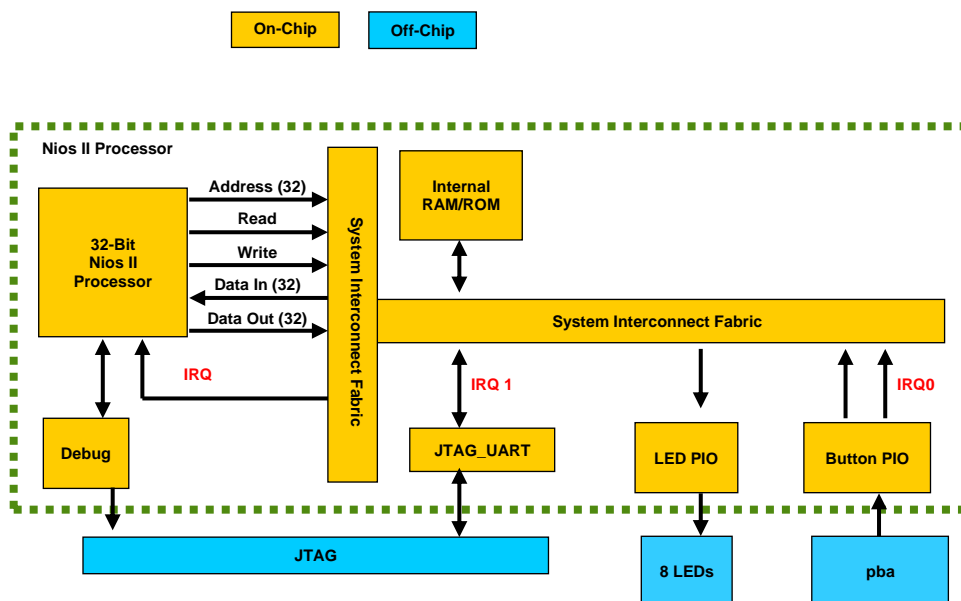


Задание labn_4

Введение:

Цель упражнения – познакомиться с возможностями процессора NIOSII по обработке прерываний.

Структура проекта



Алгоритм работы проекта:

Под управлением процессора NIOSII обеспечивается:

- Работа по прерываниям от нажатия кнопки pba
- Формирование на консоли сообщений о нажатой кнопке
- При каждом нажатии кнопки pba происходит изменение номера включенного светодиода от led1 к led8 на одну позицию (с циклическим переходом от led8 к led1)

Часть 1 – Создание проекта

1. Запустите пакет QuartusII

2. Создайте проект Lab4:

What is the working directory for this project? Рабочая папка (с помощью браузера найдите рабочую папку проекта)	C:\Intel_trn\Q_NIOS\Lab4\
What is the name of this project? Имя проекта	Lab4
What is the name of the top-level design entity for this project? Имя модуля верхнего уровня в иерархии проекта.	Lab4

- в разделе **Available devices** укажите СБИС **EP4CE6E22C8**.

Часть 2 – Создание аппаратной части проекта

1. Скопируйте систему из проекта Lab3_nios.qsys в рабочую папку проекта.
2. Выполните команду **Tools => Qsys**. Будет запущен Qsys.
3. Откройте систему Lab3_nios.qsys и сохраните ее под именем Lab4_nios.qsys
4. Настройте прерываний в модуле buttons:

PIO (Parallel I/O) - buttons

PIO (Parallel I/O)
altera_avalon_pio

Block Diagram

Show signals

clk, reset, s1, external_connection, clock, reset, avalon, conduit, interrupt, altera_avalon_pio

Basic Settings

Width (1-32 bits): 1

Direction: ☐ Bidir ☒ Input ☐ InOut ☐ Output

Output Port Reset Value: 0x0000000000000000

Output Register

☐ Enable individual bit setting/clearing

Edge capture register

☒ Synchronously capture

Edge Type: FALLING

☒ Enable bit-clearing for edge capture register

Interrupt

☒ Generate IRQ

IRQ Type: EDGE

Test bench wiring

☐ Hardwire PIO inputs in test bench

Drive inputs to field.: 0x0000000000000000

5. Подключите выход irq модуля buttons ко входу irq модуля nios2_qsys
6. Измените (щелчок в поле => новое значение) вектор прерывания для компонентов buttons и jtag_uart так, как показано на рисунке

Use	Connections	Name	Description	Export	Clock	Base	End	I...	Tags
<input checked="" type="checkbox"/>		<div>clk</div> <div>clk_in</div> <div>clk_in_reset</div> <div>clk</div> <div>clk_reset</div>	<div>Clock Source</div> <div>Clock Input</div> <div>Reset Input</div> <div>Clock Output</div> <div>Reset Output</div>	<div>clk</div> <div>reset</div> <div>Double-click to</div> <div>Double-click to</div>	<div>exported</div> <div>[clk_in]</div> <div>clk</div>				
<input checked="" type="checkbox"/>		<div>onchip_mem</div> <div>clk1</div> <div>s1</div> <div>reset1</div>	<div>On-Chip Memory (RAM o...</div> <div>Clock Input</div> <div>Avalon Memory Mapped ...</div> <div>Reset Input</div>	<div>Double-click to</div> <div>Double-click to</div> <div>Double-click to</div>	<div>clk</div> <div>[clk1]</div> <div>[clk1]</div>	# 0x4000	0x7fff		
<input checked="" type="checkbox"/>		<div>nios2_qsys</div> <div>clk</div> <div>reset</div> <div>data_master</div> <div>instruction_m...</div> <div>irq</div> <div>debug_reset_r...</div> <div>debug_mem_...</div> <div>custom_instru...</div>	<div>Nios II Processor</div> <div>Clock Input</div> <div>Reset Input</div> <div>Avalon Memory Mapped ...</div> <div>Avalon Memory Mapped ...</div> <div>Interrupt Receiver</div> <div>Reset Output</div> <div>Avalon Memory Mapped ...</div> <div>Custom Instruction Master</div>	<div>Double-click to</div> <div>Double-click to</div> <div>Double-click to</div> <div>Double-click to</div> <div>Double-click to</div> <div>Double-click to</div> <div>Double-click to</div>	<div>clk</div> <div>[clk]</div> <div>[clk]</div> <div>[clk]</div> <div>[clk]</div> <div>[clk]</div> <div>[clk]</div>			IRQ 0	IRQ 31
<input checked="" type="checkbox"/>		<div>led</div> <div>clk</div> <div>reset</div> <div>s1</div> <div>external_conn...</div>	<div>PIO (Parallel I/O)</div> <div>Clock Input</div> <div>Reset Input</div> <div>Avalon Memory Mapped ...</div> <div>Conduit</div>	<div>Double-click to</div> <div>Double-click to</div> <div>Double-click to</div>	<div>clk</div> <div>[clk]</div> <div>[clk]</div>	# 0x9010	0x901f		
<input checked="" type="checkbox"/>		<div>buttons</div> <div>clk</div> <div>reset</div> <div>s1</div> <div>external_conn...</div>	<div>PIO (Parallel I/O)</div> <div>Clock Input</div> <div>Reset Input</div> <div>Avalon Memory Mapped ...</div> <div>Conduit</div>	<div>Double-click to</div> <div>Double-click to</div> <div>Double-click to</div>	<div>clk</div> <div>[clk]</div> <div>[clk]</div>	# 0x9000	0x900f		
<input checked="" type="checkbox"/>		<div>irq</div> <div>jtag_uart</div> <div>clk</div> <div>reset</div> <div>avalon_jtag_sl...</div> <div>irq</div>	<div>Interrupt Sender</div> <div>JTAG UART</div> <div>Clock Input</div> <div>Reset Input</div> <div>Avalon Memory Mapped ...</div> <div>Interrupt Sender</div>	<div>Double-click to</div> <div>Double-click to</div> <div>Double-click to</div> <div>Double-click to</div>	<div>[clk]</div> <div>clk</div> <div>[clk]</div> <div>[clk]</div>				
<input checked="" type="checkbox"/>						# 0x9020	0x9027		

- Выполните команду System=>Assign Base Adresse
- Сохраните изменения в системе: выполните команду File=>Save
- Выполните команду Generate=>Generate HDL.
- Задайте опции так, как показано на рисунке

Generation

Synthesis

Synthesis files are used to compile the system in a Quartus Prime project.

Create HDL design files for synthesis: Verilog

☐ Create timing and resource estimates for third-party EDA synthesis tools.

☐ Create block symbol file (.bsf)

Simulation

The simulation model contains generated HDL files for the simulator, and may include simulation-only features.

Simulation scripts for this component will be generated in a vendor-specific sub-directory in the specified output directory.

Follow the guidance in the generated simulation scripts about how to structure your design's simulation scripts and how to use the *ip-setup-simulation* and *ip-make-simscript* command-line utilities to compile all of the files needed for simulating all of the IP in your design.

Create simulation model: None

Output Directory

Path: C:/Intel_trn/Q_NIOS/Lab4/Lab4_nios

- Нажмите кнопку Generate.
- Не должно появиться ошибок генерации. Предупреждение

Warning: **Lab4_nios.jtag_uart**: JTAG UART IP input clock need to be at least double (2x) the operating frequency of JTAG TCK on board

можно проигнорировать.

13. Нажмите кнопку Close, а затем Finish

Часть 3 – Интеграция аппаратной части проекта

1. Создайте в текстовом редакторе файл (имя файла Lab4.sv) верхнего уровня в иерархии проекта (для этого целесообразно использовать файл Lab4_nios_inst.v из папки C:\Intel_trn\Q_NIOS\Lab4\Lab4_nios)
 - а. Пример файла приведен на рисунке

```

1  module Lab4 (
2      input bit clk,          // Clock
3      input bit pbb,          // System reset active low
4      input bit pba,          // button
5      output bit [7:0] led
6  );
7  Lab4_nios u0 (
8      .clk_clk (clk), // clk.clk
9      .reset_reset_n (pbb), // reset.reset_n
10     .led_export (led), // led.export
11     .pba_export (pba) // pba.export
12 );
13 endmodule

```

2. Назначьте файл Lab4.sv файлом верхнего уровня в иерархии проекта.
3. Добавьте к проекту файл Lab4_nios.qip из папки C:\Intel_trn\Q_NIOS\Lab4\Lab4_nios\synthesis
4. Проверка синтаксиса проекта.
 - а. Выполните команду Processing=>Start=>Start Analysis and Elaboration
5. Назначение выводов проекта. Назначьте выводы так, как показано на рисунке ниже

Node Name	Direction	Location	I/O Bank	/REF Group	I/O Standard
in clk	Input	PIN_23	1	B1_NO	3.3-V LVTTTL
out led[7]	Output	PIN_65	4	B4_NO	2.5 V (default)
out led[6]	Output	PIN_66	4	B4_NO	2.5 V (default)
out led[5]	Output	PIN_67	4	B4_NO	2.5 V (default)
out led[4]	Output	PIN_68	4	B4_NO	2.5 V (default)
out led[3]	Output	PIN_69	4	B4_NO	2.5 V (default)
out led[2]	Output	PIN_70	4	B4_NO	2.5 V (default)
out led[1]	Output	PIN_71	4	B4_NO	2.5 V (default)
out led[0]	Output	PIN_72	4	B4_NO	2.5 V (default)
in pba	Input	PIN_64	4	B4_NO	2.5 V (default)
in pbb	Input	PIN_58	4	B4_NO	2.5 V (default)

14. Назначение опции проекта:

Assignment=>Device=>Device and Pin Options=>Unused pin=>As input tri-stated with weak pull-up resistor
15. С помощью Timing Analyzer или в текстовом редакторе создайте файл (**Lab4.sdc**) с требованиями к временным параметрам проекта. Пример файла приведен на рисунке

```
##
## DEVICE "EP4CE6E22C8"
##

#####
# Time Information
#####

set_time_format -unit ns -decimal_places 3

#####
# Create Clock
#####

create_clock -name {clock} -period 40.000 -waveform { 0.000 20.000 } [get_ports {clk}]

#####
# Set Clock Uncertainty
#####

derive_clock_uncertainty

#####
# Set Input Delay
#####

set_input_delay -add_delay -clock [get_clocks {clock}] 10.000 [get_ports {pbb}]
set_input_delay -add_delay -clock [get_clocks {clock}] 10.000 [get_ports {pba}]

set_input_delay -add_delay -clock [get_clocks {clock}] 10.000 [get_ports {altera_reserved_tdi}]
set_input_delay -add_delay -clock [get_clocks {clock}] 10.000 [get_ports {altera_reserved_tms}]

#####
# Set Output Delay
#####

set_output_delay -add_delay -clock [get_clocks {clock}] 10.000 [get_ports {led[0]}]
set_output_delay -add_delay -clock [get_clocks {clock}] 10.000 [get_ports {led[1]}]
set_output_delay -add_delay -clock [get_clocks {clock}] 10.000 [get_ports {led[2]}]
set_output_delay -add_delay -clock [get_clocks {clock}] 10.000 [get_ports {led[3]}]
set_output_delay -add_delay -clock [get_clocks {clock}] 10.000 [get_ports {led[4]}]
set_output_delay -add_delay -clock [get_clocks {clock}] 10.000 [get_ports {led[5]}]
set_output_delay -add_delay -clock [get_clocks {clock}] 10.000 [get_ports {led[6]}]
set_output_delay -add_delay -clock [get_clocks {clock}] 10.000 [get_ports {led[7]}]

set_output_delay -add_delay -clock [get_clocks {clock}] 10.000 [get_ports {altera_reserved_tdo}]
```

16. Подключите файл Lab4.sdc к проекту.

17. Осуществите полную компиляцию проекта: команда **Processing => Start Compilation**.

18. Компиляция должна завершиться без ошибок. Все требования к временным параметрам должны быть выполнены.

Часть 4 – Создание программной части проекта

1. Запустите оболочку для разработки/отладки программ - NIOSII SBT
 - а. Рабочую папку задайте как: C:\Intel_trn\Q_NIOS\Lab4\workspace
2. Создайте проект lab4_sw (lab4_sw и lab4_sw_bsp)
3. В проекте lab4_sw создайте исходный файл lab4_source.c
4. Введите исходный код подпрограммы обработки прерывания и основной программы

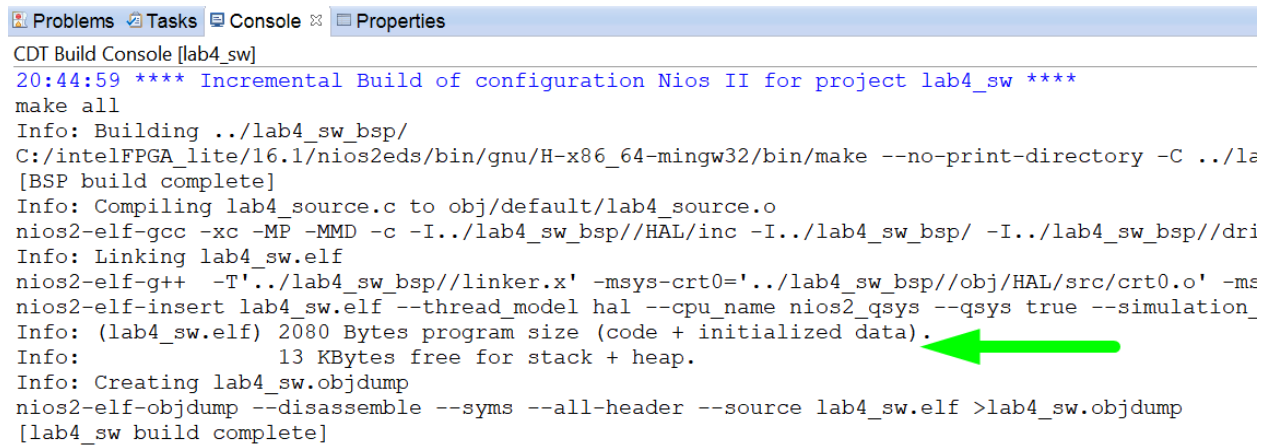
```

lab4_source.c
1  #include "system.h"
2  #include "altera_avalon_pio_regs.h"
3  #include <unistd.h>
4  #include "sys\alt_irq.h"
5  #include "sys\alt_stdio.h"
6  #define DEBOUNCE 100000 // Time in microseconds to wait for switch debounce
7  /*****
8   *   ISR Function Definition:
9   *
10  *****/
11 static void button_isr( void * context )
12 {
13 //   Declare local variable to point to context var
14 //   (Should be declared as volatile)
15 //   Cast "context" to correct type
16   volatile int * button_val_ptr = (volatile int *) context;
17 //   Set the value of the "context" pointer to the value contained in EDGE_CAP
18 //   (use IORD macro defined in "altera_avalon_pio_regs.h" header file)
19   *button_val_ptr = IORD_ALTERA_AVALON_PIO_EDGE_CAP(BUTTONS_BASE);
20 //   Then reset that register to 0
21   IOWR_ALTERA_AVALON_PIO_EDGE_CAP(BUTTONS_BASE, 0x3);
22   usleep (DEBOUNCE);
23 }

24 /*****
25 *   Main Program:
26 *
27  *****/
28 int main()
29 {
30   volatile int buttons; // Declare variable "button"
31   int led = 0x00; // Use to write to led
32   buttons = 0; // Initialize the button's value to 0
33
34   // Initialize the registers in the LED peripheral:
35   IOWR_ALTERA_AVALON_PIO_DATA(LED_BASE, led);
36
37   // Enable button interrupt.
38   IOWR_ALTERA_AVALON_PIO_IRQ_MASK(BUTTONS_BASE, 0x1);
39
40   // Reset the edge capture register.
41   IOWR_ALTERA_AVALON_PIO_EDGE_CAP(BUTTONS_BASE, 0x1);
42
43   // Register the "button_isr" routine above to the BUTTONS_IRQ:
44   alt_ic_isr_register(BUTTONS_IRQ_INTERRUPT_CONTROLLER_ID, BUTTONS_IRQ, button_isr, (void*)&buttons, 0x0 );
45
46   alt_printf(" Процессор Nios II запущен!\n ");
47   alt_printf("Нажмите кнопку pba \n \n ");
48
49   while (1)
50   {
51     if (buttons != 0) // if button pressed
52     {
53       alt_printf("Нажата кнопка pba \n ");
54       if (led == 0x80 || led == 0x00 )
55         led = 0x01;
56       else
57         led = led << 1;
58       buttons = 0;
59       // Write new value to led
60       IOWR_ALTERA_AVALON_PIO_DATA(LED_BASE, ~led);
61     }
62   }
63   return 0;
64 }

```

19. Настройте lab4_sw_bsp для минимизации размера порождаемого исполняемого файла.
20. Запустите порождение исполняемого кода для проекта lab4_sw.
21. Проверьте результат компиляции:



```
CDT Build Console [lab4_sw]
20:44:59 **** Incremental Build of configuration Nios II for project lab4_sw ****
make all
Info: Building ../lab4_sw_bsp/
C:/intelFPGA_lite/16.1/nios2eds/bin/gnu/H-x86_64-mingw32/bin/make --no-print-directory -C ../lab4_sw_bsp/
[BSP build complete]
Info: Compiling lab4_source.c to obj/default/lab4_source.o
nios2-elf-gcc -xc -MP -MMD -c -I../lab4_sw_bsp//HAL/inc -I../lab4_sw_bsp/ -I../lab4_sw_bsp//drivers
Info: Linking lab4_sw.elf
nios2-elf-g++ -T'../lab4_sw_bsp//linker.x' -msys-crt0='../lab4_sw_bsp//obj/HAL/src/crt0.o' -msys-crtlib=lib4_sw.elf
Info: (lab4_sw.elf) 2080 Bytes program size (code + initialized data).
Info: 13 KBytes free for stack + heap.
Info: Creating lab4_sw.objdump
nios2-elf-objdump --disassemble --syms --all-header --source lab4_sw.elf >lab4_sw.objdump
[lab4_sw build complete]
```

Часть 5 – Программирование СБИС и запуск программы на плате

1. Осуществите программирование СБИС ПЛ: на плате загорится зеленый светодиод.
2. В NIOSII SBT выберите папку lab4_sw и запустите загрузку и выполнение программы на плате
3. Проверьте работу программы:
 - а. После загрузки в окне консоли отобразится:



```
lab4_sw Nios II Hardware configuration - cable: USB-Blaster on localhost [USB-0] device ID: 1 instance ID: 0 name: jtaguart_0
Процессор Nios II запущен!
Нажмите кнопку pba
```

- б. При нажатии кнопки pba
 - i. в окне консоли формируется сообщение
 - ii. происходит изменение номера включенного светодиода от led1 к led8 на одну позицию (с циклическим переходом от led8 к led1)

Упражнение 4 завершено.