Układy FPGA

– programowanie bramek w zestawie uruchomieniowym Basys3 firmy Digilent -środowisko Vivado (2016.4)- firmy Xilings , język Verilog

Pierwszy projektowany układ będzie zawierał dwie bramki logiczne, AND i OR, których wejścia będą podłączone do przełączników na płytce Basys3 (oznaczone od SWO do SW15 - wykorzystamy SWO - SW3), a stan wyjść będzie wskazywany przez diody LED (oznaczone od LDO do LD15 - wykorzystamy LD0, LD1).

1.Tworzenie projektu

- Uruchamiamy środowisko Vivado 2016.4 (na dolnym pasku)
- Klikamy na ikonę tworzenia nowego projektu ("Create New Project"). Otwiera sie okno "Create a New Vivado Project" → Next
- W oknie Project name nazwij projekt simple_gates i wybierz jego lokalizację jako np. Project location:
 C:/Users/student. Pole Create project subdirectory pozostaw zaznaczone. → Next
- W oknie **Project type** wybierz opcję "**RTL project**" (Register Transfer Level) z zaznaczeniem "**Do not specify** resource at this time" (kody źródłowe zostaną dodane w późniejszym etapie). → Next

W oknie **Default part** należy wybrać: Product category: General Purpose Family: Artix-7 Sub-family: Artix-7 Package: cpg236 Speed grade: -1 Temperature grade: All Remaining Z nazostałych do wyboru zaznaczyć yc

Z pozostałych do wyboru zaznaczyć xc7a35tcpg236-1. Model ten posiada 20800 tablic LUT. \rightarrow Next

- Następnie pojawia się okno: "New project Summary" a w nim opisane są parametry charakterystyczne projektu →Finish. Przez parę sekund tworzy się projekt (Create project) i otwiera się kolejne okno czyli projekt: simple_gates.
- Po utworzeniu projektu, z lewej strony okna powinien znajdować się panel Flow Navigator. Jeżeli jest on niewidoczny, wciśnij kombinację Ctrl+Q.

simple_gates - [C/Users/root/simp	ple_gates/simple_gates.xpr) - Vivado 2015.4				
Ele Edit Flow Tools Window I	Layout Wew Help	Q ₂ - Search commande			
👌 in er 🎼 🎼 🗙 👂 🕨	🐮 🚳 💥 🔽 🧑 🔚 Default Layout 🔹 🖉 🐐 🏌 🖏	Ro			
Row Navigator 🔍	Project Manager - single_gates				
人工は	Sources _ D 2 ×	E Project Summary X			
	역 🎞 🖨 📾 🚯 🐘 🔁	received and the second			
Project Pataloger Project Settings Add Sources PLanguage Templetes P Catalog P Integrator Create Block Design	(© traje Sauces → Contrates → Contrates - Contrates	Regect ranes: anyle_gatos Regect ranes: C_Alersy to the sets Regect ranes: C_Alersy to the sets Regect ranes: A relation of the sets Regect ranes: Relation of the sets Regect ranes: Relation Regect ranes: Relation Regect ranes: Relation			
🗊 Open Block Design		Synthesis			
Generate Book Design Simulation Go Simulation Settings Go Run Simulation		Status: Notsmited Status: Notsmited Messages: No errors or warrings Messages: No errors or warrings Part: xr7x351pp228-1 Parts Xr7x351pp228-1 Status: Status: Status: Not smarted			
	Hierarchy Ubraries Comple Order	Incremental complex laces			
 RTL Analysis 	A Sources V Templates				
C Elaboration Settings	Properties 🗆 🕹 ×	DRC Violations 2 Timing			
Open Elaborated Design	$\leftarrow \rightarrow \underbrace{\circ}_{2} \models$				
 Synthesis 		STATISTICATION IN Sec. Proc. ISSN 10. Sec. Sec. ISSN 10.			
🚯 Synthesis Settings		Utilization 2 Power			
 Run Synthesis Im Open Synthesized Design 		Bun Senthese to see utilization results			
Implementation Implementation Settings Implementation Run Implementation Implementation Implementation Trogram and Debug Bitstream Settings	Select an object to see properties				
Cenerate Bitstream	Design Runs	- 0 *			
Open Hardware Manager	Q Name Constraints Status VINS	THS WHS THS TRUE Build Builder LIT FF REAM LEAM DSP Start Flavord Stratery Part Description			
	Image: Second	Naudo Synthesia Defaulti (Naudo Synthesia 2013) xr313/3109228-1 Naudo Synthesia Defaulti Naudo Synthesia Defaulti (Naudo Synthesia 2014) xr323539228-1 Naudo Synthesia Defaulti			
	Tid Console @ Messages 54 Log > Reports Design Runs	1			
3 A 💫 i	🦰 💿 🔚 🕭 🍓 💻	🖏 🕅 🥹 🎐 🧕 🏂 straylic.gater.jtc. 🥪 SXW(JK (0))			

1.Dodawanie plików do projektu

W panelu Flow Navigator będziemy rozwijali zakładkę Project Manager. Utworzymy model bramki AND o nazwie "and_gate", która posiada wyjście "y" i dwa porty wejściowe "a" i "b";

(Project Manager -> Add sources -> Add or create design sources \rightarrow Next \rightarrow Create File...)

- Pojawia sie małe okno: Create Source File

Create a new	v source file and add it to your project.
<u>Fi</u> le type: File name:	😨 Verilog 👻
Fil <u>e</u> location:	Si <local project="" to=""> ▼</local>

- File type: Verilog , File name: and_gate, File location: \rightarrow Choose location: Local to Project \rightarrow OK
- Pojawia się nowe okno z opisem i lokalizacją projektu →Finish
- Następne małe okno: Define Module . Sprawdzamy Module name: and_gate i definiujemy porty (I/OPort Definitions). Port Name/Direction kolejno: a input, b input, y output.(pole Bus zostaje niezaznaczone) → OK



W oknie Project Manager – Sources pojawia się folder : Design Sources - plik (ve): and_gate.v ,(na dole okna powinna być aktywna zakładka "Hierarchy".

Otwieramy go klikając (prawy przycisk myszy) →Open File. Pojawia się zawartość pliku (opis modułu): są w nim tylko oznaczenia wejść i wyjść - bez połączeń i funkcji

```
`timescale 1ns / 1ps
module and_gate
    (
        input a,
        input b,
        output y
    );
endmodule
```

Uzupełnij ten kod kodu, aby wyglądał jak poniżej: (trzeba dokonać połączeń i przypisać funkcje) `timescale 1ns / 1ps

```
module and_gate
    (
        input wire a, // połączenie
        input wire b,
        output wire y
        );
        assign y = a & b; //przypisanie ciągłe, & to AND bitowy
endmodule
```

Zapisujemy moduł (Ctrl + S). Po każdym zapisaniu pliku Vivado analizuje kod. Jeżeli są jakieś błędy składni, to informacja o nich zostaje wyświetlona w zakładce "Messages" okna "Messages" (na dole). Wygeneruj jakiś błąd (np. usuń średnik na końcu komendy assign) i sprawdź, jak wygląda taki komunikat błędu. Zwróć również uwagę, że pliki z błędami pojawiają się w oknie "Sources" w folderze "Design sources → Syntax Error Files"

- Utwórz analogicznie moduł bramki OR o nazwie "or_gate". Pamiętaj o symbolu OR bitowy (str 12 Verilog Quick Reference). Po zapisaniu modułu bramki OR w folderze Design Sources sa dwa pliki: and_gate nadrzedny (oznaczony trzema punktami jako hierarchia) oraz or_gate podrzędny. Prawidłowa hierarchia ułozy się dopiero po skonfigurowaniu "artix7".
- Utwórz model całości układu o nazwie "artix7" według schematu poniżej:



- Jest to moduł hierarchiczny: przypisanie a,b odpowiednio do sw oraz y do led.
- .a(sw[1]) → . kropka oznacza strukturę hierarchiczną, np: a oznacza wejscie/wyjście (lub port) niższego poziomu , zawartość nawiasu (sw[1]) oznacza wejscie/wyjście (lub port) wyższego poziomu , np. sw[1] oznacza określony przełącznik, led[0] oznacza diodę.
- Utwórz model całości układu o nazwie "artix7" :

```
`timescale 1ns / 1ps
module artix7
  (
        input wire [3:0] sw,
        output wire [1:0] led // uzupełnij analogicznie jak dla input wire ...
  );
  or_gate u_or_gate
  (
        .a(sw[1]),
        .b(sw[0]),
        .y(led[0])
  );
  and_gate u_and_gate // uzupełnij analogicznie jak dla or_gate u_or_gate_
endmodule
```

Zapisujemy moduł. Teraz ustawia się prawidłowa hierarchia w Design Sources: artix7 – moduł nadrzędny i dwa moduły podrzędne (and_gate i or_gate).

Po opracowaniu modelu obejrzyj schemat układu. Jeżeli są jakieś błędy podczas łączenia projektu (elaboracji), to należy je usunąć. W zakładce Sources w folderze Design Sources są aktualne pliki zapisane hierarchicznie w postaci:

Sources	- D & X
2 🖾 🛱 🖄 📸 📗 🛃	
Design Sources (3)	
and_gate (and_gate.v)	
······································	
Constraints	
Simulation Sources (3)	
Ularrandez Liberation Controlle Order	
nierarcity Libraries Complie Order	

- **SYMULACJA:** W celu przeprowadzenia symulacji potrzebujemy jeszcze minimum jednego modułu testowego, który zapewni nam generację przebiegów testowych (w folderze Simulation Sources).
- Utwórz moduł File name: "test_simple_gates". Project Manager →Add Sources→Add or create simulation sources→Next→Create File



Okno Define Module:

Port Name:Direction:Bus:MSB:LSB:t_swinputv30t_ledoutputv10

Moduł ma zapewnić przetestowanie bramek dla wszystkich możliwych konfiguracji wejść. Fragment kodu znajduje się poniżej.

W oknie Sources rozwijamy folder: sim_1 i otwieramy plik: test_simple_gates. Wpisujemy poleceniajak poniżej. Miejsca oznaczone należy uzupełnić kodem (do przetestowania pozostałych konfiguracji). Użyj pętli for (lub repeat). Składnię for utwórz według "Verilog Quick Reference" - punkt 7.3 str. 15 (bez memory [i]=0).

```
`timescale 1ns / 1ps
module test_simple_gates();
      reg [3:0] t sw; // patrz Quick Ref. str 2, reg to 4-bitowy vector register
      wire [1:0] t_led;
      artix7 DUT // DUT - Device Under Test
      (
          .sw (t sw), // struktura hierarchiczna
          .led(t led)
      );
      ...... // deklaracja zmiennej typu integer do iteracji (dla pętli for)
      initial
      begin
            $display("%Ot: Starting TPG.", $time);
            // set the inputs
            #10 t sw = 4'b0000;
            for ..... // petla for patrz Quick Ref. str 15, p. 7.3
          begin
            #10
             t_sw=i; //zmienną przypisujemy do t_sw
             #1
            if(t_led == { ....., , ........ }) // wpisać kolejno funkcję AND , OR dla t_sw
      //według rysunku modułu hierarchicznego
            $display("%Ot Test PASSED for pattern %b", $time, t sw);
            else
             $display("%0t Test FAILED for pattern %b - result is %b", $time, t sw, t led);
          end
            #10 $display("%Ot: Finished TPG.", $time);
             $stop;
     end
```

endmodule

Zapisujemy, rozwijamy foldery. Po załadowaniu wszystkich źródeł zakładka "Sources" powinna wyglądać tak:



Jeżeli plik **test_simple_gates** znajduje się w grupie "Design Sources" zamiast w "Simulation Sources", należy kliknąć (prawy przycisk myszy) i wybrać "Move to Simulation Sources".

- SYMULACJA : Uruchomienie symulacji
 - Aby skonfigurować parametry symulacji, w panelu Flow Navigator wybierz zakładkę Simulation -> Simulation settings.
 - Wybierz zakładkę Simulation/Simulation(w środku okna). W polu xsim.simulation.runtime dopasuj odpowiednio czas trwania pojedynczej symulacji. Pamiętaj, że nawet jeżeli ustawisz dłuższy czas trwania symulacji, to zostanie ona zatrzymana komendą \$stop. Jeżeli ustawisz zbyt krótki, zawsze możesz po zatrzymaniu kontynuować. Pole wyboru czasu trwania pozostawić 1000 ns.→OK

Simulat	ion				
<u>T</u> arget sin	nulator:	Vivado Simulator	r		
Simulator	language:	Mixed			
Simulation	set:	🛅 sim_1			
Simulation	top module name:	test_simple_gate	es		8
Clean	up simulation files				
Cor	npilation Elaborati	ion Simulation	Netlist Advanced	1 I	
xsim.s	imulate.runtime*		1000ns		
xsim.s	imulate.log_all_sign	als			
xsim.s	imulate.wdb				
xsim.s	imulate.saif_scope				
xsim.s	imulate.saif				
xsim.s	imulate.saif_all_sigr	nals		(1999)	
xsim.s	imulate.xsim.more_	options			
Select an	option above to see	e a description of i	t		

• Wybierz zakładkę General. W polu Loop count ustawić "jeden" ilość powtórzeń symulacji przy pojedynczym jej wywołaniu (UWAGA: przecinek nie jest separatorem dziesiętnym!) : →OK

General			
Name:	sinple_g	ates	
Project device:	🏶 xc7a	35tcpg236-1 (active)	
∐arget language:	Verilog		
Default library:	xl_defa	dăb	
Top module name:	artx7		
Language Options			
Verilog optio	15:	verilog_version=Verilog 2001	
Generics/Par	ameters:		
Loop count:			

- W panelu Flow Navigator wybierz zakładkę Simulation → Run Simulation →Run Behavioral Simulation (Przeprowadź symulację behawioralną). Przebiegi symulacji powinny wyświetlić się na grafie zbliżonym do:
- Najpierw należy otworzyć na pasku górnym zakładkę "Untitled1" a potem rozwinąć t_sw oraz t_led , a następnie na pasku pionowym kliknąć ikonkę "Zoom fit"

Objects		- 🗆 🖻 ×	🔞 and_gate.v 🗙 🔞 or_gat	te.v 🗙 🔞 artix7.v 🗙 🔞	est_simple_gates.v 🗙 🖀 Untitled 1 🗙 🗆 🗠 🗠
< 🎛 🗃 👪 1	6 6		20		196.000 ns
Name	Value	Data Typ ^	Pame Name	Value	10 ns 50 ns 100 ns 150 ns
🕀 🥳 t_sw[3:0]	f	Array			
🕀 🍓 t_led[1:0]	3	Array	E Sw[3:0]	Ť	
🗄 裓 i[31:0]	16	Array	Q- 16 B	1	
			🔯 🐻 [2]	1	
			Land 14 [1]	1	
			16 [0]	1	
			K = - K t_led[1:0]	3	
			N [1]	1	
			12 14 [0]	1	
			💁 🖬 📲 i[31:0]	00000010	X. 00 \ 00.
			4		

W oknie po lewej zamiast "Sources" rozwija się okno "Scope".

• Klikając w zakładkę Log znajdującą się w dole okna programu można podglądać zapisy logów:

Log Properties	_ 🗆 🖻 ×
II No Unisim elements were transformed.	*
INFO: [Common 17-83] Releasing license: Synthesis	
12 Infos, 0 Warnings, 0 Critical Warnings and 0 Errors encountered.	
synth_design completed successfully	
synth_design: Time (s): cpu = 00:00:08 ; elapsed = 00:00:08 . Memory (MB): peak = 487.129 ; gain = 297.363	
<pre># write_checkpoint -noxdef andgate.dcp</pre>	
<pre># catch { report_utilization -file andgate_utilization_synth.rpt -pb andgate_utilization_synth.pb }</pre>	
report_utilization: Time (s): cpu = 00:00:00 ; elapsed = 00:00:00.040 . Memory (MB): peak = 487.129 ; gain = 0.000	
TURON ICommon 17_2061 Eviting Viusdo at Mad Oct 14 14-50-26 2015	· · ·
Conducto Indexestina Conducto	
Synthesis and and a single strategy and a si	
📓 Td Console 🗋 🗭 Messages 🔽 🔍 Log 🕼 Reports 🕽 Design Runs	

Zakładka Log

- Implementacja w układzie FPGA
- SYNTEZA: Uruchom syntezę (-> Run Synthesis). Zwróć uwagę na pasek statusu w prawym górnym rogu ekranu. Synteza przekształca opis w języku HDL na schemat w postaci bramek . Może pojawić sie komunikat dotyczacy blokady, należy wybrać Allow Access.



- Należy śledzić w prawym górnym rogu ekranu przebieg syntezy i czekać na komunikat "Synthesis Succesfully Completed". Po zakończeniu, na razie, nie wybieramy "Run Implementation" tylko "View Reports"→OK
- Sprawdź, ile procent dostępnych tablic LUT, buforów wejściowych IBUF i buforów wyjściowych OBUF wykorzystał na układ ("Reports → Utilization Raport).

• Zauważ, że w menu Run Synthesis pojawiły się dwie nowe opcje (nie musisz ich teraz uruchamiać). Aby przeprowadzić implementację sprzętową, musimy wprowadzić ograniczenie projektowe (ang. constraints), które powiedzą Vivado, które wyprowadzenia układu FPGA są używane i jak są skonfigurowane. W tym celu:

dodaj do projektu plik ograniczeń XDC: Basys3 Master.xdc (który otrzymacie od prowadzącego).
 Następnie należy zapisać poprzez: Add Sources → Add or Create Constraints →Add Files (z projektu: simple_gates). Odkomentuj potrzebne linijki w pliku .Należy odkomentować wszystkie używane switche(sw 0-3) i ledy(led0-1) po dwa wiersze na każdy switch i led.
 Sprawdzić czy plik xdc jest w folderze "Constraints".

- IMPLEMENTACJA: Uruchom implementację ("→ Run Implementation"). Obserwuj log. Po implementacji zauważ, że pojawiły się nowe zestawy raportów "Place Design" i "Route Design". W większości z uwagi na prostotę projektu będą one większości puste.
 - Znajdź w raporcie informację o procencie wykorzystanych zasobów IO (Place Design->Utilization Raport, Bonded IOB).
 - Otwórz topografię układu (→ Implemented Design), znajdź jedną z tablic LUT (look-up table), która został wykorzystana do budowy bramki AND lub OR, podświetl połączenia wejściowe i wyjściowe .
 - Zaznacz na topografii slice i otwórz jego schemat (prawy przycisk myszy ->Schematic lub F4). Powinien on zawierać 4 bufory wejściowe, dwa elementy tablicy LUT i dwa bufory wyjściowe.
 - W "Messages" mogą się pojawić teraz ostrzeżenia, które ignorujemy:

[Timing 38-313] There are no user specified timing constraints. Timing constraints are needed for proper timing analysis.

[Chipscope 16-3] Cannot debug net 'led[0]'; it is not accessible from the fabric routing.(12 more like this)

- **PROGRAMOWANIE I DEBUGOWANIE:** Należy zaznaczyć "bin file" w Bitstream Settings. Wygeneruj bitstream (plik zawierający dane do programowania FPGA) → Generate Bitstream.
- Upewnij się, że przełączniki na układzie Basys3 są w odpowiednim ustawieniu (obie niebieskie zworki "bliżej środka" płytki):
 - -JP1 (Mode) → USB -JP2 (Power) → USB
- Podłącz układ Basys3 przy pomocy kabla USB do komputera i włącz go przełącznikiem "Power Switch".
 UWAGA: przed wsunięciem wtyku kabla do gniazda płytki Basys3 UPEWNIĆ SIĘ, że wtyk microUSB ("typ smartfonowy") jest prawidłowo ułożony wsuwamy do gniazdka "ząbkami do góry".
- Skonfiguruj podłączenie do programowania płytki Basys3:

 -Program and Debug -> Hardware Manager -> Open Target -> Open New Target ...
 -Wybierz: local server, zegar programowania domyślny 15MHz
- Program device -> xc7a35t_0

-Bitstream File powinien być domyślnie ustawiony poprawnie (sprawdzić w okienku) -Dostaniemy ostrzeżenie, które w tym przypadku możemy zignorować:

WARNING: [Labtools 27-3123] The debug hub core was not detected at User Scan Chain 1 or 3. (Okno Messages bez Errors).

• Sprawdź, czy układ działa poprawnie. Jeżeli tak – gratulacje!