

Sprawozdanie z testów platformy FPGA układu GoGeO

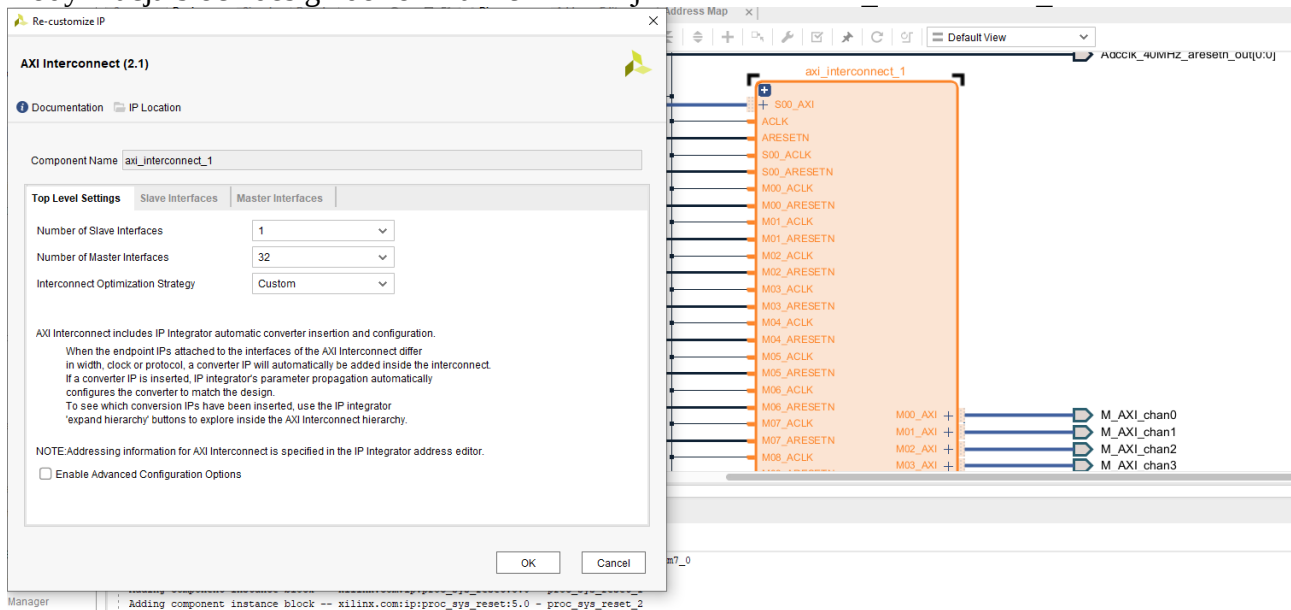
Poniżej przedstawione testy zostały wykonane na platformie zybo dla oprogramowania części PL projektu gogeo.

Wersja HW (major_minor): 0x3_DE

Przeprowadzona seria testów została wykonana w celu sprawdzenia poprawności wygenerowanego bitstreamu dostarczonego wraz z plikiem xsa części PL dla układu zynq. Wykonane testy obejmują scenariusze funkcjonalne dla nowo dodanego kodu. Dodana funkcjonalność kodu obejmuje rozszerzenie liczby dostępnych kanałów offline/online na platformie zynq. Zmienione komponenty obejmuje:

- axi_reg16_meas.vhd
- ctrl_reg.vhd
- global_pkg.vhd
- memory_reader_channel.vhd
- TOP_MODULE.vhd
- zynq_bd_wrapper.vhd
- zynq_main.bd

Zwiększenie liczby dostępnych kanałów zostało wykonane za pomocą modyfikowalnych globalnych zmiennych `C_OFFLINE_CHAN_NUM := 32` oraz `C_IRQ_SRC_NUM:= 32`. O ile modyfikacje w kodzie vhdl mogą być implementowane przy każdej generacji nowego bitstreamu, o tyle część block design jest generowana w momencie tworzenia projektu z katalogu umieszczonego na repozytorium git. Nie wydaje się aby istniała łatwa metoda zmiany liczby dla ww. części kodu oraz autor nie posiada wiedzy w jaki sposób to wykonać. W tym celu wymagana była ręczna modyfikacja block design do rozmiaru 32 interfejsów axi na IP `axi_interconnect_1`.



Każdy z dodanych interfejsów musiały zostać manualnie dodany wraz z połączeniem do portu oraz definicją obszaru pamięci. Powielono następujące komponenty przy zwiększeniu ilości dostępnych kanałów:

- ce_ctrl_cnt
- mixer_ce_meas
- code_ce_irq_meas
- integrate_irq
- axi_reg16_channel

Zwielokrotniono również zawartość rejestru `ctrl_reg` w celach obsługi ww. komponentów.

BLOCK DESIGN - zynq_main

Design | Signals | Board | ? | □ | ✕

Address Path Properties | ? | □ | ✕ | ⚙

Name: /processing_system7_0/M_AXI_GP0 | Base: 0x400

Connections: General | Path | Apertures

Address Editor | Address Map

Assigned (40) | Unassigned (0) | Excluded (3) | Hide All

Name	Interface	Slave Segment	Master Base Addr...	Range	Master High Address
M_AXI_chan13	M_AXI_chan13	Reg	0x4001_D000	4K	0x4001_DFFF
M_AXI_chan14	M_AXI_chan14	Reg	0x4001_E000	4K	0x4001_EFFF
M_AXI_chan15	M_AXI_chan15	Reg	0x4001_F000	4K	0x4001_FFFF
M_AXI_chan16	M_AXI_chan16	Reg	0x4002_0000	4K	0x4002_0FFF
M_AXI_chan17	M_AXI_chan17	Reg	0x4002_1000	4K	0x4002_1FFF
M_AXI_chan18	M_AXI_chan18	Reg	0x4002_2000	4K	0x4002_2FFF
M_AXI_chan19	M_AXI_chan19	Reg	0x4002_3000	4K	0x4002_3FFF
M_AXI_chan20	M_AXI_chan20	Reg	0x4002_4000	4K	0x4002_4FFF
M_AXI_chan21	M_AXI_chan21	Reg	0x4002_5000	4K	0x4002_5FFF
M_AXI_chan22	M_AXI_chan22	Reg	0x4002_6000	4K	0x4002_6FFF
M_AXI_chan23	M_AXI_chan23	Reg	0x4002_7000	4K	0x4002_7FFF
M_AXI_chan24	M_AXI_chan24	Reg	0x4002_8000	4K	0x4002_8FFF
M_AXI_chan25	M_AXI_chan25	Reg	0x4002_9000	4K	0x4002_9FFF
M_AXI_chan26	M_AXI_chan26	Reg	0x4002_A000	4K	0x4002_AFFF
M_AXI_chan27	M_AXI_chan27	Reg	0x4002_B000	4K	0x4002_BFFF
M_AXI_chan28	M_AXI_chan28	Reg	0x4002_C000	4K	0x4002_CFFF
M_AXI_chan29	M_AXI_chan29	Reg	0x4002_D000	4K	0x4002_DFFF
M_AXI_chan30	M_AXI_chan30	Reg	0x4002_E000	4K	0x4002_EFFF
M_AXI_chan31	M_AXI_chan31	Reg	0x4002_F000	4K	0x4002_FFFF
/axi_bram_ctrl_0/S_AXI	S_AXI	Mem0	0x4004_0000	128K	0x4005_FFFF
/axi_gpio_0/S_AXI	S_AXI	Reg	0x4120_0000	64K	0x4120_FFFF

Wstępne test funkcjonalne sprawdzą zaimplementowany kod pod kątem:

- 1) sprawdzenie poprawnej implementacji zakresu adresowego komponentów `axi_reg16_channel`
- 2) sprawdzenie poprawnej implementacji zmodyfikowanego zakresu adresowego komponentu `axi_bram_ctrl_0`
- 3) sprawdzenie poprawnej implementacji zakresu adresowego komponentów `ctrl_reg`, `reg_axi_1`, `reg_axi_meas`, `axi_bram_ctrl`
- 4) konfiguracja i uruchomienie nowo dodanych kanałów w trybie online/software.

1. Sprawdzenie poprawnej implementacji zakresu adresowego komponentów `axi_reg16_channel`
W aktualnej implementacji dodane zostały kanały z zakresu 16-32 co również implikuje dodanie nowych komponentów `axi_reg16_channel` do obsługi.

Wyczytania z zadanego obszaru adresowego 0x4002_0000-0x4002_F000 są poprawne.

```
mrd 0x4002F000 16
```

```
4002F000: 0000003F
4002F004: 00000000
4002F008: 00000000
4002F00C: 00000000
4002F010: 00000000
4002F014: 00000000
4002F018: 00000000
4002F01C: 00000000
4002F020: 00000000
4002F024: 00000000
4002F028: 00000000
4002F02C: 00000000
4002F030: 00000000
4002F034: 00000000
4002F038: 00000000
4002F03C: 00000000
```

2. Sprawdzenie poprawnej implementacji zmodyfikowanego zakresu adresowego komponentu
axi bram ctrl 0

Ze względu na powiększenie liczby rejestrów kanałów została zmodyfikowana przestrzeń adresowa komponentu `axi_bram_ctrl_0` adresu `0x40020000` → `0x40040000`

Wstępny odczyt wykazuje czystą przetrzeń:

```
mrd 0x40040000 16
```

```
40040000: 00000000
40040004: 00000000
40040008: 00000000
4004000C: 00000000
40040010: 00000000
40040014: 00000000
40040018: 00000000
4004001C: 00000000
40040020: 00000000
```

```
40040024: 00000000
40040028: 00000000
4004002C: 00000000
40040030: 00000000
40040034: 00000000
40040038: 00000000
4004003C: 00000000
```

Zapis bram:

```
mwr 0x40040000 0x12345678
```

```
xsct% mrd 0x40040000 16
```

```
40040000: 00000000
40040004: 00000000
40040008: 00000000
4004000C: 00000000
40040010: 00000000
40040014: 00000000
40040018: 00000000
4004001C: 00000000
40040020: 00000000
40040024: 00000000
40040028: 00000000
4004002C: 00000000
40040030: 00000000
40040034: 00000000
40040038: 00000000
4004003C: 00000000
```

Brak możliwości zapisu do bram (jedynie możliwy zapis z poziomu kanału fizycznego).

Zapis do bram danych z max (bez konfiguracji)

```
mwr 0x40000010 0x1
```

```
mrd 0x40000010
```

```
40000010: 30000060
```

Flaga write done na '1', odczyt z bram:

```
mrd 0x40040000 16
```

```
40040000: DDCCFFFE
40040004: EDDFEDEC
40040008: CCCFFCCC
4004000C: CCCCCCF
40040010: FFFFDDDD
40040014: DEEFFFE
40040018: DDDDDEEF
4004001C: FFEEFDDC
40040020: CFFFEEDD
40040024: DCCFFFFF
40040028: DDDCCFFE
4004002C: EEDDEEEF
40040030: FEEECDD
40040034: DCCFFEEE
40040038: CCDDDCCE
4004003C: EEEECCEE
```

Dane zapisane, test poprawny.

3) Sprawdzenie poprawnej implementacji zakresu adresowego komponentów ctrl_reg, reg_axi_1, reg_axi_meas, axi_bram_ctrl

Odczyt rejestru wersji komponentu ctrl_reg

```
mrd 0x40000000 2
```

```
40000000: 00000003
40000004: 000000DE
```

Odczyt wartości Component ver dla reg_axi_1

```
mrd 0x40002000
```

```
40002000: 00000003
```

Odczyt wartości Component ver dla reg_axi_meas

```
mrd 0x40003000
```

```
40003000: 00000004
```

Odczyt danych axi_bram_ctrl1

```
mrd 0x40001000 16
```

```
40001000: 00000000
```

Test poprawny.

4) Konfiguracja i uruchomienie nowo dodanych kanałów w trybie online/software.

Przy zmianie kanałów na nowo dodane widać poprawnie przetwarzanie danych od wejścia rejestru ctrl_reg do wyjścia i&d (sprawdzenie ścieżki przekazywania danych w nowo dodanych komponentach). Wpis w trybie software poprawne.

Konfiguracja dla kanału numer 16, sprawdzenie poprawnego ustawienia startu dla trybu odczytu z max:

```
mrd 0x40020004
```

```
40020004: 00000100
```

Dane wyczytane z rejestru I&D zmieniają się poprawnie:

```
mrd 0x40000040 10
```

```
40000040: 442F04B8
```

```
40000044: 9F18646B
```

```
40000048: 442F0836
```

```
4000004C: 9F186C98
```

```
40000050: 442F0BB7
```

```
40000054: 9F1874BE
```

```
40000058: 442F0F35
```

```
4000005C: 9F187CEB
```

```
40000060: 442F12B6
```

```
40000064: 9F188518
```

Wstępne testy poprawności funkcjonalnej nowo dodanych kanałów wykazały poprawne przekazywanie danych komponentów przetwarzania danych w dodanym kodzie. Wyklucza to błąd implementacji nowego kodu oraz wstępnie sprawdza dodaną funkcjonalność.