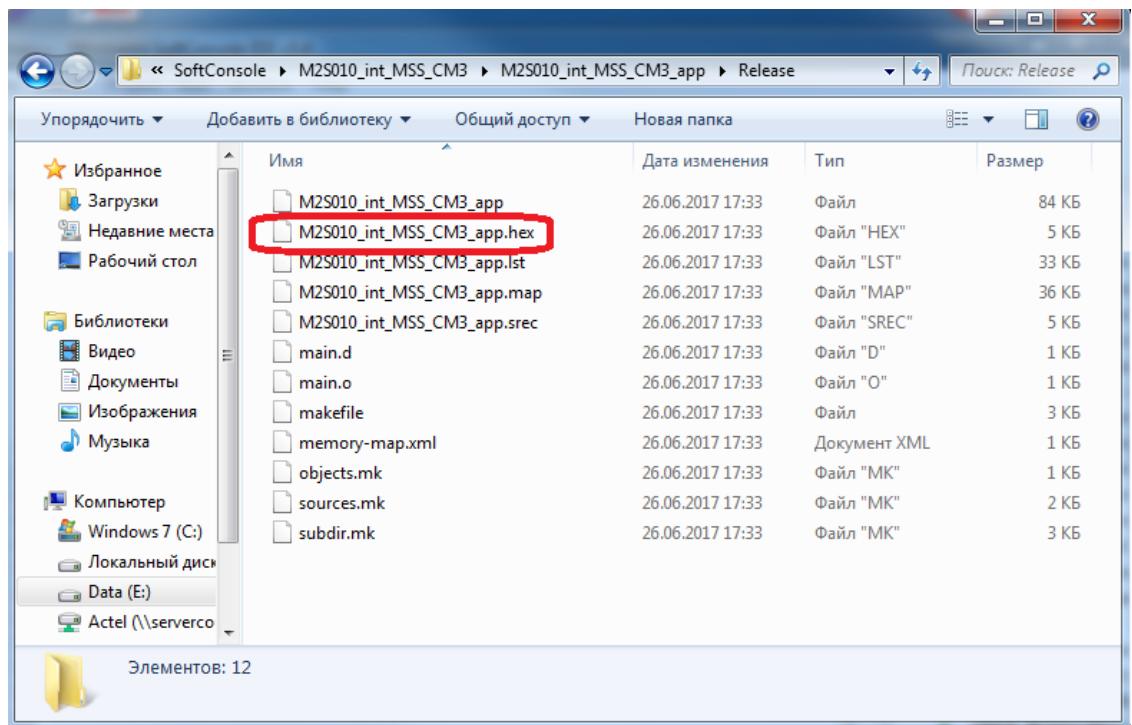


Рис. 44.

## Создание файла конфигурационной последовательности

Теперь создадим файл конфигурационной последовательности, включающий битстрим матрицы FPGA Fabric и исполняемый образ встроенного программного обеспечения процессора ARM Cortex-M3. Для этого в проекте СнК в среде Libero SoC 11.8 в окне настроек микроконтроллерной подсистемы отредактируем свойства блока eNVM, для чего дважды щелкнем на соответствующем элементе графического интерфейса (рис. 45).

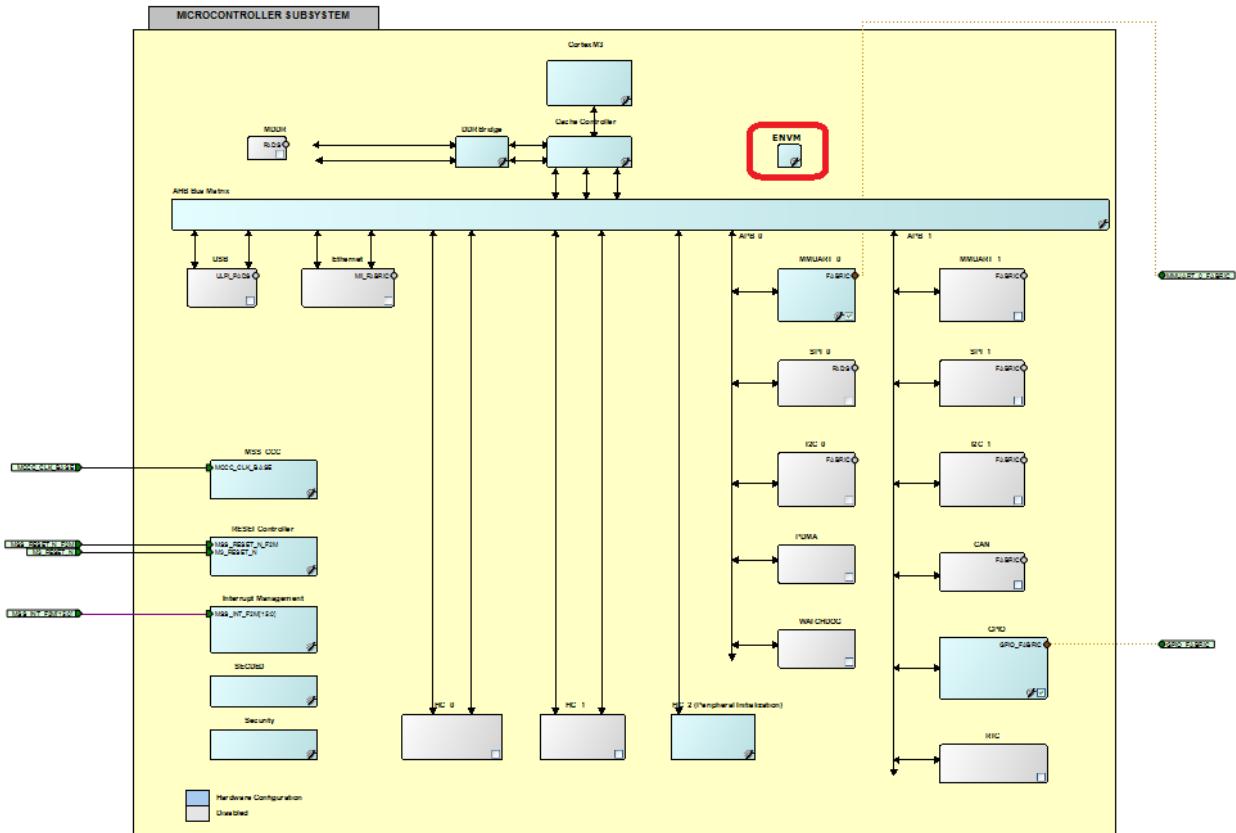


Рис. 45.

В появившемся окне **eNVM Configurator** дважды щелкнем на пункте **Data Storage** и в следующем появившемся окне **Add Data Storage Client** укажем условное название нашего приложения, а также пути к hex-файлу образа прошивки ВПО M2S010\_int\_app.hex (рис. 46).

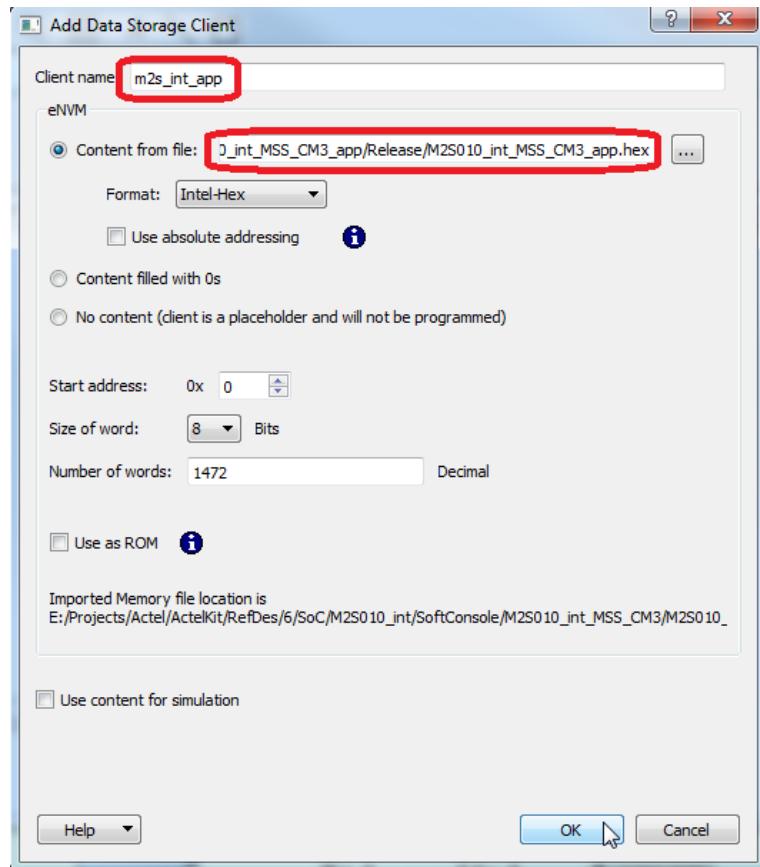


Рис. 46.

После чего окно **eNVM Configurator** примет вид, представленный на рис. 47. В окне отобразится строка с основными характеристиками загруженного файла.

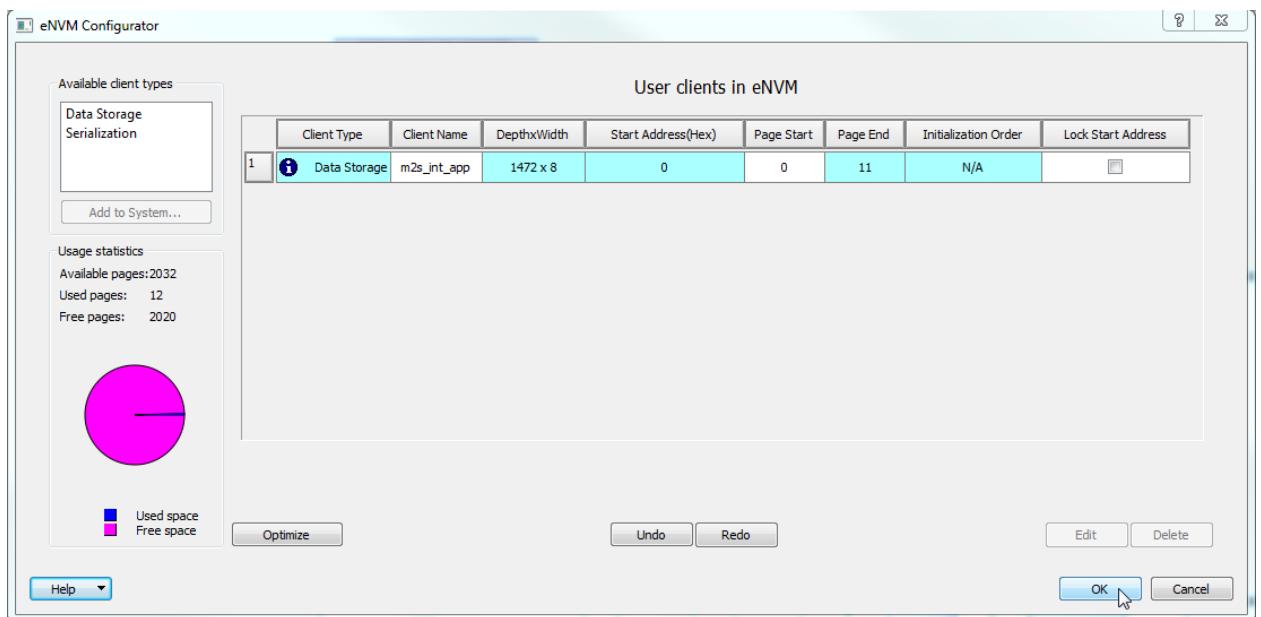


Рис. 47.

Сохраним изменения, выполним процедуру генерации компонента (рис. 48).

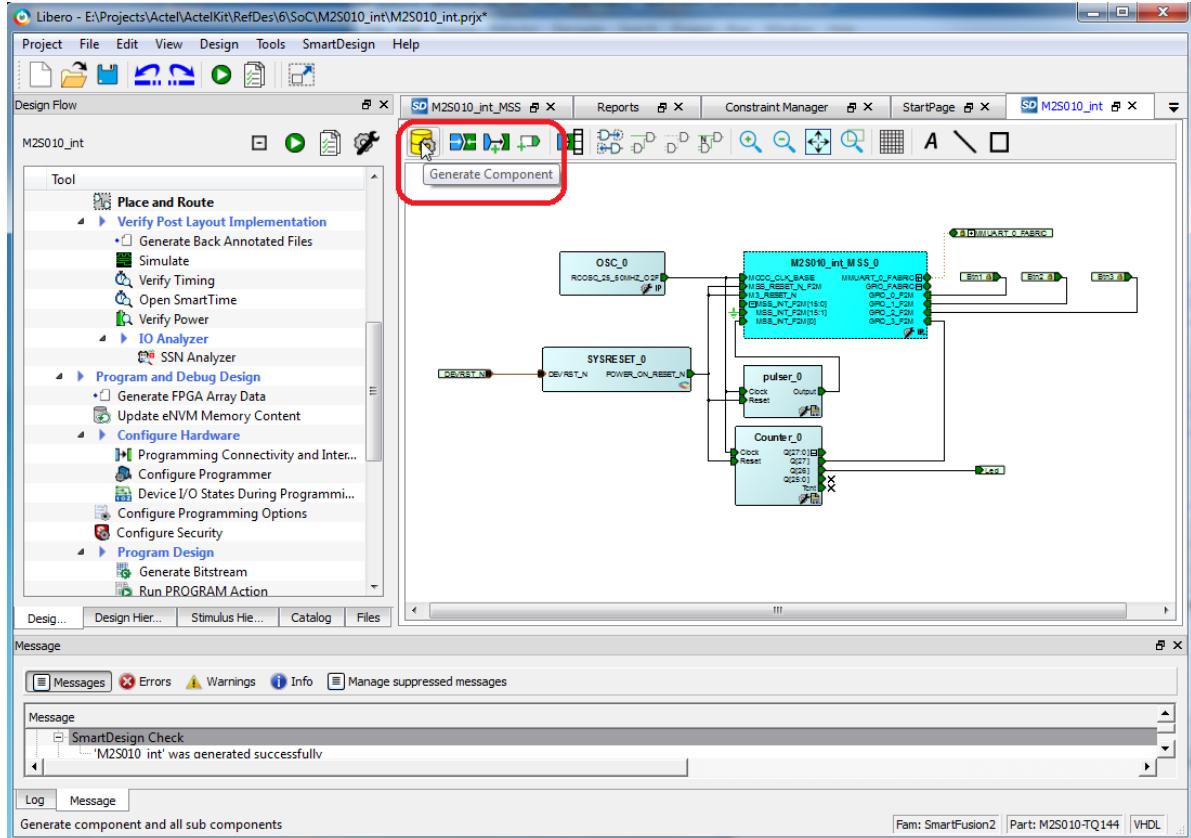


Рис. 48.

Подключим программатор к отладочному набору и персональному компьютеру, запустим процесс генерации файла прошивки и программирования (рис. 49).

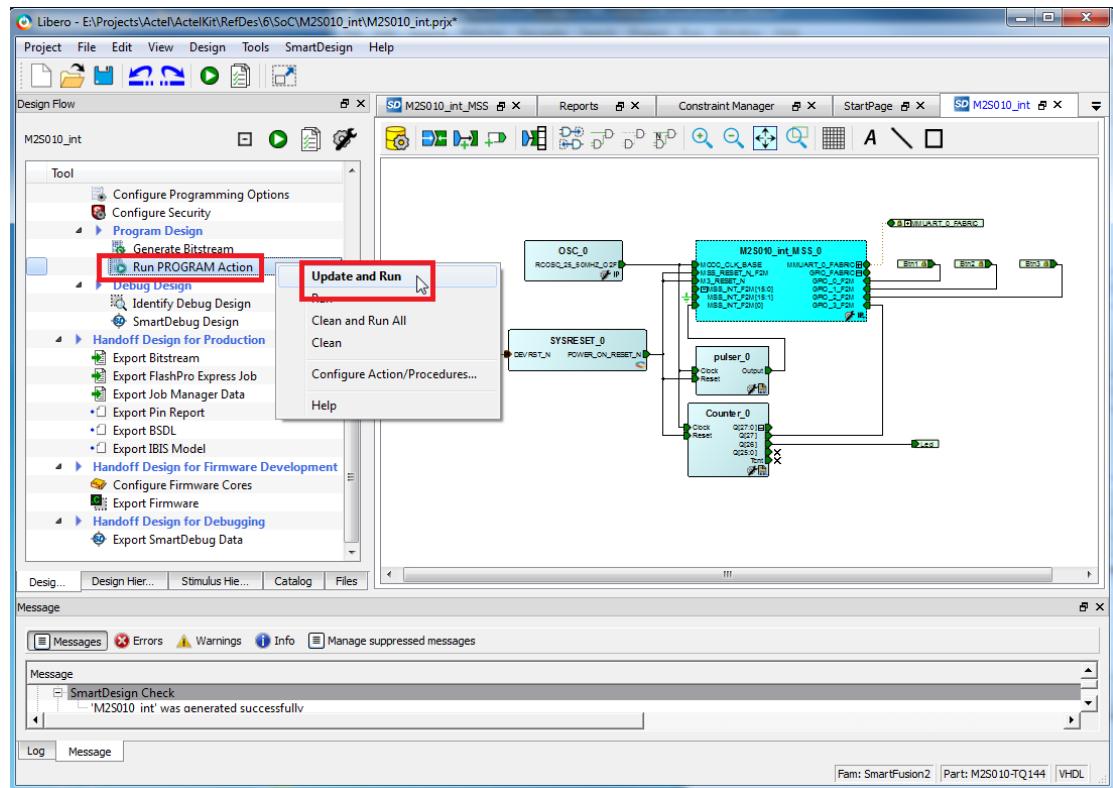


Рис. 49.

На плате отладочного набора SF2-Junior-KIT располагаются два разъема ввода-вывода общего назначения (GPIO) имеющие позиционные обозначения X9, X10 для подключения сигналов внешних устройств к контактам M2S010-TQ144 отладочного набора. В данном проекте контакты микросхемы M2S010-TQ144 назначены таким образом, чтобы вывести сигналы интерфейса UART (UART\_Rx, UART\_Tx) на разъем X10.

В состав набора SF2-Junior-KIT входят приемопередатчики USB – Bluetooth dongle и модуль UART- Bluetooth HC-06. Для связи отладочного набора с персональным компьютером (ПК) необходимо в USB разъем ПК включить USB – Bluetooth dongle, а к разъему X10 отладочного набора в соответствии с таблицей 2 подключить модуль HC-06.

Таблица 2. Подключение модуля HC-06 к разъему X10 отладочного набора SF2-Junior-KIT

№ контакта разъема X10 SF2-Junior-KIT	Номер контакта M2S010-TQ144	Название цепи проекта СнК / принципиальной схемы	Название цепи модуля HC-06
5	9	MMUART_0_RXD_F2M	RXD
6	10	MMUART_0_TXD_M2F	TXD
25	-	+3.3V	VCC
29	-	GND	GND

Выполняем подключения, описанные в таблице 2. Прошиваем СнК конфигурационной последовательностью нашего проекта.

Для отображения результатов измерений на экране ПК воспользуемся программой – терминалом TeraTerm.

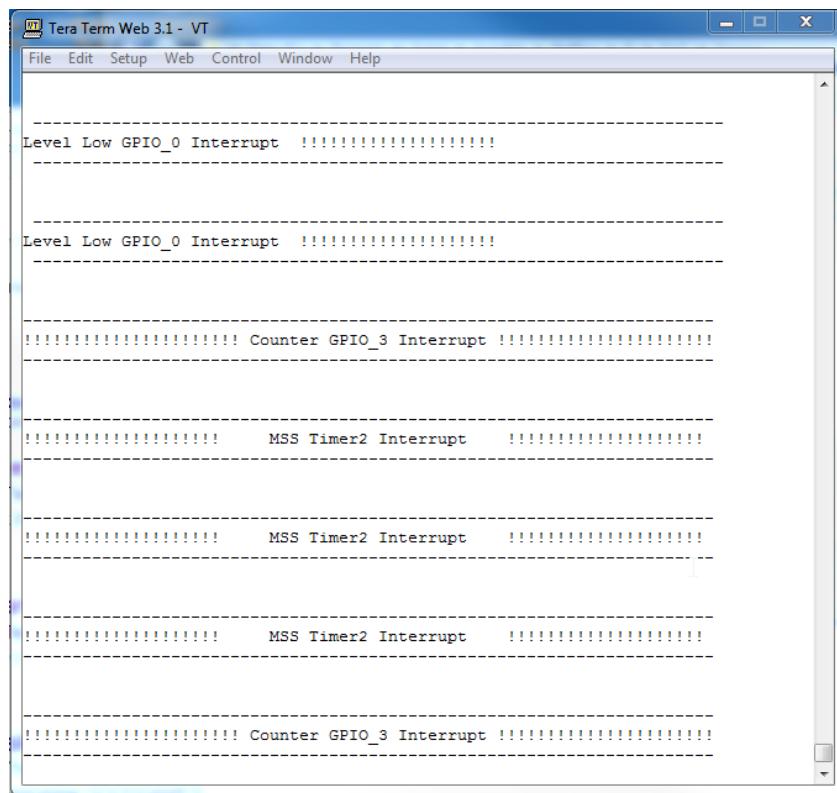


Рис. 50.

Нажимаем кнопки на плате отладочного набора, проверяем появление информационных сообщений обо всех предусмотренных проектом событиях (рис. 50).

Проекты Снк и ВПО, разработка которых описана в данном руководстве, файлы с исходным кодом встроенного программного обеспечения можно скачать по [ссылке](#).

Вопросы по материалу, изложенному в данном руководстве, можно задать сотрудникам службы технической поддержки компании ООО «ПСР Актел» по телефону **+7 (812) 740-60-09**, или по электронной почте **support@actel.ru**.