PROGRAMOWANIE UKŁADÓW FPGA

PROJEKT: OBSŁUGA SYGNAŁU HDMI PRZY UŻYCIU PŁYTKI NEXYS VIDEO.

Opiekun projektu: dr. inż. Andrzej Wojeński

Wykonujący: Wiktor Szczerek, Filip Kulik, Jacek Dobrowolski

Abstrakt i zakładana funkcjonalność.

Projekt zakładał implementację bramki dla sygnału wideo przesyłanego w protokole HDMI, zgodnie ze standardem TMDS oraz możliwość generowania „ustawionych" obrazów. Dodatkową funkcjonalnością miało być wykrywanie krawędzi na obrazie przesyłanym, w oparciu o algorytm Canny'ego, wykorzystujący filtry Sobela w celu uzyskania w/w funkcjonalności.

Zważywszy na skalę projektu, nie udało się zaprojektować od zera jednego układu, będącego „bramką" wykrywającą krawędzie na obrazie w czasie rzeczywistym oraz pozwalającego na generację wybranych obrazów. Zrealizowano więc dwa projekty:

1. Generator sygnału, połączony z enkoderem TMDS[1] i serializerem – co pozwala na wyświetlanie obrazu na kompatybilnym monitorze (posiadającym złącze HDMI).
2. Oparta na gotowych blokach IP (przygotowanych przez producenta) „bramka", z zaimplementowanym modułem wykrywającym krawędzie (dzięki Vivado HLS).

Projekt wykonywany na sprzęcie własnym, tj. płytce Nexys Video z układem FPGA Xilinx Artix-7 XC7A200T-1SBG484C. Płytka jest wyposażona w potrzebne złącza wejście HDMI i wyjście HDMI.

Standard TMDS – ogólne informacje.

W przesyle danych w standardzie TMDS, w najbardziej bazowej formie, występują cztery łącza różnicowe – trzy na kanały dla wideo+audio oraz jeden na zegar. Audio wysyłane jest jako część informacji dotyczącej pikseli – jest to dość złożona funkcjonalność, która wychodzi poza ramy tego projektu, więc nie została zaimplementowana. Dodatkowo istnieją jeszcze kanały informacyjne – DDC (Display Data Channel – pozwalający na ustalenie dostępnych rozdzielczości i innych informacji. Komunikacja zrealizowana jest za pomocą interfejsu I^2C) oraz CEC (Consumer Electronics Channel – dostarczający informacji o modelu monitora, itp.), a ponadto złącze HPD (Hot Plug Detect – pozwalająca na wykrycie podłączenia urządzenia). Dodatkowe kanały nie zostały użyte w pierwszym projekcie, ze względu na dużą złożoność. Dla samego nadajnika, dane wystawiane „w kółko" nie stanowią problemu, gdyż monitor się do tego dostosuje i spróbuje wyświetlić dane, które otrzymuje – testy wykazały, że efekt jest taki sam jak zamierzony. Obraz wysyłany jest w rozdzielczości 640x480 piskeli (tak naprawdę ramka ma rozmiar 800x525 pikseli – obszar poza obrazem wykorzystywany jest dla synchronizacji pionowej i poziomej oraz innych procesów wykonywanych przez odbiornik), co wymusza użycie sygnału zegarowego TMDS o częstotliwości 25MHz.

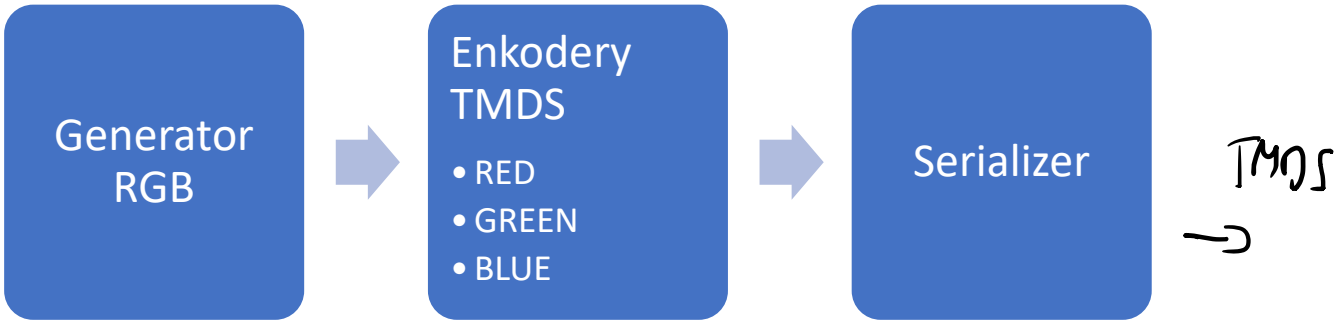
Drugi projekt zawiera pełną komunikację, ze względu na dobry projekt wszystkich bloków IP. Posiada on również konwersję sygnału TMDS na sygnał zgodny ze standardem AXI4-Stream (ARM), będący idealnym połączeniem między blokami operującymi na dużych danych. Pozwala to „wpiąć" filtr wykrywający krawędzie. Gotowe IP-core'y są przystosowane do obsługi wielu rozdzielczości i dostosowywania ustawień do wymagań użytkownika. Rozdzielczość jest zablokowana na 800x600.

Mimo tego, że karta Nexys Video powinna obsłużyć rozdzielczości do 1920x1080 pikseli włącznie, to zdecydowane zostało użycie jej ułamka, ze względu na stabilność działania oraz prostsze wykonanie.

[1] TMDS - Transition Minimized Differential Signalling – standard przesyłu wideo oraz audio w złączach DVI ora HDMI (złącza o dużych przepustowościach). Zapewnia minimalną wrażliwość na interferencję e-m między przewodami oraz na dokładne odtworzenie zegara w odbiorniku.

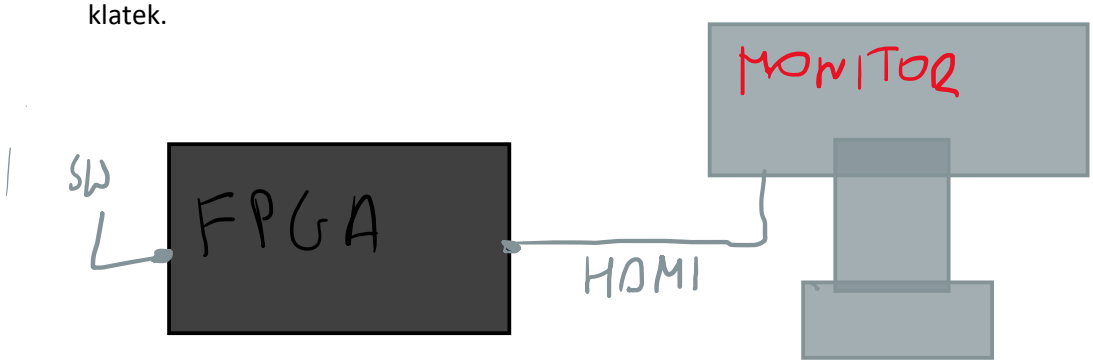
Schemat blokowy projektów.

Projekt 1



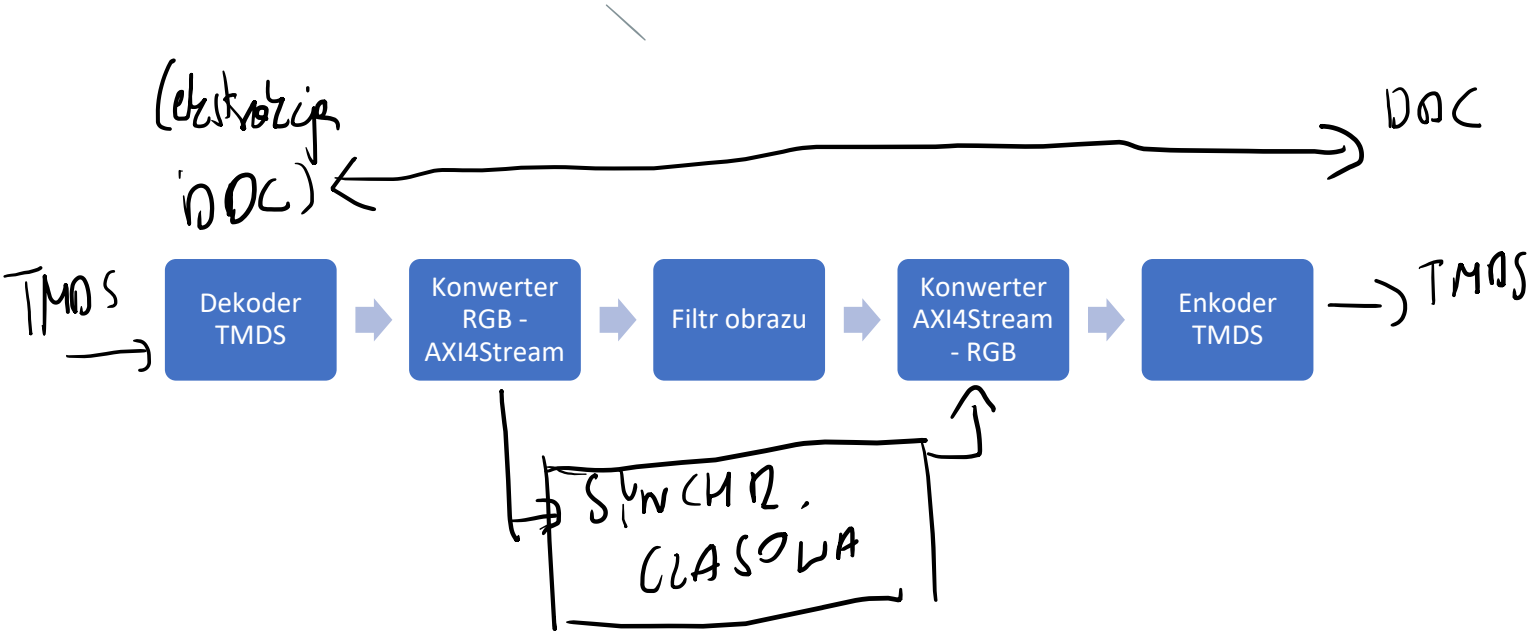
Schemat 1. Wizualizacja blokowa pierwszego projektu.

Pierwszy projekt zakłada komunikację w jedną stronę – wygenerowany obraz jest kodowany według algorytmu zgodnego ze standardem TMDS. Ma to na celu zminimalizowanie błędów w odczycie danych od strony odbiornika. Sam sygnał RGB jest zestawem trzech ścieżek, dla każdego z kanałów podstawowych, o długości 8b każdy – co daje możliwość regulacji wypełnienia koloru w zakresie 0-255 umownych stanów jasności dla każdego piksela (w sumie 24b na każdy piksel). W enkoderze, każdy kanał rozszerzany jest do 10b wg. algorytmu, którego schemat blokowy przedstawiony jest w Dodatku 1. Ostatnim elementem jest serializer – pozwala on na synchroniczne przesyłanie każdego z dziesięciu bitów w jednym okresie zegara TMDS (oznacza to, że bity są przesyłane z częstotliwością 10x większą niż częstotliwość zegara przesyłanego do odbiornika – co jest oczywiste, gdyż każdy okres zegara TMDS daje nam informacje o JEDNYM pikselu). Stąd dane są wystawiane na zewnątrz, skąd odbiera je np. monitor. Switch SW pozwala na wyświetlenie jednej z dwóch predefiniowanych klatek.



Schemat 2. Schemat sprzętowy – podłączenie FPGA-MONITOR.

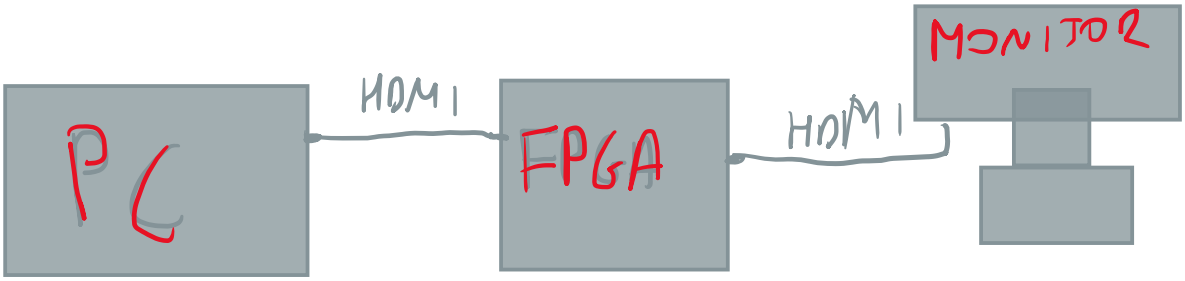
Projekt 2.



Schemat 3. Wizualizacja blokowa drugiego projektu.

Drugi projekt daje możliwość komunikacji „obustronnej", tj. odbiór sygnału ze źródła, obróbkę wewnątrz układu oraz wystawienie sygnału dla odbiornika. Na początku sygnał TMDS od źródła jest dekodowany do postaci sygnału RGB – od razu uzyskiwany jest też sygnał DDC, który jest wysyłany do odbiornika w celu ustawienia parametrów połączenia. Po ustanowieniu połączenia wysyłane są tą drogą wszelkie informacje dotyczące zmian w rozdzielczości, np.. Następnie sygnał RGB jest konwertowany na sygnał zgodny ze standardem AMBA AXI4-Stream[2], w celu łatwiejszej jego modyfikacji i współpracy (inne IP core'y firmy Digilent – w tym naprzykład softcore'owy procesor Microblaze współpracują bardzo chętnie z tym protokołem). Tak obrobiony sygnał odbierany jest przez filtr, napisany przy użyciu Xilinx Vivado HLS (High Level Synthesis) – rozwiązania umożliwiającego projektowanie układów przy użyciu języków programowania wysokiego poziomu (C/C++) – oraz biblioteki HLS OpenCV, dostarczającej API powiązanego z obróbką wideo. Użyte zostały funkcje odpowiadające krokom z dokumentacji wstępnej, będące kolejnymi krokami w algorytmie Canny'ego. Następnie pierwsze dwa procesy są odwracane, żeby uzyskać sygnał TMDS na zewnątrz. W efekcie, uzyskujemy obraz będący wynikiem działania filtru dla całego obrazu wejściowego.

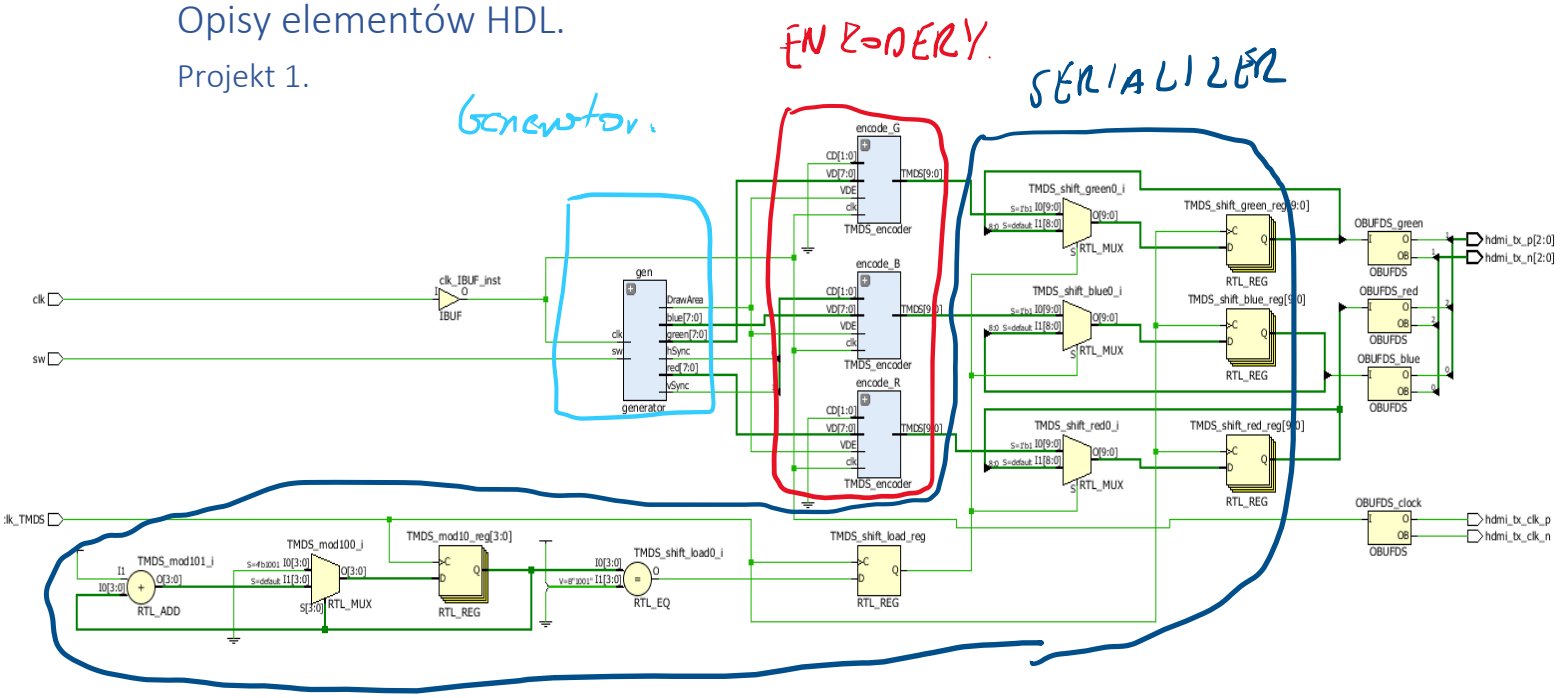
[2] The AXI4-Stream protocol is used as a standard interface to connect components that wish to exchange data. The interface can be used to connect a single master, that generates data, to a single slave, that receives data. The protocol can also be used when connecting larger numbers of master and slave components. The protocol supports multiple data streams using the same set of shared wires, allowing a generic interconnect to be constructed that can perform upsizing, downsizing and routing operations. "AMBA 4 AXI4-Stream Protocol Specification" ARM.



Schemat 4. Schemat sprzętowy – podłączenie PC – FPGA – monitor.

Opisy elementów HDL.

Projekt 1.



Schemat 5. Schemat elaborowany projektu 1.

TOP (rozumiany jako oddzielny IP core. Należy podpiąć go w odpowiednie porty w block design).

Porty:

- wejściowe:
  - clk - wejście zegarowe – zegar TMDS główny (powinien być wpięty w zegar 25MHz).
  - sw – wejście dip-switcha
  - clk_TMDS – wejście zegarowe – zegar TMDS pomocniczy (clk * 10).
- wyjściowe:
  - hdmi_tx_p/hdmi_tx_n – porty wyjściowe HDMI w parze różnicowej (3 takie pary)
  - hdmi_tx_clk_p/hdmi_tx_clk_n – porty wyjściowe zegara HDMI w parze różnicowej (1 taka para).

generator – blok generujący dane RGB. Napisany „na sztywno” dla rozdzielczości 640x480, implementuje dwa countery – jeden dla osi X, drugi dla osi Y (iterowanie po pikselach – również dla obszaru poza obrazem) - oraz przypisuje wartości wskazane prze projektanta na swoje wyjścia. Na sztywno zaimplementowane: standardowe paski RGB poziomo oraz pionowe paski z gradientami poziomymi dla Magenta-Yellow-Cyan.

Porty:

- wejściowe:
  - clk – sygnał zegarowy (odpowiadający zegarowi TMDS – tutaj 25MHz) [clock]

    - sw – wejście switcha, sterującego wyświetlanymi klatkami [1b]
- wyjścia:
    - DrawArea – sygnał sterujący włączeniem/wyłączenim wyświetlacza (wskaźnik czy dane „idą” dla obrazu, czy enkoder powinien przejść w fazę wysyłania komend) [1b]
    - red/green/blue – kanały dla poszczególnych kolorów [8b]
    - hSync/vSync – sygnały wymuszające sygnały synchronizacji [1b]

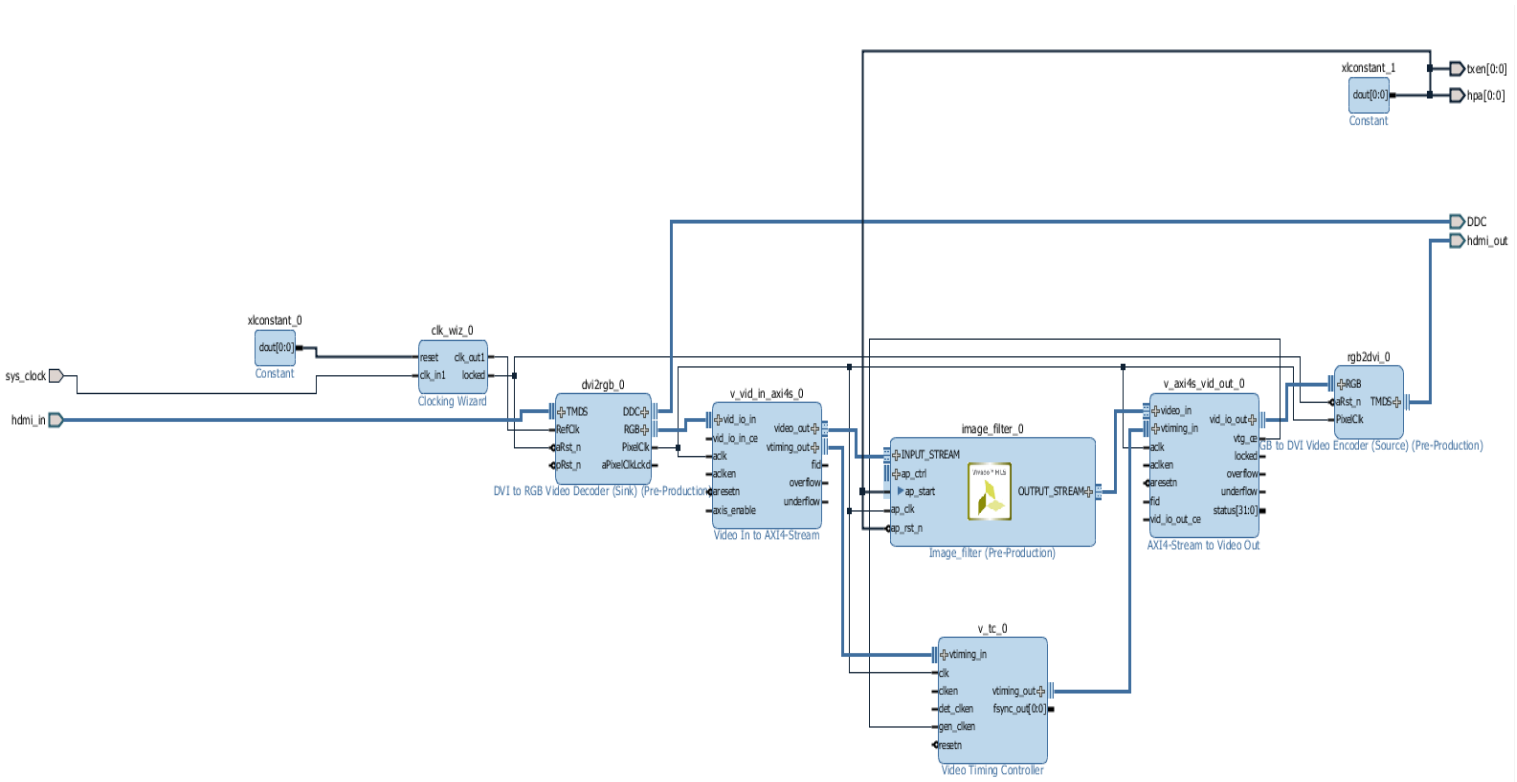
TMDS_encoder – blok kodujący dane w formacie RGB do danych zgodnych ze standardem TMDS. Pozwala na decyzję czy dane są faktycznie dotyczące obrazu, czy są to informacje kontrolne.

Porty:

- wejściowe:
    - clk – sygnał zegarowy (odpowiadający zegarowi TMDS – tutaj 25MHz) [clock]
    - VD – dane wideo (sygnał RGB dla jednego kanału) [8b]
    - CD – dane kontrolne (komendy)[2b]
    - VDE – sygnał informujący czy dane „znalazły się” w obszarze poza obrazem [1b]
- wyjściowe:
    - TMDS – dane wyjściowe w standardzie TMDS [10b]

Serializer – blok który wypuszcza informacje o TMDS w odpowiedniej formie (szeregowo, 10b w ciągu okresu zegaru TMDS). Moduł został zaimplementowany jako część modułu głównego, ze względu na problemy z timingami. W ogólności – odlicza od 0 do 9 i bierze odpowiadające indeksowi bity wektorów dla każdego kanału TMDS, wystawiając je na wyjście układu. Potem następuje przeładowanie na nową wartość i cykl się powtarza.

Projekt 2.



Schemat 6. Schemat blokowy (block design) projektu 2.

W przypadku tego projektu i użycia IP Core'ów od Digilenta, przystępniejszym sposobem projektowania było użycie tzw. Block Designu. Xilinx dostarcza możliwość tworzenia układów, bez napisania nawet linijki kodu w językach VHDL/Verilog – dzięki użycie wyprodukowanych już implementacji. Wynika to ze złożoności tych implementacji i z zasady, że „nie ma sensu wynajdywać koła na nowo". Z tego powodu, nawet prosta stała jest oddzielnym blokiem. Takie podejście pozwala na szybkie utworzenie żądanego układu, znając mechaniki działające w układach FPGA (np. dopasowanie odpowiedniego zegara, dostosowanie plików constraints (wymogów) – czyli zdefiniowania odpowiednich portów, oraz ich standardów elektrycznych.
Opisy bloków zawierają opisy portów UŻYTYCH w projekcie.

TOP:

Porty:

- wejściowe:
  - sys_clk – wejście zegarowe, podpięte pod oscylator wbudowany w płytkę (ustawiony na 100MHz)
  - hdmi_in – **interfejs** HDMI, wpięty w piny odpowiedzialne za odpowiednie linie przesyłowe w porcie wejściowym HDMI na płytce
- wyjściowe:
  - txen oraz hpa – wyjścia ustawiające tryb „sink" układu jako aktywny – ustawione na „1" pozwalają zrozumieć generatorowi (PC), że urządzenie może przyjmować dane. Znaczą tyle co „transmission enable" oraz „hot-plug assert".
  - hdmi_out – **interfejs** HDMI, wpięty w piny odpowiedzialne za odpowiednie linie przesyłowe w porcie wyjściowym HDMI na płytce
  - DDC – **interfejs** DDC, wpięty w piny odowiadające za transmisję I2C w HDMI.

Constant0/1 – blok przechowujący stałą – tutaj 0 albo 1. 0 jest podłączone na stałe do generatora sygnału zegarowego, wymuszając brak resetu.

Porty:

- wyjściowe:
    - dout – wyjście stałej [1b]

Clocking Wizard – blok pozwalający na konfigurację sygnałów zegarowych, generowanych z sygnału zegara głównego (w domyśle 100MHz). Tak naprawdę steruje innymi blokami związanymi z generacją zegarów: MCMM (Mixed Mode Clock Manager) oraz PLL (Phase-Locked Loop). Oba rozwiązania są stworzone do trochę innych celów oraz zależą od topografii zaprojektowanego układu. W naszym przypadku został wybrany tryb MCMM. Liczba wyjść i wejść zegarowych jest modyfikowalna.

Porty:

- wejściowe:
    - reset – wejście resetu (spięte na stałe z logicznym „0")
    - clk_in1 – wejście zegara referencyjnego (w tym przypadku sys_clk)
- wyjściowe:
    - clk_out1 – wyjście zegara wygenerowanego (w tym przypadku jest to 200MHz – potrzebne kolejnym blokom)
    - locked – wyjście wskazujące, że sygnał zegara jest poprawnie wygenerowany – steruje resetem reszty bloków (decyzja wykonawcy)

DVI2RGB – Digilent – blok dostarczony przez producenta. Konwertuje sygnał TMDS na RGB i przesyła odpowiednim **interfejsem**[3] (dane wysyłane są jako 24b, jedną linią; istnieją dodatkowe linie sterujące) do następnych bloków. Na podstawie sygnału zegarowego referencyjnego, regenerowany jest sygnał zegarowy TMDS[4], który służy jako podstawa dla innych bloków. Odzyskiwany jest również sygnał DDC – przesyłany dalej (do odbiornika).

Porty:

- wejściowe:
    - TMDS – wejście **interfejsu** TMDS
    - RefClk – wejście sygnału zegarowego referencyjnego
    - aRst_n – reset asynchroniczny
- wyjściowe:
    - DDC – wyjście **interfejsu** DDC
    - RGB – wyjście **interfesju** RGB – dane do obróbki
    - PixelClk – odzyskany sygnał zegarowy TMDS.

[3] Te odpowiednie interfejsy (oznaczone na schemacie grubą niebieską linią) oznaczają interfejsy zdefiniowane przez producenta – dzięki temu komunikacja między blokami jest ujednolicona. Zaznaczone będą tutaj **pogrubieniem.**

[4] W trakcie propagacji sygnału przez kabel, mogą zdarzyć się przekłamania i zniekształcenia sygnału. Konieczny jest jakiś mechanizm regenaracji informacji zegarowej, żeby sygnał się potem „nie rozjechał".

Video In to AXI4-Stream – Digilent – blok dostarczony przez producenta. Konwertuje sygnał RGB na AXI4-Stream.

Porty:

- wejściowe:
  - vid_io_in – wejście **interfejsu** RGB
  - aclk – wejście zegara (tutaj jest to zegar TMDS)
- wyjściowe:
  - video_out – wyjście **interfejsu** AXI4-Stream, zawierającego oprócz 32b wektora danych, informacje na temat startu/stopu operacji, itp.
  - vtiming_out – wyjście **interfejsu** zegarowego, zawierającego dane dotyczące zegara. Pozwala na synchronizację wrażliwych punktów, np. konwerterów VIDEO-AXI i AXI-VIDEO.

Video Timing Controller – Digilent – blok dostarczony przez producenta. Pozwala na synchronizację czasową krytycznych bloków obsługi wideo.

Porty:

- wejściowe:
  - vtiming_in – wejście **interfejsu** zegarowego, zawierającego dane dotyczące zegara.
  - clk – wejście zegara referencyjnego
  - gen_clken - ?
- wyjściowe:
  - vtiming_out – wyjście **interfejsu** zegarowego, zawierającego dane dotyczące zegara. Pozwala na synchronizację wrażliwych punktów, np. konwerterów VIDEO-AXI i AXI-VIDEO.

AXI4-Stream to Video Out – Digilent – blok dostarczont przez produtcenta. Konwertuje sygnał AXI4-Stream na **interfejs** RGB.

Porty:

- wejściowe:
  - video_in – wejście **interfejsu** AXI4-Stream, zawierającego oprócz 32b wektora danych, informacje na temat startu/stopu operacji, itp.
  - vtiming_in – wjście **interfejsu** zegarowego, zawierającego dane dotyczące zegara. Pozwala na synchronizację wrażliwych punktów, np. konwerterów VIDEO-AXI i AXI-VIDEO.
  - aclk – wejście zegarowe
- wyjściowe:
  - vid_io_out – wyjście **interfejsu** RGB.
  - vtg_ce - ? //prawdopodobnie generuje 1 żeby timing leciał.

RGB2DVI – Digilent – blok dostarczony przez producenta. (Taki sam schemat działania jak DVI2RGB, ale na odwrót. Znaczenie portów identyczne (tylko, że zamiast wyjść- wejścia i na odwrót).

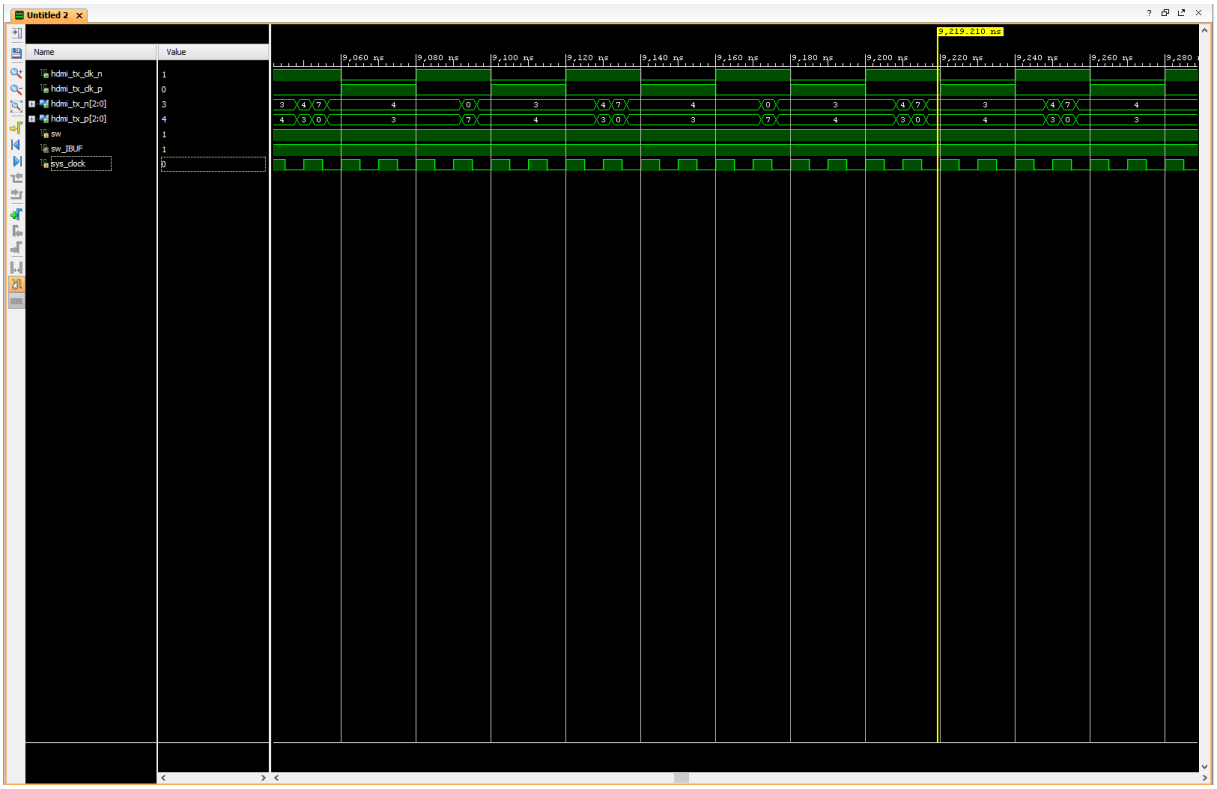
Image Filter – blok zaprojektowany w HLS. Przyjmuje dane zgodne z interfejsem AXI4-Stream, które są konwertowane (po odpowiedniej akwizycji) na format matrycowy, który można „obrobić” w wymagany sposób. Podobnie jak to miało miejsce w dokumentacji wstępnej (projekt obrazujący zachowanie), generowanych jest kilka obrazów (na każdym kroku oddzielny) który jest obrabiany kolejnymi filtrami (gauss,sobel) oraz następuje korekcja współczynników, żeby wyłuskać krawędzie mocne od szumu.

Porty:

- wejściowe:
  - INPUT_STREAM – wejście interfejsu AXI4-Stream
  - ap_ctrl – wejście **interfejsu** sterującego pracą bloku (używany jest jedynie port „start” – zwarty z „1” oraz ap_rst_n – wejście resetujące)
- wyjściowe:
  - OUTPUT_STREAM – wyjście interfejsu AXI4-Stream

Rezultaty.

Ze względu na złożoność interfejsu, najprostszą formą symulacji było wygenerowanie bitstreamu i programowanie płytki, żeby zobaczyć czy działa w przypadku projektu złożonego z bloków IP zamiast żmudnego sprawdzania każdego przebiegu w symulacji funkcjonalnej. W przypadku projektu pierwszego, można było sprawdzić czy generowane są poszczególne sygnały na stopniach (generator,endkoder, wyjście).



Symulacja funkcjonalna projektu 1.

Działanie obu układów widać na dołączonych do archiwum filmach wideo.

Podsumowanie.

Interfejs HDMI jest interfejsem złożonym, wymagającym wzięcie pod uwagę wielu jego składowych, jeżeli planuje się wykorzystać jego pełen potencjał. Istnieje możliwość napisania „od zera” wszystkich bloków w języku opisu sprzętu do sterowania przepływem informacji w standardzie TMDS, konfigurację urządzenia odbiorczego, modyfikację sygnału, jednakże ze względu na powszechnie dostępne bloki IP dostarczone przez producenta – nie ma to wielkiego sensu. Gotowe bloki są dobrze zoptymalizowane, oferują przyjemny interfejs użytkowania i ich użycie jest relatywnie proste do zrozumienia.

Dodatek 1.

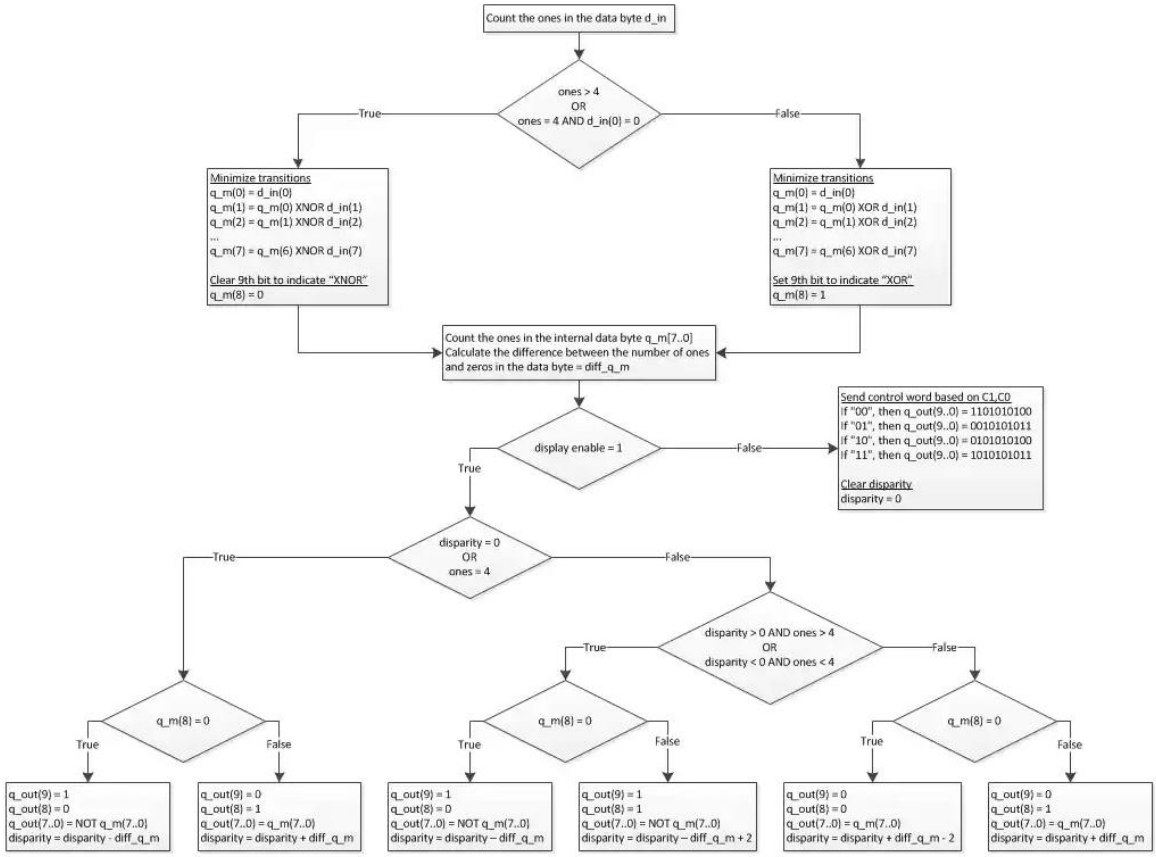


Figure 2. Encoding Algorithm

Algorytm enkodowania sygnału TMDS z sygnału RGB.

źródło: https://www.digikey.com/eewiki/pages/viewpage.action?pageId=36569119