Damian MODRZYK, Dariusz KANIA

Politechnika Śląska, Instytut Elektroniki

Asynchroniczna wymiana danych w układzie GALS ukierunkowana na minimalizację poboru mocy

Streszczenie. *W artykule przedstawiono nowatorską metodę asynchronicznej wymiany danych pomiędzy modułami lokalnie synchronicznymi w systemie GALS. Metoda ukierunkowana jest na minimalizację poboru mocy w złożonych układach kompresji sygnału obrazu oraz sygnału wizyjnego. Rozwiązanie polega na implementacji modułów nadrzędnych dla poszczególnych bloków kodera, które wyposażone są w interfejsy asynchroniczne umożliwiające blokowanie lokalnego sygnału zegarowego. W artykule zaprezentowano wyniki eksperymentalne otrzymane po zastosowaniu przedstawionej metody w układzie kodera Motion JPEG2000.*

Abstract*. In the paper there is presented a novel method of asynchronous data exchange between locally synchronous blocks of a GALS system. The method is oriented toward the minimization of power consumption within complex image or video compression designs. The solution is based on implementation of the asynchronous modules for individual functional blocks of the encoder, which are equipped with asynchronous interfaces and enable gating of a local clock signal. In the article there are presented experimental results obtained after adopting the method in the Motion JPEG2000 encoder design.* ***Asynchronous data exchange between locally synchronous blocks of a GALS system***

Słowa kluczowe: interfejs asynchroniczny, układy globalnie asynchroniczne lokalnie synchroniczne (GALS), bramkowanie zegara
Keywords: asynchronous interface, globally asynchronous locally synchronous (GALS) circuits, clock gating, video compression

doi:10.12915/pe.2014.02.35

Wstęp

Od kilku lat obserwujemy pewnego rodzaju rewolucję w obszarze urządzeń elektronicznych powszechnego użytku. Era komputerów osobistych PC kończy się, a główne zainteresowanie skupia się obecnie wokół multimedialnych urządzeń mobilnych takich jak „smartfony" czy „tablety". Co więcej urządzenia te realizują bardzo złożone obliczenia typu akwizycja, kompresja oraz dekompresja sygnałów obrazu oraz wideo o wysokiej rozdzielczości. Przetwarzanie i transmisja wysokiej jakości sekwencji wizyjnych jest tutaj możliwa dzięki sprzętowym układom kompresji wspierającym najnowocześniejsze standardy kodowania jak H.264/AVC czy JPEG2000 [1].

Wysokie wymagania dotyczące przepustowości obecnych koderów wideo oraz jednostek graficznych GPU (ang. Graphics Processing Unit) wymuszają ich implementację w postaci układów scalonych wielkiej skali integracji VLSI (ang. Very Large Scale Integration). Systemy „on-chip" stosowane obecnie w przemyśle borykają się z dwoma bardzo istotnymi problemami. Pierwszym z nich jest wielkość poboru mocy, który wprost określa czas pracy urządzenia zasilanego bateryjnie. Bardzo rygorystyczne wymagania dotyczące poboru mocy powodują, że w układach synchronicznych za wszelką cenę dąży się do odłączenia sygnału zegarowego w blokach systemu, które są aktualnie niewykorzystywane, lub nawet odcina się napięcie zasilania dla tych modułów [2]. Metody te znane w literaturze jako bramkowanie sygnału zegarowego lub dynamiczne skalowanie napięcia zasilania (ang. dynamic voltage frequency scaling), są skuteczne, lecz często zależne od technologii – ASIC lub FPGA. Drugim istotnym problemem złożonych systemów „on-chip" jest ich synchroniczność. Propagacja jednego sygnału zegarowego do różnych komórek układu kodera wideo, składającego się z kilku milionów bramek logicznych jest zazwyczaj bardzo problematyczna i indukuje zjawisko „skew". Aby skompensować opóźnienia w propagacji sygnału zegarowego stosuje się specjalizowane bufory, które spełniają swoją funkcję, ale jednocześnie zwiększają pobór mocy układu. Innym ograniczeniem płynącym z synchronicznej specyfiki układu jest nieoptymalna szybkość przetwarzania. Jeśli cały układ kompresji złożony z kilku modułów funkcjonalnych synchronizowany jest sygnałem zegarowym o takiej samej częstotliwości, wyznaczonej przez ścieżkę krytyczną najwolniejszego bloku, to osiągana przepustowość kodera nie jest maksymalna. Istotnym problemem synchronicznych układów scalonych, który często jest poruszany w literaturze, jest także emisja zaburzeń elektromagnetycznych EMI (ang. electromagnetic interference), ze względu na generację regularnych impulsów prądowych o dużym natężeniu [3][4].

Wobec istotnych ograniczeń układów synchronicznych szczególnie atrakcyjne są systemy asynchroniczne, które ze względu na brak sygnału zegarowego cechują się znacznie niższym poborem mocy, umożliwiają osiągnięcie większych szybkości przetwarzania danych oraz w mniejszym stopniu emitują zaburzenia elektromagnetyczne. Jednak ze względu na bardzo skomplikowany proces syntezy, złożone mechanizmy wymiany danych oraz brak spójnej metodologii projektowania i testowania, układy w pełni asynchroniczne nie znajdują obecnie szerokiego zastosowania w praktyce inżynierskiej.

Zupełnie nowym podejściem do projektowania złożonych układów scalonych może być realizacja architektury układu w postaci struktury globalnie asynchronicznej lokalnie synchronicznej (GALS). Istota systemu GALS polega na tym, że przetwarzanie danych w poszczególnych blokach synchronizowane jest przez lokalny sygnał zegarowy, natomiast na poziomie globalnym układu wymiana danych następuje w sposób asynchroniczny. Wśród głównych zalet takiej realizacji należy wymienić znacznie prostszą strukturę lokalnych drzew zegarowych czy możliwość dostosowania częstotliwości pracy poszczególnych bloków do ich złożoności obliczeniowej. Istotną cechą układów GALS jest możliwość ograniczenia poboru mocy, wynikająca z faktu, że lokalne sygnały zegarowe poszczególnych modułów mogą być aktywne tylko wtedy, gdy następuje wykonywanie obliczeń. Transmisja danych otrzymanych w wyniku przetwarzania następuje w sposób całkowicie asynchroniczny.

Artykuł prezentuje nowatorskie podejście do realizacji kodera Motion JPEG2000 w postaci struktury GALS. Głównym celem jest zaprezentowanie oryginalnej metody komunikacji bloków lokalnie synchronicznych bazującej na budowie specjalizowanych interfejsów asynchronicznych dla portów wejściowych oraz wyjściowych. Czterofazowy protokół wymiany danych, wykorzystujący sygnały żądania REQ i potwierdzenia ACK zrealizowany jest za pomocą autorskich, asynchronicznych automatów sekwencyjnych AFSM (ang. asynchronous finite state machine). W dalszej części artykułu przedstawione są główne zagadnienia teorii układów GALS, elementy syntezy AFSM oraz oryginalny

mechanizm sterowania lokalnym sygnałem zegarowym, uwzględniający eliminację zjawiska metastabilności przy wyjściu z „trybu uśpienia”. Artykuł podsumowany jest wynikami eksperymentalnymi w formie wykresów czasowych, które otrzymano po zastosowaniu prezentowanej metody w modułach kodera.

Wprowadzenie do układów GALS

Układy globalnie asynchroniczne lokalnie synchroniczne zostały po raz pierwszy scharakteryzowane w rozprawie doktorskiej D. Chapiro [5] w 1984 r. Od tego czasu powstała znacząca liczba prac, które można usystematyzować wg sposobu synchronizacji danych wymienianych pomiędzy blokami lokalnie synchronicznymi, pracującymi w różnych obszarach zegarowych systemu. Bardziej obszerna charakterystyka realizacji GALS została przedstawiona m.in. w [6][7][8]. Jednak w ogóle spotykanych w literaturze prac szczególną uwagę należy skupić na metodach typu „pausible clocking”, rozwiązaniach opartych na asynchronicznych blokach FIFO oraz systemach jawnie wykorzystujących synchronizatory, określanych także jako „boundary synchronization”.

Metoda „pausible clocking” polega na dostosowywaniu lokalnego sygnału zegarowego do asynchronicznych transferów danych na wejściu i wyjściu bloku lokalnie synchronicznego. Chodzi tutaj o zatrzymywanie lub rozciąganie sygnału zegarowego w czasie trwania asynchronicznego protokołu wymiany danych, aby całkowicie wyeliminować lub zminimalizować zjawisko metastabilności. W realizacjach GALS typu „pausible clocking”, które zostały zapoczątkowane pracą Chapiro, sygnały zegarowe dla poszczególnych bloków synchronicznych są najczęściej otrzymywane z lokalnych oscylatorów zbudowanych z nieparzystej liczby inwerterów (ang. ring oscillators) [9]. Jedne z pierwszych metod kontroli układów typu „pausible clocking” zostały przedstawione w pracach Yun, Donohue i Dooply [10][11]. Struktura GALS była tutaj oparta na asynchronicznych modułach FIFO, które tworzyły kanał wymiany danych pomiędzy dwoma blokami w systemie. Komunikacja modułów lokalnie synchronicznych z blokiem FIFO następowała według czterofazowego protokołu, wykorzystującego poziomy sygnałów żądania i potwierdzenia (ang. request-acknowledge handshaking). Naruszenia czasów ustalania i podtrzymania podczas transferu danych zostały zminimalizowane przez zastosowanie specjalizowanych struktur typu „mutex” (ang. mutual exclusion) zapewniających rozciąganie lub zatrzymywanie lokalnego sygnału zegarowego w trakcie aktywnego protokołu. Istotną wadą tej metody był fakt, że tylko jeden port bloku lokalnie synchronicznego mógł być aktywny w jednostce czasu, gdyż układ arbitrażu stawał się bardzo złożony i niepraktyczny przy zwiększaniu liczby obsługiwanych portów wejściowych oraz wyjściowych.

W 1997 r. Borman i Cheung w [12] zaproponowali koncepcję „asynchronous wrappers”, w której każdy blok lokalnie synchroniczny w systemie GALS był otoczony przez układ zapewniający asynchroniczną komunikację z innym modułem lokalnie synchronicznym. Asynchroniczne maszyny stanów, odpowiedzialne za wymianę danych na porcie wejściowym oraz wyjściowym, generowały sygnały , które wstrzymywały lokalny sygnał zegarowy w trakcie transferów. Ważną cechą struktury GALS przedstawionej w [12] było automatyczne przejście bloków lokalnie synchronicznych do trybu oczekiwania z wyłączonym sygnałem zegarowym w przypadku, gdy w module nie było żadnych danych do przetwarzania. Koncepcja Bormana i Cheunga dała początek kilku kolejnym pracom, które były bardzo istotne dla rozwoju systemów GALS. Mowa tutaj m.in. o badaniach Muttersbacha i pozostałych [13][14], które znacznie rozszerzyły zagadnienie „asynchronous wrappers”. W pracach tych podjęto próbę przedstawienia kompletnego schematu projektowania rzeczywistego systemu GALS na przykładzie sprzętowej implementacji algorytmu szyfrowania. Istotnym wkładem w teorię układów globalnie asynchronicznych było wprowadzenie różnego typu asynchronicznych kontrolerów portów wraz z dokładnym opisem AFSM w formie grafów, wg specyfikacji „extended-burst-mode” [15] oraz zapewnienie jednoczesnej obsługi wielu portów bloku lokalnie synchronicznego. W odniesieniu do systemów GALS bazujących na idei „pausible clocking” należy także wspomnieć o pracach naukowców z grupy IHP Microelectronics, którzy w [16] zaproponowali optymalizację sposobu kontroli lokalnie generowanego sygnału zegarowego celem zwiększenia przepustowości w transmisji danych oraz zwiększenia niezawodności całego systemu GALS. Wyniki tych prac są znaczące ze względu na fakt, że zostały sprawdzone w rzeczywistej implementacji procesora FFT, ukierunkowanej na redukcję zaburzeń elektromagnetycznych w „chipie” [4].

Do pozostałych metod realizacji układów globalnie asynchronicznych lokalnie synchronicznych zalicza się stosowanie asynchronicznych bloków FIFO czy wykorzystywanie synchronizatorów dla sygnałów przekraczających tzw. obszary częstotliwościowe (ang. clock domains) [7][8]. Cechą wspólną tych sposobów realizacji systemów GALS jest fakt, że nie naruszają one sygnałów zegarowych, synchronizujących poszczególne moduły w systemie. Pierwsze prace w tym kierunku prowadzone były przez Jakova Seizovica, który przedstawił koncepcję synchronizacji potokowej (ang. pipeline synchronization). Metoda polegała na zapewnieniu komunikacji bloku synchronicznego z otoczeniem asynchronicznym za pomocą szeregu zaprojektowanych w tym celu synchronizatorów [17]. Kolejne istotne prace spotykane w literaturze dotyczą sposobu implementacji modułów FIFO, w których interfejsy zapisu oraz odczytu danych synchronizowane są przez różne sygnały zegarowe. W [18][19] Chelcea i Nowick proponują asynchroniczny układ FIFO zapewniający minimalne opóźnienie przy transferze danych z wejścia na wyjście, nie pogarszając przy tym przepustowości. Ich koncepcja „token rings” oparta jest na pierścieniowej strukturze identycznych bloków, których stan wejściowy i wyjściowy zależny jest od dwóch krążących symboli. Inne koncepcje asynchronicznego modułu pamięci typu FIFO oparte są na wykorzystaniu kodu Gray’a do kodowania liczników zapisu i odczytu danych. W takim przypadku kontrola zajętości modułu FIFO jest realizowana przez tradycyjną, dwu-przerzutnikową synchronizację liczników celem wyznaczenia flag – pusty, pełny [20]. Należy zaznaczyć, że metody bazujące na asynchronicznych blokach FIFO, czy wykorzystujące różnego rodzaju synchronizatory, często kojarzone są z systemami wielozegarowymi (ang. mixed-clock systems), gdzie kanały wymiany danych muszą być wyposażone w sygnały zegarowe bloku nadającego i bloku odbierającego.

Oprócz najistotniejszych prac dotyczących metod realizacji struktur GALS, znaczące są także wyniki porównań tych systemów z układami stricte synchronicznymi przedstawione m.in. w [21][22][23]. W większości przypadków analizy te wskazują na pewne pogorszenie szybkości przetwarzania danych oraz kilkuprocentowy przyrost wielkości układu w strukturze ASIC czy FPGA. Istotną przewagą układów GALS nad realizacjami synchronicznymi jest możliwość ograniczenia poboru mocy w systemie. W tym kierunku prowadzone są badania, których częściowe wyniki zaprezentowano w niniejszym artykule.

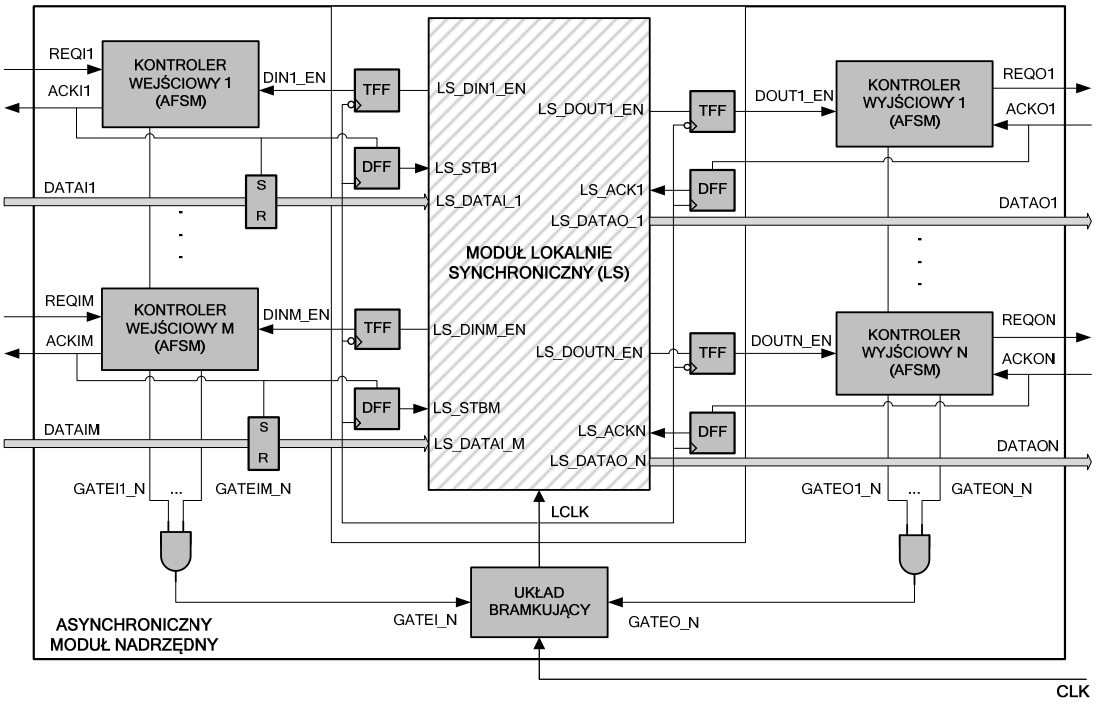
Koncepcja asynchronicznych modułów nadrzędnych

Istota badań sprowadza się do opracowania globalnie asynchronicznej lokalnie synchronicznej architektury kodera Motion JPEG2000, w której dominującą rolę odgrywa oryginalna metoda wymiany danych. Celem jest realizacja kodera, w której moduły funkcjonalne będą synchronizowane za pomocą szeregu sygnałów zegarowych blokowanych w momentach, w których odpowiednie bloki znajdują się z w stanach bezczynności. Dodatkowo, częstotliwość pracy modułów powinna być dopasowana do wymaganej przepustowości całego systemu. Założono, że transfer wyników pośrednich kompresji pomiędzy modułami kodera powinien następować w sposób całkowicie asynchroniczny.

Punktem wyjścia dla autorskiej metody asynchronicznego transferu danych była metoda realizacji struktury GALS zaproponowana przez Amini i pozostałych w [24][25]. Koncepcja ta bazuje na idei zatrzymywania sygnału zegarowego, jednak elementem nowatorskim jest zastosowanie zewnętrznych sygnałów zegarowych zamiast sygnałów otrzymanych z lokalnych generatorów. Takie podejście ma szereg zalet w porównaniu do metody „pausible clocking", w której częstotliwości sygnałów zegarowych są bardzo podatne na wahania temperatury, co sprawia, że konieczny jest skomplikowany proces kalibracji lokalnych generatorów.

Autorską metodę kontroli transferu danych w systemie GALS przedstawiono na Rys.1 w formie asynchronicznego modułu nadrzędnego dla uogólnionego bloku lokalnie synchronicznego (LS). Prezentowany moduł posiada M portów wejściowych oraz N portów wyjściowych. Kiedy blok LS jest gotowy na przyjęcie danych na odpowiednim porcie wejściowym ustawia flagę LS_DIN_EN, natomiast gdy żąda wysłania danych na porcie wyjściowym, wtedy ustawia flagę LS_DOUT_EN. Obydwa sygnały synchronizowane są narastającym zboczem lokalnego sygnału zegarowego LCLK i wprowadzane na przerzutniki typu T (TFF). Te z kolei synchronizowane są opadającym zboczem LCLK i powodują zmianę stanu wyjścia z 0 na 1, lub z 1 na 0, gdy tylko wejście LS_DIN_EN lub LS_DOUT_EN jest w stanie aktywnym. Przerzutnik TFF zastosowano, aby zrealizować sygnalizację zmian wartości sygnałów DIN_EN oraz DOUT_EN, które uaktywniają odpowiednio kontroler portu wejściowego oraz kontroler portu wyjściowego. Asynchroniczna wymiana danych pomiędzy każdymi dwoma modułami kodera następuje wg czterofazowego protokołu, w którym odpowiednie fazy sygnalizowane są poziomami sygnałów żądania i potwierdzenia – REQ = 1, ACK = 1, REQ = 0, ACK = 0. Kontrolery portu wejściowego oraz wyjściowego zaimplementowano w formie asynchronicznych automatów sekwencyjnych i zamknięto w odrębnych modułach. W obydwu kontrolerach uaktywnienie protokołu następuje po wykryciu zmiany sygnału wejściowego, odpowiednio DIN_EN, albo DOUT_EN, co automatycznie powoduje bramkowanie sygnału zegarowego. Aby wyeliminować hazardy na sygnale zegarowym, wyłączenie sygnału LCLK następuje zawsze na nieaktywnym zboczu. Bramkowanie sygnału zegarowego następuje, jeśli tylko jakikolwiek kontroler portu wejściowego wyzeruje wyjście GATEI_N lub dowolny kontroler portu wyjściowego wyzeruje wyjście GATEO_N. Kiedy blok LS jest gotowy na przyjęcie danych, sygnał zegarowy jest bramkowany tak długo, aż na wejściu pojawi się ważny transfer. Takie zachowanie pozwala na minimalizację poboru mocy w koderze, gdyż sygnały zegarowe dla poszczególnych bloków funkcjonalnych są wyłączone, jeśli nie ma żadnych danych do przetwarzania.

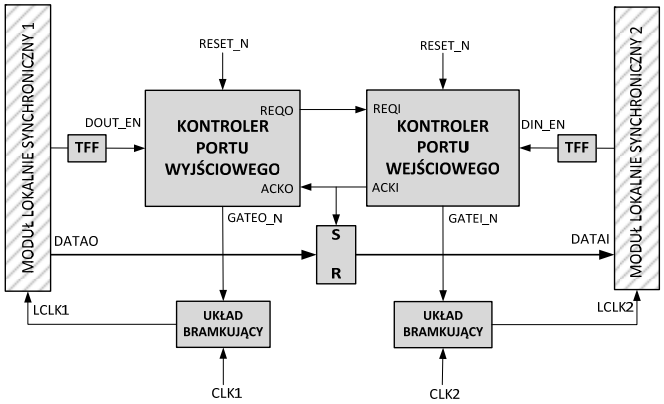
Na Rys.1 widoczny jest również zatrzask RS, w którym następuje zatrzaśnięcie danych wejściowych DATAI, kiedy sygnał ACKI jest w stanie wysokim. Na końcu każdego transferu wejściowego, bądź wyjściowego sygnał zegarowy jest z powrotem włączany, aby umożliwić wykonywanie obliczeń w bloku lokalnie synchronicznym. W celu dopasowania asynchronicznych transferów danych do synchronicznego protokołu stosowanego w modułach kodera wprowadzono pomocnicze sygnały LS_STB oraz LS_ACK, które sygnalizują odpowiednio ważne dane na wyjściu zatrzasku RS oraz potwierdzenie wysłania danych na porcie wyjściowym. Obydwa sygnały ustawiane są asynchronicznie, gdy sygnał ACK jest w stanie aktywnym, natomiast przechodzą w stan 0 na narastającym zboczu LCLK (automatycznie po jego włączeniu).



Rys.1. Schemat blokowy asynchronicznego modułu nadrzędnego

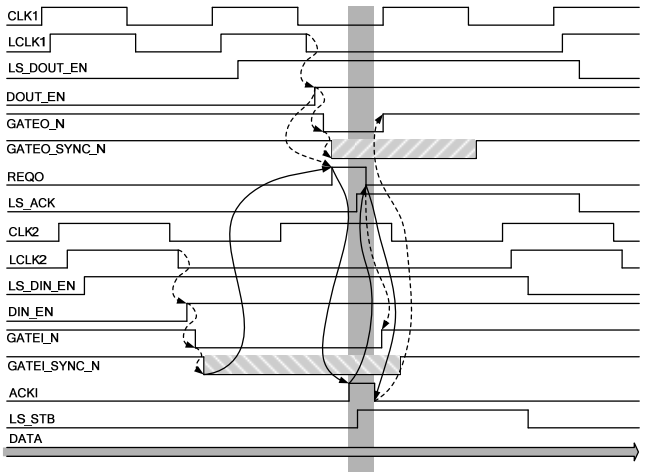
Implementacja kontrolerów portów

Na Rys.2 przedstawiono schemat jednoportowej, asynchronicznej wymiany danych pomiędzy dwoma modułami lokalnie synchronicznymi. Obydwa kontrolery posiadają 3 wejścia i 2 wyjścia oraz są pozbawione sygnału zegarowego. Ustawianie ich sygnałów wyjściowych następuje całkowicie asynchronicznie. Para sygnałów REQ, ACK jest bezpośrednio wykorzystywana do realizacji czterofazowego protokołu wymiany danych, w którym kontroler wyjściowy generuje żądanie REQ, natomiast kontroler wejściowy potwierdza ich odbiór sygnałem ACK.



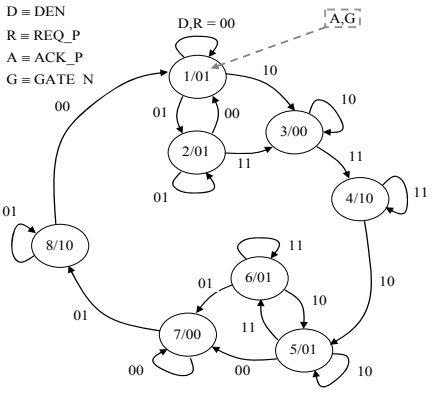
Rys.2. Schemat asynchronicznej komunikacji pomiędzy kontrolerem portu wejściowego, a kontrolerem portu wyjściowego

Poszczególne etapy asynchronicznej wymiany danych z żądaniem inicjowanym poprzez stan 1 sygnału REQO i potwierdzeniem realizowanym przez sygnał ACKI widoczne są na wykresie czasowym pokazanym na Rys.3. Szary, prostokątny obszar na wykresie wskazuje moment, kiedy w porcie wejściowym następuje zatrzaskiwanie danych DATA. Natomiast zakreskowane fazy w przebiegu sygnałów GATEO_SYNC_N i GATEI_SYNC_N określają czas, w którym lokalne sygnały zegarowe, odpowiednio LCLK1 i LCLK2, są bramkowane.



Rys.3. Wykres czasowy asynchronicznej wymiany danych pomiędzy portami

Funkcjonalność kontrolera portu wejściowego odzwierciedla graf automatu przedstawiony na Rys.4. Dla uproszczenia nazwy poszczególnych sygnałów skrócono do pierwszej litery. Mając opis układu wykonano syntezę automatu, rozpoczynając od minimalizacji stanów, która doprowadziła do minimalnej tablicy przejść-wyjść zawierającej cztery stany. Rozpatrywano różne rozwiązania chcąc uzyskać najlepszą strukturę pod względem właściwości dynamicznych.



Rys.4. Graf asynchronicznego automatu sekwencyjnego opisujący kontroler portu wejściowego

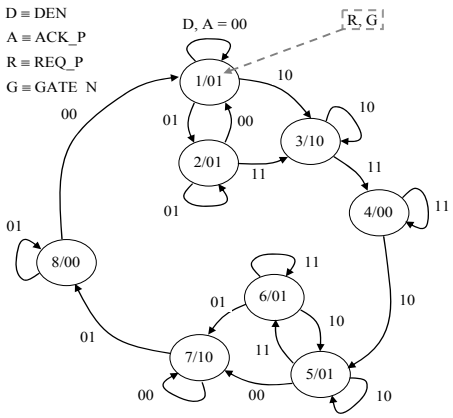
Ostatecznie zdecydowano się na realizację układu w postaci automatu Mealy'ego, kojarząc wszystkie przejścia z krawędziami hipersześcianu, decydując się na cykliczne przejścia występujące na pierwotnym grafie pomiędzy stanami 2-3-4 oraz 6-7-8. Realizując funkcje wzbudzeń wyeliminowano statyczne hazardy strukturalne, a tym samym wyścigi krytyczne. Wynikiem syntezy bloku pamięci są zależności (1).

$$Q_1 = q_1\overline{q_0} + q_1\overline{R} + q_1 D + q_0 D\ R \qquad (1)$$
$$Q_0 = q_1\overline{D} + q_0 R + \overline{q_1} D$$

Realizując blok wyjść szczególną uwagę zwrócono na eliminację hazardów strukturalnych i wypracowanie na wyjściach możliwie najszybszych zmian istotnych w procesie synchronizacji wymiany danych. Uzyskane w wyniku syntezy funkcje wyjść przedstawiają zależności (2).

$$A = q_0 R \qquad G = \overline{\overline{q_1 q_0 \overline{D}}} + q_1\overline{q_0} D \qquad (2)$$

Sposób realizacji funkcji A gwarantuje wystąpienie zmiany na wyjściu A z 1 na 0 przed zmianą sygnału G z 0 na 1, przy przejściach 4→5 oraz 8→1.



Rys.5. Graf asynchronicznego automatu sekwencyjnego opisujący kontroler portu wyjściowego

Kontroler portu wyjściowego jest bliźniaczo podobny do kontrolera portu wejściowego. Graf opisujący jego działanie przedstawiono na rys. 5. Blok pamięci w jednym i drugim przypadku jest identyczny. W wyniku syntezy automatu uzyskano następujące zależności opisujące kombinacyjny blok wyjściowy (3):

$$R = q_1 q_0 \overline{D} + \overline{q_1} q_0 D \qquad G = \overline{\overline{q_1 q_0 \overline{D}}} + q_1\overline{q_0} D \qquad (3)$$

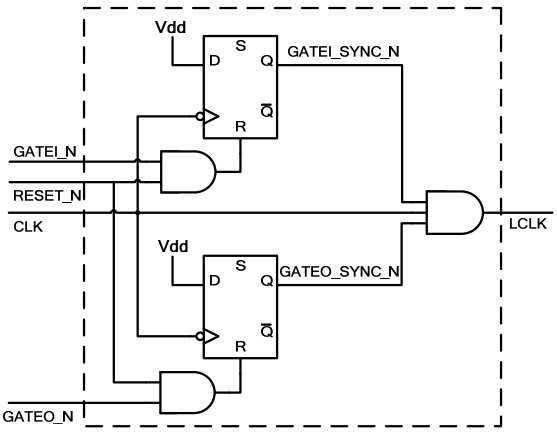
Sposób realizacji bloku wyjść, tzn. odpowiedni rozkład opóźnień, zapewnia wymaganą sekwencję zmian sygnałów R i G. Przedstawione opisy i projekty asynchronicznych kontrolerów portów nie uwzględniały wejścia RESET_N, które w obu przypadkach pełni rolę nadrzędną, wywołując wyzerowanie bloku pamięci Q_0, Q_1, co jest odzwierciedlone w równaniach przedstawionych w (4).

$$q_0 = Q_0 reset\_n \qquad q_1 = Q_1 reset\_n \tag{4}$$

Zaprojektowane automaty sekwencyjne zostały opisane w języku opisu sprzętu VHDL.

Układ bramkowania sygnału zegarowego

Aby wyeliminować hazardy w lokalnym sygnale zegarowym LCLK, w momencie jego włączania po zakończonej wymianie danych, konieczna jest odpowiednia synchronizacja flag GATEI_N oraz GATEO_N. Sposób generowania, czy też blokowania lokalnych sygnałów zegarowych w asynchronicznym module nadrzędnym pokazany jest na Rys.6.



Rys.6. Układ bramkowania sygnału zegarowego

Jeśli przynajmniej jeden z sygnałów GATEI_N lub GATEO_N jest w stanie niskim, co oznacza, że w kontrolerze portu trwa obsługa asynchronicznego transferu danych, to wyjście jednego z przerzutników D (Rys.6) przyjmuje wartość 0. To z kolei wymusza niski stan na wyjściu bramki AND – lokalny sygnał zegarowy LCLK jest blokowany. Natomiast, gdy obydwa sygnały bramkowania GATEI_N i GATEO_N przyjmują stan 1, sygnał LCLK jest odblokowywany dopiero po opadającym zboczu zewnętrznego sygnału zegarowego CLK. Zaproponowane rozwiązanie zapewnia zachowanie odpowiednich czasów ustalania-podtrzymania w bloku lokalnie synchronicznym, ponieważ sygnał LCLK nigdy nie jest ponownie włączany na narastającym zboczu zegara CLK, lub w trakcie, gdy ten sygnał jest w stanie 1. Wykresy czasowe sygnałów bramkujących GATEI_N, GATEO_N oraz flag GATEI_SYNC_N, GATEO_SYNC_N po synchronizacji, wraz z ostatecznym wykresem czasowym sygnału zegarowego LCLK widoczne są na Rys.3. Zastosowana metoda zarządzania lokalnym sygnałem zegarowym zapewnia bezbłędną, asynchroniczną wymianę danych na portach bloku LS oraz wyklucza zjawisko metastabilności przy zatrzaskiwaniu danych wejściowych.

Wyniki eksperymentów

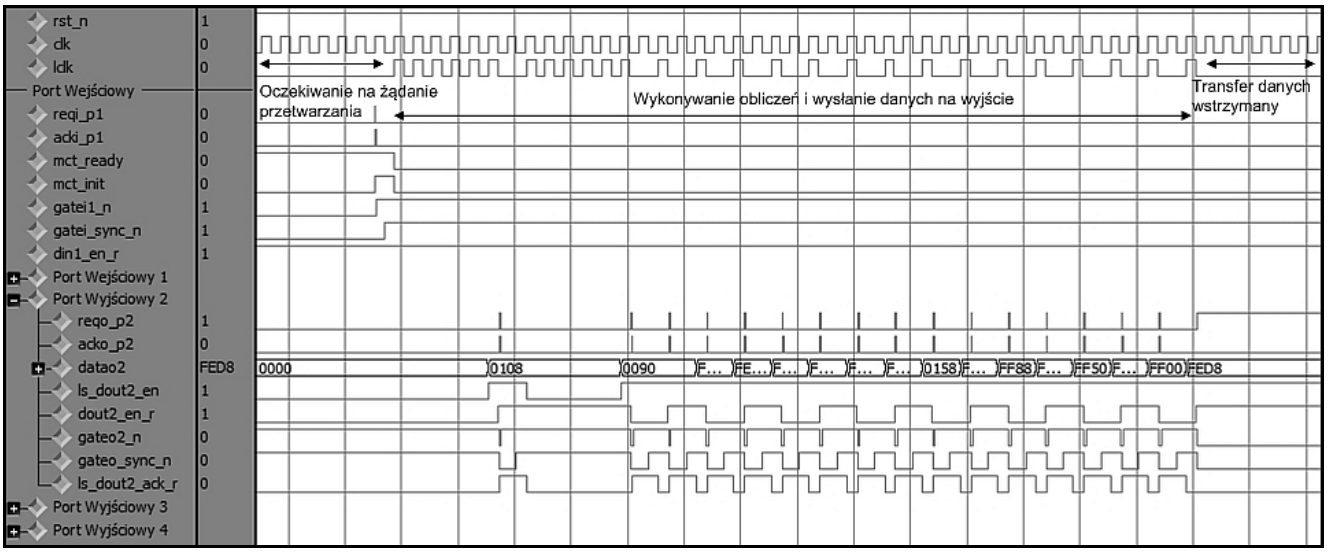
Proponowaną metodę asynchronicznej wymiany danych sprawdzono w układzie sprzętowego kodera Motion JPEG2000 opisanego w języku VHDL. Koder ten składa się z kilku bloków funkcjonalnych, które komunikują się ze sobą przesyłając wyniki pośrednie algorytmu kompresji. Jednym z pierwszych bloków przetwarzania jest moduł transformacji koloru, który realizuje konwersję wejściowych próbek RGB do przestrzeni YC_BC_R. Otrzymane w ten sposób próbki luminancji i chrominancji przesyłane są do bloku zarządzania dostępami do pamięci zewnętrznej. Obydwa moduły synchronizowane są różnymi sygnałami zegarowymi. W nowej wersji kodera dla tych bloków zaimplementowano asynchroniczne moduły nadrzędne, wg koncepcji pokazanej na Rys.1. W celu przybliżenia rzeczywistych warunków pracy całego układu w opisie sprzętowym AFSM wprowadzono opóźnienia bramek AND, OR i NOT charakterystyczne dla elementów technologii ASIC TSMC 40 nm LP.

Na Rys.7. przedstawiony jest wykres czasowy symulacji asynchronicznego modułu transformacji koloru, który komunikuje się z asynchronicznym modułem kontrolera pamięci. Pokazany układ wyposażony jest w jeden asynchroniczny port wejściowy oraz cztery asynchroniczne porty wyjściowe. Na wykresie widoczne są różne fazy przetwarzania danych. Po starcie układu blok transformacji uaktywnia kontroler portu wejściowego sygnałem MCT_READY, co sygnalizuje oczekiwanie na flagę rozpoczęcia obliczeń MCT_INIT. Aktywne żądanie REQI_P1 na porcie wejściowym, potwierdzone wysokim stanem sygnału ACKI_P1 umożliwia rozpoczęcie obliczeń w module transformacji koloru. Na wyjściach 2-4 pojawiają się 16-bitowe próbki luminancji Y i chrominancji C_BC_R, które wysyłane są do pamięci zewnętrznej. Transmisję tych danych kontrolują 3 asynchroniczne automaty sekwencyjne.

Wykresy sygnałów widoczne na Rys.7. potwierdzają prawidłową, całkowicie asynchroniczną wymianę danych pomiędzy zaimplementowanymi kontrolerami portów. W trakcie aktywnych transferów lokalny sygnał zegarowy LCLK jest bramkowany, a następnie poprawnie włączany bez efektu skrócenia jego okresu. Bardzo istotny jest fakt, że w trakcie oczekiwania na rozpoczęcie obliczeń, wstrzymania transferu danych na portach wyjściowych czy zakończenia przetwarzania, lokalny sygnał zegarowy jest wyłączony, co minimalizuje rozpraszanie mocy w koderze.

Podsumowanie

Zaprezentowana w artykule metoda asynchronicznej wymiany danych stanowi rozwinięcie znanych z literatury koncepcji układów GALS. Specyfika realizowanego kodera Motion JPEG2000 wymusiła poszukiwanie architektury układu, w której bloki lokalnie synchroniczne mogłyby się komunikować ze sobą za pomocą asynchronicznego protokołu, stwarzającego możliwość ograniczenia poboru mocy bez istotnego pogorszenia przepustowości kodera. Oryginalność przedstawionej metody wymiany danych polega w szczególności na zastosowaniu autorskich, asynchronicznych automatów sekwencyjnych, które zapewniają wymagane sekwencje zmian sygnałów w protokole asynchronicznym. Lokalne sygnały zegarowe otrzymuje się poprzez bramkowanie zewnętrznych sygnałów zegarowych z zastosowaniem oryginalnej metody synchronizacji sygnałów bramkujących. Zaprezentowane rozwiązanie ukierunkowane jest na implementację układów zarówno w technologiach ASIC jak i FPGA. Obecnie trwają prace eksperymentalne, których celem jest opracowanie ostatecznej architektury kodera, w której możliwe będzie precyzyjne określenie poboru mocy oraz szybkości kompresji osiąganej dla różnych sygnałów wizyjnych.



Rys.7. Wyniki symulacji asynchronicznego modułu transformacji koloru w układzie kodera Motion JPEG2000

LITERATURA

[1] Modrzyk D., Staworko M., A high-performance architecture of JPEG2000 encoder, *Proc. of the 19th European Signal Processing Conf. EUSIPCO*, 2011, 569-573

[2] Modrzyk D., Ograniczenie mocy dynamicznej w architekturze sprzętowego kodera standardu JPEG2000, *Pomiary Automatyka Kontrola,* Vol.56, nr 7, 2010, 793-795

[3] Mocha J., Kania D., Woźnica T., Wykorzystanie przesuniętych w fazie sygnałów zegarowych do redukcji zaburzeń elektromagnetycznych w układach FPGA, *Przegląd Elektrotechniczny*, R. 85 nr 7, 2009, 200-202

[4] Xin F., Krstic M., Wolf Ch., Grass E., A GALS FFT processor with clock modulation for low-EMI applications, *Proc. 21st IEEE Int'l Conf. on Application-specific Systems Architectures and Processors (ASAP),* 2010, 273-278

[5] Chapiro D., Globally-Asynchronous Locally-Synchronous Systems, *PhD thesis, Stanford University*, 1984

[6] Mullins R. and Moore S., Demystifying Data-Driven and Pausible Clocking Schemes, *Proc. 13th IEEE Int. Symp. Asynchronous Circuits and Systems*, 2007, 175-185

[7] Krstic M., Grass E., Gurkaynak F.K., Vivet P., Globally Asynchronous, Locally Synchronous Circuits: Overview and Outlook, *IEEE Design & Test of Computers*, Vol. 24, No. 5, 2007, 430-441

[8] Krstic M., Xin F., Marinkovic M., Gurkaynak F., Heer Ch., Sonntag S., Deliverable – D3, Specification of optimized GALS interfaces and application scenarios, *GALS InterfAce for CompleX Digital System Integration (GALAXY)*, 2008, Issue 2

[9] Taylor G., An on-chip dynamically recalibrated delay line for embedded self-timed systems, *Proc. 6th Int. Symp. on Advanced Research in Asynchronous Circuits and Systems (ASYNC 2000)*, 2000, 45-51

[10] Yun K., Donohue R., Pausible Clocking: A First Step Toward Heterogenous Systems, *Proc. IEEE Int. Conf. Computer Design: VLSI in Computers and Processors*, 1996, 118-127

[11] Yun K., Dooply A., Pausible Clocking-Based Heterogenous Systems*, IEEE Trans. on Very Large Scale Integration (VLSI) Systems*, Vol. 4, n.5, 1999, 482-488

[12] Bormann D., Cheung P., Asynchronous Wrapper for Heterogeneous Systems, *Proc. Int. Conf. on Computer Design (ICCD)*, 1997, 307-314

[13] Muttersbach J., Villiger T., Kaeslin H., Felber N., Fichtner W., Globally-Asynchronous Locally-Synchronous Architectures to Simplify the Design of On-Chip Systems, *Proc. of the 12th IEEE Int. ASIC/SoC Conference*, 1999, 317-321

[14] Muttersbach J., Villiger T., Fichtner W., Practical Design of Globally-Asynchronous Locally-Synchronous Systems, *Proc. 6th Int'l Symp. Advanced Research in Asynchronous Circuits and Systems (ASYNC 00)*, 2000, 52-59

[15] Yun K.Y, Dill D.L, Automatic synthesis of extended burst-mode circuits: Part I and II, *IEEE Trans. on Computer-Aided Design of Integrated Circuits and Systems*, 1999, Issue 2, 101-132

[16] Xin F., Krstic M., Grass E., Analysis and optimization of pausible clocking based GALS design, *Proc. 26th IEEE Int'l Conf. on Computer Design (ICCD)*, 2009, 358-365

[17] Seizovic J.N., Pipeline Synchronization, *Proc. Int'l Symp. on Advanced Research in Asynchronous Circuits and Systems*, 1994, 87-96

[18] Chelcea T., Nowick S.M., Low-latency asynchronous FIFO's using token rings, *Proc. 6th Int'l Symp. on Advanced Research in Asynchronous Circuits and Systems,* 2000, 210-220

[19] Chelcea T., Nowick S.M., Low-latency FIFO for mixed-clock systems, *Proc. IEEE Computer Society Workshop on VLSI,* 2000, 119-126

[20] Beigne E., Vivet P., Design of on-chip and off-chip interfaces for a GALS NoC architecture, *Proc. 12th IEEE Int'l Symp. on Asynchronous Circuits and Systems,* 2006, 172-181

[21] Iyer A., Marculescu D., Power and Performance Evaluation of Globally Asynchronous Locally Synchronous Processors, *29th Annual Int'l Symp. on Computer Architecture*, 2002, 158-168

[22] Hemani A., Meincke T., Kumar S., Postula A., Olsson T., Nilsson P., Oberg J., Ellervee P., Lundqvist D., Lowering power consumption in clock by using Globally Asynchronous Locally Synchronous design style, *Proc. of 36th Design Automation Conference,* 1999, 873-878

[23] Kulmala A., Hamalainen T.D., Hannikainen M., Comparison of GALS and Synchronous Architectures with MPEG-4 Video Encoder on Multiprocessor System-on-Chip FPGA, *Proc. 9th EUROMICRO Conference on Digital System Design: Architectures, Methods and Tools,* 2006, 83-88

[24] Amini E., Najibi M., Pedram H., A Novel Clock Generation Scheme for Globally Asynchronous Locally Synchronous Systems: An FPGA-Validated Approach, *Proc. 15th Great Lakes Symposium on VLSI,* 2005, 296-301

[25] Amini E., Najibi M., Pedram H., Globally Asynchronous Locally Synchronous Wrapper Circuit Based on Clock Gating, *Proc. IEEE Computer Society Annual Symp. on Emerging VLSI Technologies and Architectures,* 2006, 193-199

Autorzy:
mgr inż. Damian Modrzyk, Instytut Elektroniki, ul. Akademicka 16, 44-100 Gliwice, E-mail:dmodrz@wp.pl
dr hab. inż. Dariusz Kania, prof. Pol. Śl., Instytut Elektroniki, ul. Akademicka 16, 44-100 Gliwice, E-mail: dkania@polsl.pl

mgr inż. Damian Modrzyk jest stypendystą projektu DoktoRIS – Program stypendialny na rzecz innowacyjnego Śląska współfinansowanego przez Unię Europejską w ramach Europejskiego Funduszu Społecznego