

Opracowanie uproszczonej implementacji generatora sygnałów GNSS w strukturze układu programowanego, umożliwienie przeprowadzenia testów pętli śledzącej odbiornika działającej w trybie generowania próbek w przez układ FPGA z maksymalną częstotliwością próbkowania.

- **Wstęp**

Zakres pracy obejmował kontynuację pracy nad blokami umożliwiającymi wykonanie pomiarów stanu pętli odbiornika GNSS, utrzymanie oraz poprawianie istniejącego kodu, weryfikację planowanego pinoutu FPGA dla odbiornika GoGeo, opracowanie dokumentacji na temat możliwości weryfikacji i debugowania przy użyciu części PL platformy ZYNQ. Opracowano wersję filtrów DLL/PLL analogicznie do przedstawionego kodu C, rozpoczęto kodowanie dyskryminatorów DLL/PLL. Opracowano kontroler przerwań oraz ustalono wstępne przydzielenie odpowiednich linii do odpowiednich zgłoszeń.

Dodatkowo została wykonana weryfikacja pinoutu dla nowego PCB gogeo. Utworzono w tym celu nowy projekt bazujący na wersji 0x3_48 z przeniesieniem platformy z płyty ewaluacyjnej Zybo na układ xc7z020clg484-1. Zostały zmodyfikowane odpowiednie komponenty w celu implementacji poprawnego projektu dla tego przypadku.

Weryfikacja została zakończona wygenerowaniem bitstreamu dla zadanego pinoutu, wersji DDR oraz układu FPGA. Całość projektu została umieszczona na git wraz ze skryptem umożliwiającym automatyczne wygenerowanie projektu Vivado pod ten przypadek. Sprawozdanie zostało zawarte w dodatku A.

- **Opis wersji**

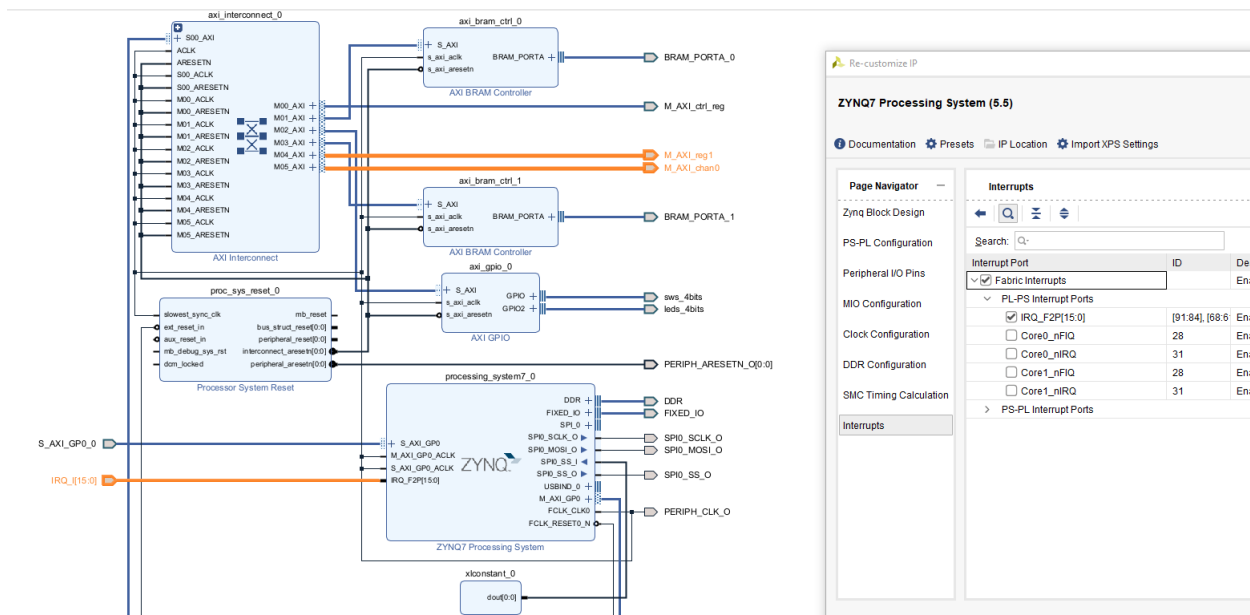
Postęp prac został zakończony wygenerowaniem wersji 0x3_80, pozwalającą na wywołaniu przerwania od zdefiniowanego kanału przetwarzania online w momencie zdefiniowanej ilości przekroczeń licznika przepełnień generatora kodu rozpraszającego. Dodano możliwość rozpoczęcia pracy kanału online w momencie zdefiniowanym odgórnie, w zależności od globalnego licznika 64 bitowego. Wcześniej zdefiniowane przerwania w kierunku PL-PS, wraz z kontrolerem przerwań, zostały zaimplementowane oraz zweryfikowane przez oprogramowanie wbudowanego mikroprocesora ARM części PS. Stworzono filtr DLL/PLL oraz przetestowano implementację analogiczną do kodu C.

- ***Wersja 3_80***

Wersja 3_80 została wykonana 29.04.2022 oraz została wstępnie przetestowana za pomocą wbudowanej konsoli xsct oprogramowania Vitis.

System z tej wersji został zawarty w modułach:

zynq_bd_wrapper: dodano 2 dodatkowe porty axi dla nowych rejestrów axi_reg1 oraz axi_chan0. Dodano obsługę przerwań z poziomu block designu.



ce_ctrl_cnt: komponent zajmujący się wyborem odpowiedniego źródła danych i podawanie na tor przetwarzania danych odbiornika. Komponent posiada dodatkową funkcję pomijania zadanej liczby próbek. Zmodyfikowany komponent ce_ctr o możliwość wystartowania odpowiedniego kanału w zadanym momencie zależy od globalnego licznika w trybie przetwarzania online.

ctrl_reg – jest to rejestr 32x32 bity, którego głównym zadaniem jest kontrolowanie systemu poprzez zawarte w nim rejestry. Dostęp do tego komponentu jest wykonywany poprzez interfejs AXI4-Lite. Mapa rejestrów znajduje się w załączonym repozytorium. Dodana obsługa 8 kanałów została wykonana poprzez mapowanie odpowiednich rejestrów w zależności od wybranego kanału. Pozwala to na wykorzystanie jednolitej przestrzeni adresowej dla wszystkich kanałów bez potrzeby nadmiarowego generowania dodatkowych rejestrów kontrolnych. Rejestry mapowane w zależności od zdefiniowanego kanału zostały zaznaczone kolorem czerwonym.

reg_axi_0

Nazwa	Adres	Bit	Opis	Default	Tryb
Major ver	0x00	31:0	Wersja buildu major	0x3	RO
Major ver	0x04	31:0	Wersja buildu minor	0x80	RO
Test reg	0x08	31:28	-	0x0	RO
		27:24	Ilość kanałów offline	0x0	RO
		23:19	-	0x0	RO
		18:16	Aktualny kanał offline, 0 – 7 wskazuje odpowiedni kanał	0x0	RW
		15:0	Rejestr testowy, wpisana wartość zostaje zanegowana	0x0	RW
Transfer size	0x0c	31:0	Wielkość transferu do DDR (w słowach 4B)	0x8000	RW
Start channel	0x10	31:28	Ilość kanałów fizycznych (ilość układów max)	0x3	RO
		27:25	-	0x0	RO
		24:19	-	0x0	RO
		18:16	Stop write online channel (max 0, max 1, max 2 data stream)	0x0	WO
		15:13	Start write online channel (max 0, max 1, max 2 data stream)	0x0	WO
		12	Done memory read ddr	0	W1C

		<u>11</u>	<u>Busy memory read ddr</u>	<u>0</u>	<u>RO</u>
		<u>10</u>	<u>Busy online channel</u>	<u>0</u>	<u>RO</u>
		<u>9</u>	<u>Reset NCO</u>	<u>0</u>	<u>WO</u>
		<u>8</u>	<u>Busy DDR</u>	<u>0</u>	<u>RO</u>
		<u>7</u>	<u>Busy bram</u>	<u>0</u>	<u>RO</u>
		<u>6</u>	<u>Flaga done DDR</u>	<u>0</u>	<u>W1C</u>
		<u>5</u>	<u>Flaga done bram</u>	<u>0</u>	<u>W1C</u>
		<u>4</u>	<u>Start read channel DDR</u>	<u>0</u>	<u>WO</u>
		<u>3:0</u>	<u>Start write channel</u>	<u>0x0</u>	<u>WO</u>
<u>Counter Latch MSB</u>	<u>0x14</u>	<u>31:0</u>	<u>63:32 bity globalnego licznika, zatrzaskiwane w momencie wpisu do bitu start channel</u>	<u>0x0</u>	<u>RO</u>
<u>Counter Latch LSB</u>	<u>0x18</u>	<u>31:0</u>	<u>31:0 bity globalnego licznika, zatrzaskiwane w momencie wpisu do bitu start channel</u>	<u>0x0</u>	<u>RO</u>
<u>Counter Live MSB</u>	<u>0x1c</u>	<u>31:0</u>	<u>63:32 bity globalnego licznika, wartość aktualna</u>	<u>-</u>	<u>RO</u>
<u>Counter Live LSB</u>	<u>0x20</u>	<u>31:0</u>	<u>31:0 bity globalnego licznika, wartość aktualna</u>	<u>-</u>	<u>RO</u>
<u>DDR start address</u>	<u>0x24</u>	<u>31:0</u>	<u>Adres ddr zapisu danych</u>	<u>0x200000</u>	<u>RW</u>
<u>FCW mixer</u>	<u>0x28</u>	<u>31:0</u>	<u>FCW dla NCO mixer</u>	<u>0x1FFFFFFF</u>	<u>R/WS</u>
<u>Mixer test</u>	<u>0x2C</u>	<u>31:24</u>	<u>-</u>	<u>0x00</u>	<u>RO</u>
		<u>23:16</u>	<u>Dane z wyjścia mixera (23:20 I) (19:16 Q)</u>	<u>0x00</u>	<u>RO</u>
		<u>15:10</u>	<u>-</u>	<u>-</u>	<u>RO</u>
		<u>9:8</u>	Tryb miksera: „00” - mode 0: $I = i * \cos - q * \sin;$ $Q = q * \cos + i * \sin;$ „01” - mode 1: $I = i * \cos + q * \sin;$ $Q = q * \cos - i * \sin;$ „10” - mode 2: $I = i * \cos + q * \sin;$ $Q = -q * \cos + i * \sin;$ „11” - mode 3: $I = i * \cos - q * \sin;$ $Q = -q * \cos - i * \sin;$	<u>00</u>	<u>RW</u>
		<u>7:4</u>	<u>-</u>	<u>-</u>	<u>RO</u>
		<u>3:0</u>	<u>Dane I i Q na wejście mixera (3:2 I) (1:0 Q)</u>	<u>0x0</u>	<u>R/WS</u>
<u>FCW code</u>	<u>0x30</u>	<u>31:0</u>	<u>FCW dla NCO kodu</u>	<u>0x1FFFFFFF</u>	<u>R/WS</u>
<u>Code ctrl</u>	<u>0x34</u>	<u>31</u>	<u>-</u>	<u>-</u>	<u>RO</u>
		<u>30</u>	<u>Wymuś kod</u>	<u>'0'</u>	<u>W1C</u>
		<u>29</u>	<u>-</u>	<u>'0'</u>	<u>RO</u>
		<u>28:25</u>	<u>Wymuszony wskaźnik chip_div</u>	<u>0x0</u>	<u>RW</u>
		<u>24:20</u>	<u>Aktualny kod (MSB VE,E,P,L,VL LSB)</u>	<u>0x00</u>	<u>RO</u>
		<u>19:16</u>	<u>chip_div - ilość przepełnień nco kodu po którym zostanie wczytany nowy bit chip (0x0 - 1, 0x1 - 2, 0x2 - 3 itd...)</u>	<u>0x0</u>	<u>R/WS</u>
		<u>15:14</u>	<u>spacing_len - ilość przepełnień nco kodu po którym zostaje przesunięty wektor kodu veplv_code ("00" - 1, "01" - 2 "10" - 3 "11" - 4)</u>	<u>„10”</u>	
		<u>13:0</u>	<u>Długość ciągu kodowego (w bitach-1)</u>	<u>0x03FE</u>	
<u>Code data0</u>	<u>0x38</u>	<u>31:28</u>	<u>Dane z wyjścia VE I (kod)</u>	<u>0x0</u>	<u>RO</u>
		<u>27:24</u>	<u>Dane z wyjścia VE Q (kod)</u>	<u>0x0</u>	<u>RO</u>

		<u>23:20</u>	<u>Dane z wyjścia E I (kod)</u>	<u>0x0</u>	<u>RO</u>
		<u>19:16</u>	<u>Dane z wyjścia E Q (kod)</u>	<u>0x0</u>	<u>RO</u>
		<u>15:12</u>	<u>Dane z wyjścia P I (kod)</u>	<u>0x0</u>	<u>RO</u>
		<u>11:8</u>	<u>Dane z wyjścia P Q (kod)</u>	<u>0x0</u>	<u>RO</u>
		<u>7:4</u>	<u>Dane z wyjścia L I (kod)</u>	<u>0x0</u>	<u>RO</u>
		<u>3:0</u>	<u>Dane z wyjścia L Q (kod)</u>	<u>0x0</u>	<u>RO</u>
<u>Code data1</u>	<u>0x3C</u>	<u>31:28</u>	<u>Dane z wyjścia VL I (kod)</u>	<u>0x0</u>	<u>RO</u>
		<u>27:24</u>	<u>Dane z wyjścia VL Q (kod)</u>	<u>0x0</u>	<u>RO</u>
		<u>23:17</u>	<u>-</u>	<u>0x0</u>	<u>RO</u>
		<u>16</u>	<u>Wyczyść Integrate & Dump</u>	<u>0x0</u>	<u>W1C</u>
		<u>15</u>	<u>-</u>	<u>0</u>	<u>RO</u>
		<u>14:0</u>	<u>Wymuszony indeks chip</u>	<u>0x0</u>	<u>RO</u>
<u>I&D data 0</u>	<u>0x40</u>	<u>31:0</u>	<u>Dane z wyjścia VE I (Integrate & dump)</u>	<u>0x0</u>	<u>RO</u>
<u>I&D data 1</u>	<u>0x44</u>	<u>31:0</u>	<u>Dane z wyjścia VE Q (Integrate & dump)</u>	<u>0x0</u>	<u>RO</u>
<u>I&D data 2</u>	<u>0x48</u>	<u>31:0</u>	<u>Dane z wyjścia E I (Integrate & dump)</u>	<u>0x0</u>	<u>RO</u>
<u>I&D data 3</u>	<u>0x4C</u>	<u>31:0</u>	<u>Dane z wyjścia E Q (Integrate & dump)</u>	<u>0x0</u>	<u>RO</u>
<u>I&D data 4</u>	<u>0x50</u>	<u>31:0</u>	<u>Dane z wyjścia P I (Integrate & dump)</u>	<u>0x0</u>	<u>RO</u>
<u>I&D data 5</u>	<u>0x54</u>	<u>31:0</u>	<u>Dane z wyjścia P Q (Integrate & dump)</u>	<u>0x0</u>	<u>RO</u>
<u>I&D data 6</u>	<u>0x58</u>	<u>31:0</u>	<u>Dane z wyjścia L I (Integrate & dump)</u>	<u>0x0</u>	<u>RO</u>
<u>I&D data 7</u>	<u>0x5C</u>	<u>31:0</u>	<u>Dane z wyjścia L Q (Integrate & dump)</u>	<u>0x0</u>	<u>RO</u>
<u>I&D data 8</u>	<u>0x60</u>	<u>31:0</u>	<u>Dane z wyjścia VL I (Integrate & dump)</u>	<u>0x0</u>	<u>RO</u>
<u>I&D data 9</u>	<u>0x64</u>	<u>31:0</u>	<u>Dane z wyjścia VL Q (Integrate & dump)</u>	<u>0x0</u>	<u>RO</u>
<u>Memory read len</u>	<u>0x68</u>	<u>31:27</u>	<u>-</u>	<u>0x0</u>	<u>RO</u>
		<u>26:0</u>	<u>Ilość danych do przetwarzania z DDR (w słowach 32B)</u>	<u>0x1000</u>	<u>RW</u>
<u>Ce ctrl</u>	<u>0x6C</u>	<u>31:7</u>	<u>-</u>	<u>0x0</u>	<u>RO</u>
		<u>6</u>	<u>Set ce ctrl</u>	<u>0x0</u>	<u>W1C</u>
		<u>5:4</u>	<u>Mode ce ctrl: „00” - software data „01” - max data „10” - ddr read data</u>	<u>0x0</u>	<u>RW</u>
		<u>3:0</u>	<u>Drop data (saples)</u>	<u>0x0</u>	<u>RW</u>
<u>Force CE code NCO</u>	<u>0x70</u>	<u>31:0</u>	<u>Wymuszona nastawa NCO kodu</u>	<u>0x0</u>	<u>RW</u>
<u>Reserved</u>	<u>0x74 - 0x7C</u>		<u>-</u>	<u>0x0</u>	<u>RO</u>

axi_reg16 – jest to rejestr 16x32 bity, dostępny poprzez interfejs AXI4-Lite. Głównym zadaniem komponentu jest konfiguracja oraz zarządzanie komponentem filtra oraz kontrolera przerwań. Poniżej aktualna mapa rejestrów:

reg_axi_1

<u>Nazwa</u>	<u>Adres</u>	<u>Bit</u>	<u>Opis</u>	<u>Default</u>	<u>Tryb</u>
Component ver	0x00	31:0	ID danego komponentu axi	0x3	RO
Coeff 0	0x04	31:19	-	-	RO
		18	Ustaw wartości współczynnika 0 i 1	0x0	W1C
		17:0	Signed współczynnik 0	0x0	RW
Coeff 1	0x08	31:18	-	-	RO
		17:0	Signed współczynnik 1	0x0	RW
Filter data in	0x0C	31:26	-	-	RO
		25	Wpisz dane na wejście filtra	0x0	W1C
		24:0	Signed dane wejściowe filtra	0x0	RW
Filter data out0	0x10	31:0	Dane wyjściowe z filtra 31-0	0x0	RO
Filter data out1	0x14	31:13	-	-	RO

		12:0	Dane wyjściowe z filtra 44-32	0x0	RO
IRQ reg 0	0x18	31:16	IRQ type – 0 high lvl 1 pulse	0x0	RW
		15:0	IRQ status / IRQ clear	0x0	W1C
IRQ reg 1	0x1C	31:16	IRQ enable	0x0	RW
		15:0	IRQ force	0x0	RW
Filter state 0	0x20	31:0	Dane state filtra(31 – 0) in/out signed	0x0	RW
Filter state 1	0x24	31:12	-	-	RO
		11:0	Dane state filtra(43 - 32) in/out signed	0x0	RW
<u>Reserved</u>	0x28 - 0x3C		-	=	<u>RO</u>

axi_reg16_new – komponent rejestrów axi 16x32 bit, wykonany w celu przyszłego wykorzystania w przypadku tworzenia nowych IP rejestrów z dostępem przez interfejs AXI4-lite.

axi_reg16_channel - jest to rejestr 16x32 bity, dostępny poprzez interfejs AXI4-Lite. Głównym zadaniem komponentu jest konfiguracja oraz zarządzanie funkcjonalnością przerwania po zadanej liczbie przepełnień kodu generatora. Posiada rejestry zatrzymujące wartości akumulatorów/integrate, pozwala zdefiniować liczbę N przepełnień kodu oraz posiada funkcjonalność startu zadanego kanału online przy odpowiedniej wartości globalnego licznika. Poniżej aktualna mapa rejestrów:

reg_chan 0

<u>Nazwa</u>	<u>Adres</u>	<u>Bit</u>	<u>Opis</u>	<u>Default</u>	<u>Tryb</u>
Component ver	0x00	31:0	ID danego komponentu axi	0x10	RO
N overflow	0x04	31:9	-	-	RO
		8	Start kanału ok	0x0	W1C
		7	Stop kanału online	0x0	WO
		6	Ustaw licznik startu kanału online	0x0	WO
		5	Ustaw wartość licznika przepełnień	0x0	WO
		4:0	Liczba przepełnień N kodu po której zostanie wysłane przerwanie	0x0	RW
I&D data 0	0x8	31:0	Zatrzaśnięte dane z wyjścia VE I (Integrate & dump)	0x0	RO
I&D data 1	0xC	31:0	Zatrzaśnięte dane z wyjścia VE Q (Integrate & dump)	0x0	RO
I&D data 2	0x10	31:0	Zatrzaśnięte dane z wyjścia E I (Integrate & dump)	0x0	RO
I&D data 3	0x14	31:0	Zatrzaśnięte dane z wyjścia E Q (Integrate & dump)	0x0	RO
I&D data 4	0x18	31:0	Zatrzaśnięte dane z wyjścia P I (Integrate & dump)	0x0	RO
I&D data 5	0x1C	31:0	Zatrzaśnięte dane z wyjścia P Q (Integrate & dump)	0x0	RO
I&D data 6	0x20	31:0	Zatrzaśnięte dane z wyjścia L I (Integrate & dump)	0x0	RO
I&D data 7	0x24	31:0	Zatrzaśnięte dane z wyjścia L Q (Integrate & dump)	0x0	RO
I&D data 8	0x28	31:0	Zatrzaśnięte dane z wyjścia VL I (Integrate & dump)	0x0	RO
I&D data 9	0x2C	31:0	Zatrzaśnięte dane z wyjścia VL Q (Integrate & dump)	0x0	RO
Counter Latch LSB	0x30	31:0	31:0 bity globalnego licznika, zatrzaskiwane w momencie N przepełnienia kodu	0x0	RO
Counter Latch	0x34	31:0	63:32 bity globalnego licznika, zatrzaskiwane w	0x0	RO

MSB			momencie N przepełnienia kodu		
Online start LSB	0x38	31:0	Bit 31:0 startu przetwarzania danych online kanału	0x0	RW
Online start MSB	0x3C	31:0	Bit 63:32 startu przetwarzania danych online kanału	0x0	RW

code_ce_irq – zmodyfikowany komponent generatora kodu rozpraszającego o możliwość generowania zewnętrznego przerwania po zadanej ilości przepełnień ciągu kodowego.

dyskr_dll – opracowywany komponent dyskryminatora dll, bez końcowego etapu dzielenia. Komponent w opracowaniu, należy przystosować komponent do wygenerowanego IP Xilinx divide generator 5.1 w celu zakończenia pracy nad nim.

dyskr_pll – opracowywany komponent dyskryminatora pll, bez końcowego etapu dzielenia. Komponent w opracowaniu, należy przystosować komponent do wygenerowanego IP Xilinx divide generator 5.1 w celu zakończenia pracy nad nim.

Filtr_v1/2/3 – wersja filtra DLL/PLL wygenerowana na podstawie kodu C. Wersja 1 zawiera zminimalizowaną architekturę filtra, przystosowaną pod minimalizację zasobów FPGA. Wykorzystuje zaledwie 2 komponenty DSP, w celu realizacji zadanej funkcjonalności. Wersja 2 jest wersją testową wygenerowaną w celu sprawdzenia wykorzystania komponentów DSP przy zdefiniowaniu wejść jako 64-bitowe wektory. Przy implementacji ilość wykorzystywanych komponentów DSP wzrosła do 32. Wersja 3 jest wersją z dodatkowym wejściem status, pozwalającym na wprowadzeniu dodatkowej zmiennej status, zawierającej zatrzaśniętą wartość przeliczoną z poprzedniej operacji przetwarzania przez filtr. Implementacja ta powinna być najbliższa zaprezentowanej implementacji kodu, z możliwością wykorzystania jej przy obliczania danych filtra PLL/DLL na zmianę. W celu minimalizacji wykorzystania zasobów FPGA, założone została możliwość wykorzystywania ww. komponentu w przypadku różnych kanałów online dla PLL i DLL.

integrate_irq – zmodyfikowany komponent akumulatora Integrate and dump z rozszerzoną funkcjonalnością o zatraskiwanie wartości akumulatorów w przypadku otrzymania sygnału ze strony code_ce_irq. Komponent również generuje sygnały potrzebne do zatrzaśnięcia wartości akumulatorów dla komponentu axi_reg16_channel.

irq_ctrl – kontroler przerwania przystosowany do zastosowania na platformie ZYNQ. Zaimplementowana funkcjonalność pozwala na maskowanie przerwania, zatraskiwanie wartości przychodzących przerwania, czyszczenia flag aktualnych przerwania oraz definiowania typu przerwania (pule lub high level). Komponent został zaimplementowany do obsługi 16 przerwania do części PS. Kontrola komponentu odbywa się przez rejestr axi_reg1.

TOP_MODULE – komponent wrappera topowego modułu projektu. Zostały dodane odpowiednie komponenty zawarte w tej wersji, dodano interfejsy axi z block design do nowo utworzonych rejestrów axi.

Reszta komponentów nie uległa znacznej zmianie.
Poniżej raport z użycia wersji 0x3_80:

Utilization				Post-Synthesis	Post-Implementation
				Graph	Table
Resource	Utilization	Available	Utilization %		
LUT	12458	53200	23.42		
LUTRAM	706	17400	4.06		
FF	17565	106400	16.51		
BRAM	50.50	140	36.07		
DSP	2	220	0.91		
IO	16	125	12.80		
BUFG	2	32	6.25		

Warto zaznaczyć że niektóre części implementacji zostaną zmodyfikowane/usunięte w przyszłych wersjach, co pozwoli na zmniejszenie zajętości pojedynczego kanału oraz dostępność większego obszaru FPGA (data_packer oraz część block designu zostanie usunięta ze względu na brak potrzeby testowania pewnej części kodu w następnych wersjach). Ze wstępnej użycia dla 8 kanałów, zakłada się brak problemów przy implementacji 16 kanałów online.

Poniżej raport czasowy oraz raport pobieranej mocy przez układ FPGA.

Timing		Setup	Hold	Pulse Width
Worst Negative Slack (WNS):	5.756 ns			
Total Negative Slack (TNS):	0 ns			
Number of Failing Endpoints:	0			
Total Number of Endpoints:	46243			
Implemented Timing Report				

Power		Summary	On-Chip
Total On-Chip Power:	1.606 W		
Junction Temperature:	43,5 °C		
Thermal Margin:	41,5 °C (3,4 W)		
Effective θ_{JA} :	11,5 °C/W		
Power supplied to off-chip devices:	0 W		
Confidence level:	Low		
Implemented Power Report			

W porównaniu do poprzedniej wersji kodu, pogorszył się WNS. Jest to nadal akceptowalna wartość, jednak należy mieć na uwadze potrzebę modyfikacji niektórych komponentów w przyszłości w celu maksymalizacji timingu. Czasówki pogorszyły się po implementacji filtrów wykorzystujących elementy DSP FPGA oraz komparatorów globalnego licznika 64 bit. W przypadku filtrów powinien wystarczyć odpowiedni pipelining przychodzących danych. Gorsza sprawa może być w przypadku globalnego licznika. Ze względu na wymóg porównywania długiego wektora bitowego, implementację na zasadzie LUT oraz wymogi czasowe może być to krytyczna część FPGA w przypadku zwiększonej zajętości układu. Problematiczny jest wymóg porównywania długiego wektora z wymogiem generowania sygnału startu kanału online w jednym cyklu zegarowym. Jeżeli istniałoby rozwiązanie umożliwiające opóźnienie startu o kilka cykli zegarowych, przez co możliwe by było rozbieżenie części komparacyjnej na kilka cykli, na pewno pozwoliłoby to lepsze czasówki. Być może rozwiązanie wpisywania wartości globalnego licznika przesuniętego o odpowiednią ilość cykli była by wystarczająca. Istnieje również możliwość pipelingu całej części ścieżki danych, bez potrzeby generowania wyzwalania przesuniętego o kilka cykli, jednak wymagałoby to implementacji dodatkowego pipeliningu dla dużej części komponentów. Również w takim wypadku globalny licznik byłby przesunięty w czasie w stosunku do całej ścieżki danych. Jest to temat do przedyskutowania oraz jak na razie najbardziej newralgiczny element designu.

- **Rozwój oprogramowania**

Oprogramowanie zapewnia w pełni możliwość komunikacji oraz pozwala na testy od strony procesora (PS) SoC Zynq przy odpowiednich scenariuszach. Dzięki implementacji przerwań, został udoskonalony sposób komunikacji pomiędzy częścią logiki programowalnej a procesorem.

Odpowiednie wersje oprogramowania zostały zapakowane w repozytoria z dokumentacją oraz niezbędnymi plikami do stworzenia całego projektu pod vivado. Dostarczone zostały pliki xsa zawierające wygenerowane pliki bistream potrzebne do oprogramowania części PL soc zynq z poziomu oprogramowania Vitis. Została dołączona dokumentacja oraz opis tworzenia i uruchomienia systemu z poziomu Vitis.

Repozytorium po rozpakowaniu:

doc – folder zawierający dokumentację:

-opis.docx – opis przestrzeni adresowej części PL, mapę rejestrów komponentu ctrl_reg, opis i podstawowe przypadki uruchomienia kodu z poziomu MCU

-vitis.docx – sposób stworzenia projektu Vitis, wczytania wymaganych plików w celu uruchomienia systemu, sposób korzystania z xsct, przykładowy scenariusz działania, odczytanie otrzymanych danych, wykonanie memorydump, konwersja memory dumpa z formy binarnej.

-ila.docx – opis wykorzystywania core Xilinx Integrated Logic Analyser oraz sposób generowania dla kodu RTL. Przykładowa funkcjonalność oraz scenariusz został przedstawiony na bazie wersji -0x3_7B dla kontrolera przerwań irq_ctrl.

Src – folder zawierający wszystkie pliki vhd, bd oraz bloki IP potrzebne do wygenerowania projektu z poziomu vivado:

- bd – folder zawiera block design systemu zynq_main oraz wrapper potrzeby aby zaimplementować block design z poziomu vhd

- const – plik zawierający plik constrain, definiujący piny zewnętrzne

- ip – plik zawierający wygenerowane bloki z IP katalogu Vivado

- probes – plik próbów wykorzystywany przy korzystaniu z IP ILA,

- vhd – katalog wszystkich plików vhd wykorzystywanych przy generowaniu projektu vivado

create_run.tcl – skrypt tworzący design run przy generowaniu projektu vivado

filelist.tcl – skrypt wczytujący wszystkie potrzebne pliki przy generowaniu projektu vivado

sw – zawiera plik main.c potrzebny przy uruchomieniu SW za pomocą oprogramowania vitis

gogeo_3_80.xsa – plik hw wyeksportowany po zbudowaniu bitstreamu w wersji 3_80, potrzebny przy uruchomieniu SW za pomocą oprogramowania vitis

make_project.tcl – skrypt tcl do stworzenia projektu pod vivado, należy uruchomić oprogramowanie Vivado, z konsoli tcl przejść do katalogu z plikiem, uruchomić skrypt za pomocą komendy source ./make_project.tcl

README.txt – krótki opis

• Bibliografia

- UG585 Zynq-7000 SoC Technical Reference Manual
- AMBA AXI and ACE ProtocolSpecification

• Dodatki

- Generowanie danych z pliku binarnego: http://tomeko.net/online_tools/file_to_hex.php?lang=en

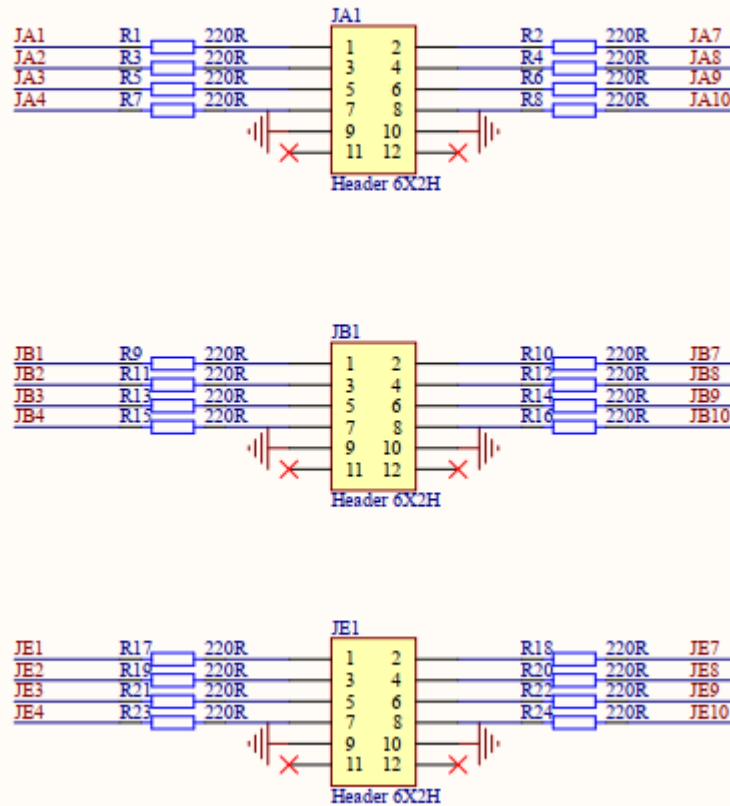
Dodatek A

Weryfikacja pinów dla układu Zynq w projekcie gogeo. 22.03.2022

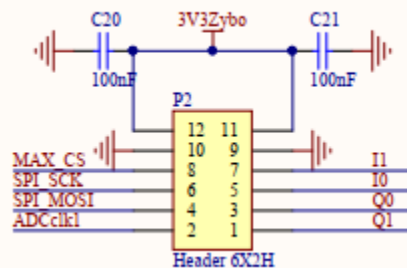
Podstawowa weryfikacja zdefiniowanych pinów od strony PL (FPGA) platformy Zynq została wykonana za pomocą narzędzia floorplan oprogramowania Vitis 2021.2 dla układu Zynq xc7z020clg484-1. Dane do weryfikacji zostały pobrane z załączonego schematu OnGeo.pdf.

W celu wstępnej weryfikacji został utworzony projekt Vivado z odpowiedni plikiem constraints, który definiuje umieszczenie zewnętrznych pinów w stosunku do wewnętrznej, programowalnej logiki.

Do banków PL zostały doprowadzone 3 zestawy pinów wyprowadzone na headery:



Funkcje odpowiednich pinów zostały wzięte z poprzedniego schematu definiującego headery następująco:



Został zdefiniowany plik constraints phys_const.xdc:

```
#port A
set_property -dict { PACKAGE_PIN A18 IOSTANDARD LVCMOS33 } [get_ports { I0[0] }]; #JA3 5HDR I0
set_property -dict { PACKAGE_PIN C18 IOSTANDARD LVCMOS33 } [get_ports { I0[1] }]; #JA4 7HDR I1
set_property -dict { PACKAGE_PIN B17 IOSTANDARD LVCMOS33 } [get_ports { Q0[0] }]; #JA2 3HDR Q0
set_property -dict { PACKAGE_PIN A19 IOSTANDARD LVCMOS33 } [get_ports { Q0[1] }]; #JA1 1HDR Q1
set_property -dict { PACKAGE_PIN B16 IOSTANDARD LVCMOS33 } [get_ports { ADC_CLK0 }]; #JA7 2HDR ADC_CLK
set_property -dict { PACKAGE_PIN A17 IOSTANDARD LVCMOS33 } [get_ports { SPI0_DATA_O }]; #JA8 4HDR SPI_MOSI
set_property -dict { PACKAGE_PIN A16 IOSTANDARD LVCMOS33 } [get_ports { SPI0_CLK }]; #JA9 6HDR SPI_SCK
set_property -dict { PACKAGE_PIN C17 IOSTANDARD LVCMOS33 } [get_ports { SPI0_CS }]; #JA10 8HDR MAX_CS

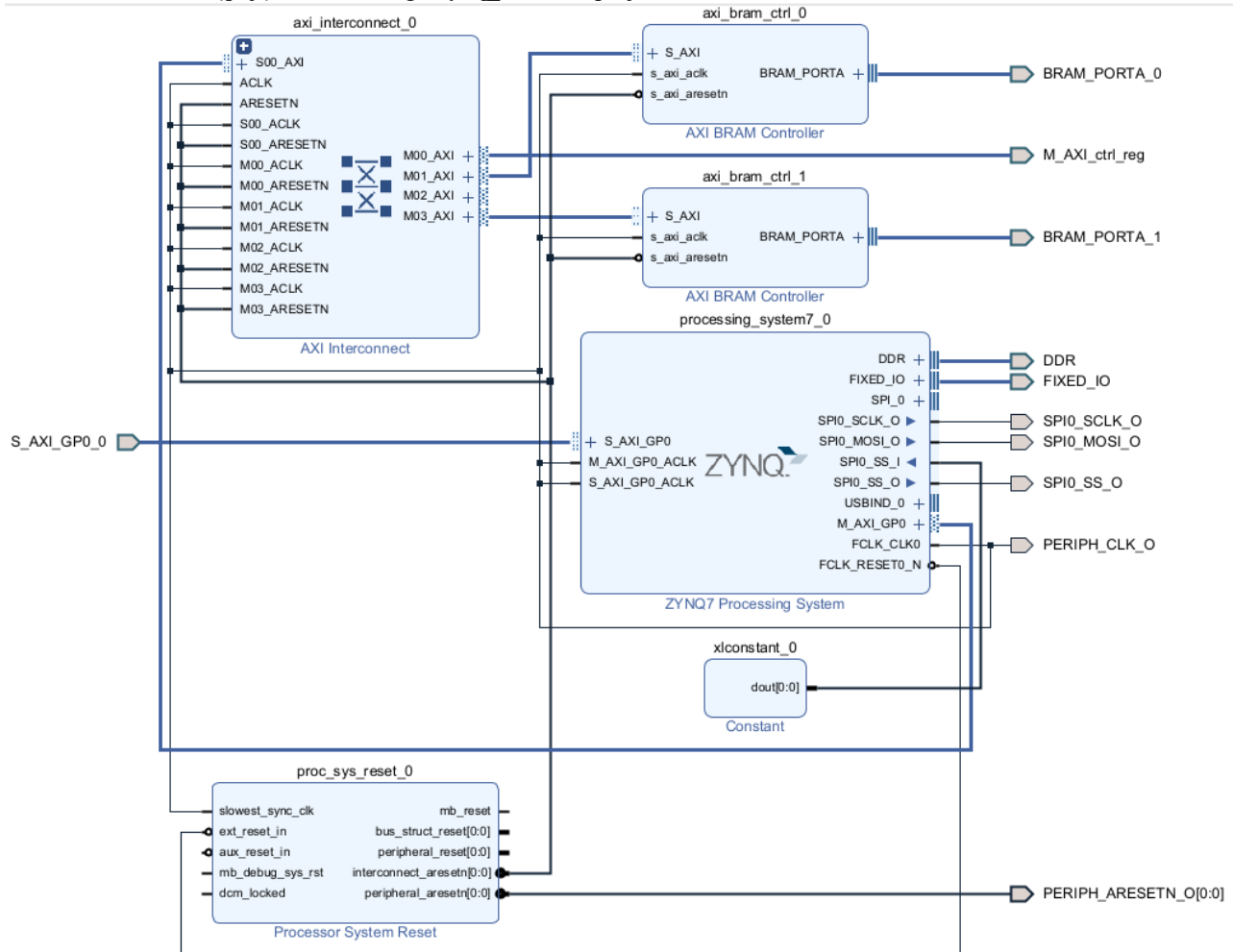
#port B
set_property -dict { PACKAGE_PIN AB21 IOSTANDARD LVCMOS33 } [get_ports { I1[0] }]; #JB3 5HDR I0
set_property -dict { PACKAGE_PIN AA19 IOSTANDARD LVCMOS33 } [get_ports { I1[1] }]; #JB4 7HDR I1
set_property -dict { PACKAGE_PIN Y20 IOSTANDARD LVCMOS33 } [get_ports { Q1[0] }]; #JB2 3HDR Q0
set_property -dict { PACKAGE_PIN Y21 IOSTANDARD LVCMOS33 } [get_ports { Q1[1] }]; #JB1 1HDR Q1
set_property -dict { PACKAGE_PIN AA22 IOSTANDARD LVCMOS33 } [get_ports { ADC_CLK1 }]; #JB7 2HDR ADC_CLK
set_property -dict { PACKAGE_PIN AB22 IOSTANDARD LVCMOS33 } [get_ports { SPI1_DATA_O }]; #JB8 4HDR SPI_MOSI
```

```

set_property -dict { PACKAGE_PIN AB20 IOSTANDARD LVCMOS33 } [get_ports { SPI1_CLK }]; #JB9 6HDR SPI_SCK
set_property -dict { PACKAGE_PIN AB19 IOSTANDARD LVCMOS33 } [get_ports { SPI1_CS }]; #JB10 8HDR MAX_CS
#port E
set_property -dict { PACKAGE_PIN E21 IOSTANDARD LVCMOS33 } [get_ports { I2[0] }]; #JE3 5HDR I0
set_property -dict { PACKAGE_PIN D21 IOSTANDARD LVCMOS33 } [get_ports { I2[1] }]; #JE4 7HDR I1
set_property -dict { PACKAGE_PIN C22 IOSTANDARD LVCMOS33 } [get_ports { Q2[0] }]; #JE2 3HDR Q0
set_property -dict { PACKAGE_PIN D22 IOSTANDARD LVCMOS33 } [get_ports { Q2[1] }]; #JE1 1HDR Q1
set_property -dict { PACKAGE_PIN B21 IOSTANDARD LVCMOS33 } [get_ports { ADC_CLK2 }]; #JE7 2HDR ADC_CLK
set_property -dict { PACKAGE_PIN B22 IOSTANDARD LVCMOS33 } [get_ports { SPI2_DATA_O }]; #JE8 4HDR SPI_MOSI
set_property -dict { PACKAGE_PIN H19 IOSTANDARD LVCMOS33 } [get_ports { SPI2_CLK }]; #JE9 6HDR SPI_SCK
set_property -dict { PACKAGE_PIN H20 IOSTANDARD LVCMOS33 } [get_ports { SPI2_CS }]; #JE10 8HDR MAX_CS

```

Zdefiniowano następująco block design zynq_main dla projektu:



W porównaniu do płyty ewaluacyjnej został usunięty obszar pamięci, oraz komponent odpowiedzialny za sterowanie led oraz switchami. Została zdefiniowana pamięć w komponencie zynq (model MT41K256M16TW-107:P), ze względu na analogiczne piny, ustawienia pamięci zostały nie zmieniony.

Zostały dodane pliki projektu z wersji 0x3_48, wraz z dodanymi modyfikacjami: usunięte porty led oraz sws, dodano multipleksację spi poprzez rejestr Test reg 0x08 20:19. Zmodyfikowano wersję do 0x4_00. Multipleksacja SPI została wykonana za pomocą sterowania sygnałem spi_cs, przypisano sygnał data oraz clk do wszystkich 3ch interfejsów:

```

spi0_cs_out <= '1' when v_spi_chan /= "00" else spi_cs_out;
spi1_cs_out <= '1' when v_spi_chan /= "01" else spi_cs_out;

```

```
spi2_cs_out <= '1' when v_spi_chan /= "10" else spi_cs_out;
```

```
SPI0_DATA_O <= spi_data_out;
SPI0_CLK <= spi_clk_out;
SPI0_CS <= spi0_cs_out;
SPI1_DATA_O <= spi_data_out;
SPI1_CLK <= spi_clk_out;
SPI1_CS <= spi1_cs_out;
SPI2_DATA_O <= spi_data_out;
SPI2_CLK <= spi_clk_out;
SPI2_CS <= spi2_cs_out;
```

Przystąpiono do weryfikacji pinoutu za pomocą floorplaning po poprawnej syntezie (open synthetised design → constraints wizard):

SYNTHESIZED DESIGN - synth_1 xc7z020dpg484-1													
Tcl Console Messages Log Reports Design Runs IP Status Package Pins IO Ports x													
Name	Direction	Neg Diff Pair	Package Pin	Fixed	Bank	I/O Std	Vcco	Vref	Drive Strength	Slew Type	Pull Type	Off-Chip Termination	IN_TERM
CLK_SPI0_SCLK_O_5096 (3)	OUT			✓	(Multiple)	LVCNMOS33*	3.300		12		NONE	FP_VTT_50	
Scalar ports (3)													
SPI0_CLK	OUT		A16	✓	35	LVCNMOS33*	3.300		12		NONE	FP_VTT_50	
SPI1_CLK	OUT		AB20	✓	33	LVCNMOS33*	3.300		12		NONE	FP_VTT_50	
SPI2_CLK	OUT		H19	✓	35	LVCNMOS33*	3.300		12		NONE	FP_VTT_50	
DDR_5095 (71)	INOUT			✓	502	(Multiple)*	1.350	(Multiple)		(Multiple)	NONE	FP_VTT_50	(Multiple)
FIXED_JO_5096 (59)	INOUT			✓	(Multiple)	(Multiple)*	(Multiple)	(Multiple)		(Multiple)	(Multiple)	(Multiple)	
SPI0_57874 (3)	OUT			✓	(Multiple)	LVCNMOS33*	3.300		12		NONE	FP_VTT_50	
Scalar ports (3)													
SPI0_DATA_O	OUT		A17	✓	35	LVCNMOS33*	3.300		12		NONE	FP_VTT_50	
SPI1_DATA_O	OUT		AB22	✓	33	LVCNMOS33*	3.300		12		NONE	FP_VTT_50	
SPI2_DATA_O	OUT		B22	✓	35	LVCNMOS33*	3.300		12		NONE	FP_VTT_50	
I0 (2)	IN			✓	35	LVCNMOS33*	3.300				NONE	NONE	
I1 (2)	IN			✓	33	LVCNMOS33*	3.300				NONE	NONE	
I2 (2)	IN			✓	35	LVCNMOS33*	3.300				NONE	NONE	
Q0 (2)	IN			✓	35	LVCNMOS33*	3.300				NONE	NONE	
Q1 (2)	IN			✓	33	LVCNMOS33*	3.300				NONE	NONE	
Q2 (2)	IN			✓	35	LVCNMOS33*	3.300				NONE	NONE	
Scalar ports (6)													
ADC_CLK0	OUT		B16	✓	35	LVCNMOS33*	3.300		12		NONE	FP_VTT_50	
ADC_CLK1	OUT		AA22	✓	33	LVCNMOS33*	3.300		12		NONE	FP_VTT_50	
ADC_CLK2	OUT		B21	✓	35	LVCNMOS33*	3.300		12		NONE	FP_VTT_50	
SPI0_CS	OUT		C17	✓	35	LVCNMOS33*	3.300		12		NONE	FP_VTT_50	
SPI1_CS	OUT		AB19	✓	33	LVCNMOS33*	3.300		12		NONE	FP_VTT_50	
SPI2_CS	OUT		H20	✓	35	LVCNMOS33*	3.300		12		NONE	FP_VTT_50	

Została sprawdzona poprawność wprowadzonych portów wraz z poniższą tabelą, oraz została uruchomiona generacja bitstream.

Nazwa pinu constraint	Lokalizacja pinu zynq pl	Nazwa portu schematic	Header pin
SPI0_CLK	A16	JA9	6 PIN JA SPI_SCK
SPI1_CLK	AB20	JB9	6 PIN JB SPI_SCK
SPI2_CLK	H19	JE9	6 PIN JE SPI_SCK
SPI0_DATA_O	A17	JA8	4 PIN JA SPI_MOSI
SPI1_DATA_O	AB22	JB8	4 PIN JB SPI_MOSI
SPI2_DATA_O	B22	JE8	4 PIN JE SPI_MOSI
I0[1]	C18	JA4	7 PIN JA I1
I0[0]	A18	JA3	5 PIN JA IO
I1[1]	AA19	JB4	7 PIN JB I1
I1[0]	AB21	JB3	5 PIN JB IO
I2[1]	D21	JE4	7 PIN JE I1
I2[0]	E21	JE3	5 PIN JE IO
Q0[1]	A19	JA1	1 PIN JA Q1
Q0[0]	B17	JA2	3 PIN JA Q0
Q1[1]	Y21	JB1	1 PIN JB Q1
Q1[0]	Y20	JB2	3 PIN JB Q0
Q2[1]	D22	JE1	1 PIN JE Q1

Q2[0]	C22	JE2	3 PIN JE Q0
ADC_CLK0	B16	JA7	2 PIN JA ADCclk1
ADC_CLK1	AA22	JB7	2 PIN JA ADCclk1
ADC_CLK2	B21	JE7	2 PIN JA ADCclk1
SPI0_CS	C17	JA10	8 PIN JA MAX_CS
SPI1_CS	AB19	JB10	8 PIN JB MAX_CS
SPI2_CS	H20	JE10	8 PIN JE MAX_CS

Poprawność pinów została potwierdzona. Wygenerowano nową wersję 0x4_00 pod platformę Zynq xc7z020clg484-1 w celu pełnej weryfikacji.

Overview | Dashboard

Status: ✔ Complete

Messages: 🚨 989 warnings

Active run: synth_1

Part: xc7z020clg484-1

Strategy: Vivado Synthesis Defaults

Report Strategy: Vivado Synthesis Default Reports

Incremental synthesis: Automatically selected checkpoint

DRC Violations

No DRC violations were found.
[Implemented DRC Report](#)

Utilization

Post-Synthesis | **Post-Implementation**

Graph | **Table**

Resource	Utilization	Available	Utilization %
LUT	9094	53200	17.09
LUTRAM	334	17400	1.92
FF	12219	106400	11.48
BRAM	38	140	27.14
IO	16	200	8.00
BUFG	1	32	3.13

Status: ✔ Complete

Messages: 🚨 1 warning

Active run: impl_1

Part: xc7z020clg484-1

Strategy: Vivado Implementation Defaults

Report Strategy: Vivado Implementation Default Reports

Incremental implementation: None

Timing

Setup | Hold | Pulse Width

Worst Negative Slack (WNS): 9.442 ns

Total Negative Slack (TNS): 0 ns

Number of Failing Endpoints: 0

Total Number of Endpoints: 34253

[Implemented Timing Report](#)

Power

Summary | On-Chip Power

Total On-Chip Power: **1.589 W**

Junction Temperature: **43,3 °C**

Thermal Margin: 41,7 °C (3,5 W)

Effective θJA: 11,5 °C/W

Power supplied to off-chip devices: 0 W

Confidence level: Low

[Implemented Power Report](#)