Adam MILIK, Jan MOCHA
POLITECHNIKA ŚLĄSKA, INSTYTUT ELEKTRONIKI

Samorekonfigurowalny system cyfrowy

Dr inż. Adam MILIK

Ukończył studia na Wydziale Automatyki, Elektroniki i Informatyki Politechniki Śląskiej. Pracę doktorską obronił w 2003 r. Jest adiunktem w Instytucie Elektroniki Politechniki Śląskiej. Jego zainteresowania naukowe to układy logiki programowalnej, sterowniki programowalne, modelowanie i synteza złożonych układów sprzętowo-programowych.

e-mail: adam.milik@polsl.pl

Mgr inż. Jan MOCHA

Ukończył studia na Wydziale Automatyki, Elektroniki i Informatyki Politechniki Śląskiej w 2007 roku. Jest doktorantem w Instytucie Elektroniki Politechniki Śląskiej. Jego zainteresowania naukowe koncentrują się wokół układów logiki programowalnej.

e-mail: jan.mocha@polsl.pl

Streszczenie

W artykule przedstawiono propozycję sprzętowej platformy samorekonfigurowalnej, implementowanej w układzie FPGA. Aby ułatwić zarządzanie konfiguracjami, został zaprojektowany niewielki rdzeń układu, pozwalający na szybką podmianę fragmentu konfiguracji układu. W celu ułatwienia procesu projektowania układów samorekonfigurowalnych, zaproponowano narzędzie przeznaczone do tworzenia projektu oraz generacji szkieletu modułów, jak i skryptów do przetwarzania wsadowego projektu.

Słowa kluczowe: PLD, FPGA, dynamiczna rekonfiguracja, układy kontekstowe, synteza logiczna, dekompozycja.

Self Reconfigurable Digital System

Abstract

The paper propose the selfreconfigurable hardware platform implemented in an FPGA (Spar-tan II/ Spartan 3). The key factor of the design is hardware configuration manager. This is carefully designed small hardware core that manages system configuration. Based on request and configuration registration table it finds partial configuration bit stream start address in external memory and transfers it through SelectMAP interface. In the same it asserts internal BUSY signal until reconfiguration is completed and newly created circuit is properly initialized. There is also presented wizard for partial reconfiguration design flow. It allow to create design skeleton from signal definitions and their assignments between static and dynamic part of the design. Wizard automatically inserts configuration manager core. All those improvements allow to concentrate on implementing functionality instead of taking care of design processing details.

Keywords: PLD, FPGA, dynamic reconfiguration, logic synthesis, decomposition.

1. Wprowadzenie

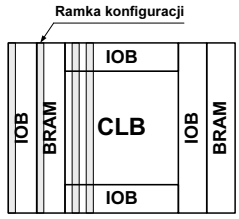
Układy FPGA uzyskały ugruntowaną pozycję w grupie programowalnych układów cyfrowych. Pojemność logiczna tych układów sięga kilku milionów bramek przeliczeniowych, a szybkość przełączania elementów sekwencyjnych osiągają częstotliwości rzędu 500MHz. Najistotniejszą cechą tych układów jest zdolność programowania. W zależności od wykorzystanej technologii sterowania wewnętrznymi zasobami logicznymi, liczba przeprogramowań układu w czasie życia może być: jednokrotna (Actel, antifuse), ograniczona (EEPROM) lub praktycznie nieograniczona (pamięć statyczna RAM). Każda z tych technologii kształtuje także własności układów po włączeniu zasilania. Układy o nieulotnej konfiguracji są zdolne do natychmiastowego rozpoczęcia działania. Układy, których konfiguracja przechowywana jest w pamięciach ulotnych, wymagają przed rozpoczęciem pracy wprowadzenia konfiguracji. Konieczność konfigurowania układu można wskazać jako pewną wadę, bowiem układ nabywa właściwej funkcjonalności, po upływie pewnego czasu od momentu ustabilizowania się napięć zasilających. Mimo wymienionej wady, rozwiązania wykorzystujące pamięci ulotne do przechowywania konfiguracji wykazują wiele zalet w stosunku do układów z nielotną pamięcią konfiguracji. Istotnym aspektem wyróżniającym to rozwiązanie, jest potencjalna możliwość nieograniczonego przeprogramowania układu w czasie jego funkcjonowania. Otwiera to możliwość modyfikacji układu w czasie działania.

Wiele artykułów prezentuje zagadnienia związane z projektowaniem układu dynamicznie zmiennego [1, 2]. Rozwiązania zawierające mikroprocesor implementowany w zasobach układu FPGA, wydają się być nadmiarowe. Rozwiązania sprzętowe realizujące w sposób bezpośredni zadania sterowania, posiadają najlepsze parametry wydajności obliczeniowej i czasu reakcji na zdarzenie zewnętrzne. Podobnie w przypadku układów o dynamicznie zmiennej architekturze gdzie istotnym parametrem jest czas modyfikacji układu. Powierzając powyższe zadanie układowi sprzętowemu, proces reprogramowania będzie odbywał się z możliwie największą prędkością – uzależnioną jedynie od parametrów użytych elementów. Złożoność układu nadzorującego proces reprogramowania, jest dość niewielka w stosunku do złożoności sprzętowej wbudowanego mikroprocesora oraz dodatkowych niezbędnych zasobów sprzętowych związanych z przyłączeniem się do interfejsu konfiguracyjnego.

W dalszej części artykułu przedstawione zostaną wybrane zagadnienia architektury układów FPGA oraz projektowania urządzeń związanych z tworzeniem projektów o dynamicznie zmiennej konfiguracji. Przedstawione zostaną opracowane metody oraz doświadczenia w tej dziedzinie. W rozwiązaniach układowych przedstawiony został układ zarządzania konfiguracjami. Jest to specjalizowany rdzeń układowy włączany w projekt układu dynamicznie zmiennego, który zapewnia sterowanie procesem przeprogramowania układu w czasie jego pracy.

1.1. Podstawy i ograniczenia dynamicznej rekonfiguracji

W pierwszych konstrukcjach układów FPGA proces konfiguracji był niepodzielną procedurą inicjalizującą wewnętrzną pamięć konfiguracji, a następnie wczytującą w sposób aktywny lub pasywny konfigurację. W drugiej połowie lat 90 ubiegłego wieku powstały pierwsze konstrukcje układów, umożliwiające swobodny dostęp do pamięci konfiguracji. Była to seria układów oznaczona symbolem XC6200. Z bliżej niewyjaśnionych przyczyn koncepcja architektury o swobodnym dostępie do pamięci konfiguracji – zaproponowana w przedstawionym rozwiązaniu – została bardzo szybko zarzucona. Obok wymienionych układów powstała również rodzina układów Virtex. Została ona wyposażona w całkowicie nowy system konfiguracji, który zupełnie odróżniał się od wcześniej używanego. Zasadniczą różnicą w systemie konfiguracji było podzielenie jej na ramki, które są interpretowane przez wewnętrzny układ sterowania konfiguracją [3, 5]. Podejście oparte o ramki pozwala na selektywną podmianę fragmentów konfiguracji układu. Najmniejszym wymiennym elementem jest pojedyncza ramka, która oddziałuje na „pionowe" fragmenty układu programowalnego. Przykładowe rozmieszczenie ramek w przestrzeni układu pokazano na rysunku (rys. 1).

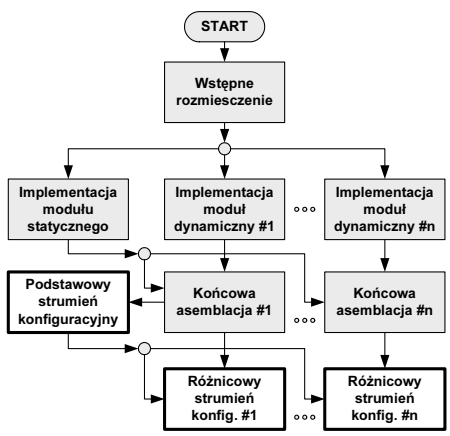


Rys. 1. Konfiguracja oparta na ramkach
Fig. 1. Frame based configuration

Sposób oddziaływania ramek na fragmenty układu oraz połączenia pomiędzy nimi, będzie determinował możliwość użycia niektórych elementów, ze względu na możliwość utraty kontroli nad nimi w czasie podmiany konfiguracji. W dalszej części będą prowadzone rozważania dotyczące dynamicznej rekonfiguracji w oparciu o układy rodzin Spartan II oraz Spartan 3 firmy Xilinx. Układy te mogą podlegać rekonfiguracji w czasie pracy bez wpływu na działanie części układu – część statyczna. W procesie tym należy zwrócić szczególną uwagę na usunięcie ramek komendowych (odpowiedzialnych za uruchomienie sekwencji startowej układu) oraz sterujących globalnymi zasobami logicznymi (sygnał zegarowy). Pozostawienie ramek rozruchowych, spowoduje niepożądaną inicjalizację elementów sekwencyjnych, co spowoduje utratę stanu układu.

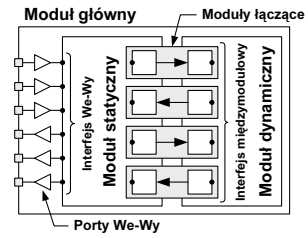
2. Projektowanie układów z dynamicznie zmiennymi elementami sprzętowymi

Ogólny schemat blokowy procesu projektowego przedstawiono na rysunku (rys. 2). W odróżnieniu od projektów jednomodułowych metodyka projektowania jest ściśle związana z rozmieszczeniem elementów w przestrzeni układu. Podmieniany fragment układu musi zostać włączony w już istniejącą strukturę. Narzuca to ograniczenia co do przestrzeni układowej jak i węzłów sygnałów interfejsowych. Narzędzia wspomagające projektowanie i implementację są optymalizowane głównie do użycia z projektami o charakterze statycznym.



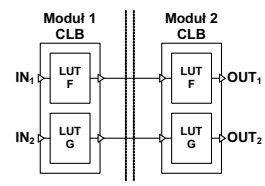
Rys. 2. Proces projektowy układu z dynamicznie wymiennymi modułami
Fig. 2. The design flow of circuit with dynamically exchangeable modules

W przypadku systemów rekonfigurowalnych projektowanie rozpoczyna się od wstępnego rozmieszczenia poszczególnych jego elementów w układzie programowalnym. Wymaga to specyfikacji moduł głównego, w skład którego będą wchodziły: moduł statyczny, moduły dynamiczne oraz moduły łączące część statyczną i dynamiczną (rys. 3). Na wstępnym etapie projektowania nie jest konieczne określanie wnętrza modułu dynamicznego i statycznego. W fazie wstępnej wystarczającym jest opisanie interfejsu wyżej wymienionych modułów. W module głównym specyfikowane są również porty wejścia-wyjścia. Szczególną uwagę należy zwrócić na porty wyjściowe, które są sterowane z wyjścia modułu dynamicznego. Należy zapewnić aby posiadały one wyjście rejestrowe, którego stan nie będzie aktualizowany w czasie zmiany modułu dynamicznego. Moduły dynamiczne będą łączyły się zawsze z tą samą częścią modułu statycznego. Interfejs łączący moduł statyczny i moduły dynamiczne jest nadzbiorem wszystkich sygnałów wykorzystywanych przez różne warianty modułów dynamicznych. Obok specyfikacji sygnałowej moduły dynamiczne posiadają z góry określoną przestrzeń układu, w której mogą się znajdować. Aby było możliwe prowadzenie procesu implementacji, konieczne jest zdefiniowanie punktów charakterystycznych, do których są przyłączane sygnały w układzie fizycznym. Punkty charakterystyczne muszą znajdować się w obrębie obszaru dopuszczonego do prowadzenia procesu implementacji oraz posiadać istniejącą strukturę połączeń przed rozpoczęciem implementacji modułu dynamicznego. Jest to niezwykle istotne zagadnienie warunkujące właściwe rozmieszczenie i połączenie elementów modułu dynamicznego, prowadzące do powstania poprawnie funkcjonujących implementacji dynamicznie wymiennych.



Rys. 3. Rozmieszczenie modułów w układzie FPGA
Fig. 3. Module placement in an FPGA

Biorąc pod uwagę powyższe ograniczenia, moduły łączące części statyczną i dynamiczną, a stanowiące faktycznie połączenie jedno lub dwukierunkowe, muszą powstać przed rozpoczęciem fazy wstępnego rozmieszczania – jako gotowe zaimplementowane w układzie elementy biblioteczne makr sprzętowych (hard macro). Koncepcja przedstawiona w [4] została zmodyfikowana do postaci nadającej się do wykorzystania w rodzinie Spartan II i Spartan 3, przez wykorzystanie generatorów LUT (rys. 4). Bardzo wygodnym narzędziem do operowania modułami jest język XDL . Jest on dość słabo udokumentowany, jednak daje możliwości pełnej kontroli oraz wprowadzania modyfikacji w celu osiągnięcia pożądanych własności przez moduł.



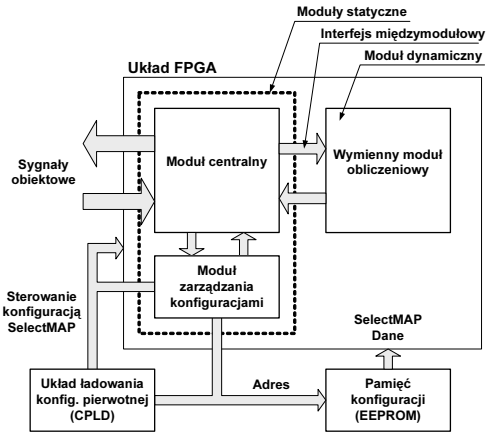
Rys. 4. Makro sprzętowe (NMC) modułu łączącego
Fig. 4. Link module hard macro

3. Układ samorekonfigurowalny

Na rysunku (rys. 5) przedstawiono ogólną architekturę blokową układu o zdolności samorekonfiguracji. Z punktu widzenia użytkownika układ składa się z dwóch modułów. Statycznego modułu centralnego, który koordynuje proces obliczeniowy realizowany w systemie oraz modułu dynamicznie wymiennego, w którym są realizowane obliczenia.

Aby układ mógł rozpocząć pracę należy skonfigurować go konfiguracją rozruchową. Jest to pełny strumień konfiguracji programujący wszystkie elementy układowe znajdujące się w stanie początkowym. Przesłanie konfiguracji pierwotnej, pozwala na uruchomienie części statycznej układu. Konfiguracja oraz dynamiczna rekonfiguracja w układach rodziny Xilinx jest możliwa z wykorzystaniem trybu równoległego podrzędnego SelectMAP [3, 5]. W trybie tym konfiguracja jest przesyłana w słowach 8-bitowych. Przepustowość interfejsu sięga 60MB/s. Niestety prędkość ta jest najczęściej ograniczona przez czas dostępu do

pamięci EEPROM przechowującej konfigurację, której czas dostępu będzie wyznaczał maksymalną prędkość przesyłania danych.



Rys. 5. Architektura układu samorekonfigurowalnego
Fig. 5. Architecture of self reconfigurable circuit

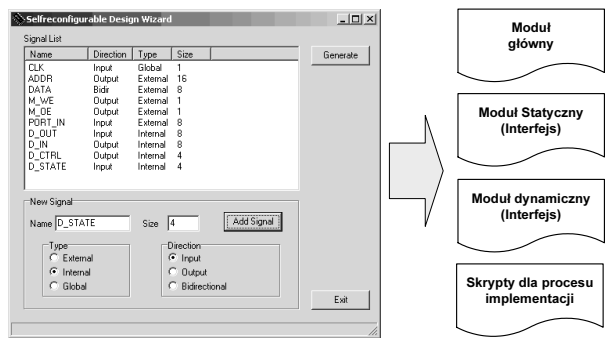
W celu pobrania konfiguracji pierwotnej, najlepiej wykorzystać rozwiązanie bazujące na niewielkim układzie CPLD oraz równoległej pamięci EEPROM (istotna jest możliwość podmiany zawartości). W układzie ładującym konfigurację jest zaimplementowany licznik, który po osiągnięciu przez układ gotowości do konfiguracji (INIT=0), przesyła bajt po bajcie zawartość pamięci aż do uaktywnienia sygnału oznaczającego poprawne skonfigurowanie układu (DONE=1).

Od tego momentu w układzie FPGA znajdują się niezbędne elementy do zarządzania konfiguracją części dynamicznej. Przedstawioną sytuację, można porównać do ładowania jądra systemu operacyjnego. W zależności od potrzeby prowadzonych obliczeń, moduł centralny kieruje zlecenie wprowadzenia właściwej konfiguracji do części wymiennej. Zlecenie jest kierowane do modułu zarządzania konfiguracją. Na podstawie zlecenia w postaci numeru identyfikacyjnego konfiguracji określany jest adres bazowy oraz długość strumienia wybranej konfiguracji. W celu bezpiecznej podmiany konfiguracji, zaproponowano umieszczenie w początkowym obszarze pamięci konfiguracji dynamicznych tablicy, która zawiera deskryptory konfiguracji. Deskryptor konfiguracji składa się z dwóch pól. Adresu początkowego strumienia oraz długości strumienia. W celu łatwego wyznaczania adresów deskryptorów, rozmiar deskryptora równy jest 8 bajtów. Pozwala to przeznaczyć po 4 bajty na adres i długość strumienia konfiguracyjnego. Tablica deskryptorów pozwala również zabezpieczyć się przed wystąpieniem błędów w działaniu układu. Jeżeli w deskryptorze zostanie wykryty adres oraz długość składająca się z samych 1 lub 0, układ zaprzestanie wczytywania konfiguracji. Powyższe kombinacje liczbowe oznaczają odpowiednio, skasowaną komórkę pamięci EEPROM lub zapisaną wielokrotnie. W wypadku wykrycia takiej sytuacji, uaktywniona zostaje linia sygnalizująca błąd realizacji żądania. Dalsza obsługa sytuacji wyjątkowej przekazywana jest do modułu statycznego.

Układ zarządzania konfiguracjami zaprojektowano w wersji pozwalającej na obsługę pamięci o pojemności do 16MB i zawierającej do 16 konfiguracji. Złożoność logiczna układu wynosi 67 komórek slice. W przypadku układu Spartan II 2S200 układ zajmuje 2.7% (67/2352) ogólnych zasobów logicznych. Maksymalna częstotliwość pracy wynosi 100MHz (T_{CLK} = 9.8ns). W przedstawionej wersji, układ wymaga 34 wyprowadzeń na potrzeby komunikacji z zewnętrzną pamięcią zawierającą konfiguracje oraz w celu sterowania interfejsem SelectMAP.

3.1. Wspomaganie procesu implementacji układów samorekonfigurowalnych

Przedstawiony proces implementacji jest złożony i wymaga wykonania wielu czynności w ściśle określonej kolejności. W celu ułatwienia projektowania układów samoreknifgurowalnych, zaproponowano narzędzie do tworzenia projektu w oparciu o definicje użytkownika. Za pomocą narzędzia graficznego, użytkownik definiuje zestaw sygnałów oraz deklaruje ich przynależność. Sygnały mogą być zdeklarowane jako zewnętrzne lub międzymodułowe. Na podstawie zadeklarowanego opisu sygnałów, można rozpocząć generację zasadniczego modułu: głównego, w którego skład wchodzi moduł statyczny oraz moduły dynamiczne (rys. 6). W procesie generacji w module głównym zostają zainstancjonowane zgodnie z opisem moduł statyczny i dynamiczny wraz z odpowiednimi modułami łączącymi oraz układ zarządzania konfiguracjami. Zostają wygenerowane pliki modułów statycznego i dynamicznego wraz z deklaracją interfejsu. Interfejs modułu statycznego zostaje wzbogacony o interfejs do układu zarządzania konfiguracjami. Dodatkowo wygenerowane zostają skrypty, do przetwarzania wsadowego projektu. Po zakończeniu operacji użytkownikowi pozostaje jedynie wypełnienie treścią funkcjonalną obu modułów.



Rys. 6. Narzędzie wspomagające przygotowanie projektu układu samorekonfigurowalnego.
Fig. 6. Selfregoncfigurable circuit design preparation wizard

4. Podsumowanie

W artykule przedstawiono kompleksowe rozwiązanie projektowania układów samorekonfigurowalnych. Optymalizowany rdzeń menadżera konfiguracji zapewnia bezpieczne sterowanie procesem reprogramowania, a przy tym wymaga bardzo niewielu zasobów logicznych układu FPGA. Jak wykazano – niewielki nadmiar sprzętowy, pozwala na uzyskanie dynamicznej wymiany fragmentów układu. Zaproponowano również proste narzędzie, które wspomaga projektanta w tworzeniu projektu układu samorekonfigurowalnego. Zadaniem narzędzia jest wyeliminowanie uciążliwych działań na początkowym etapie tworzenia projektu.

Przedstawione zagadnienia pozwolą na prowadzenie dalszych prac w zakresie układów o dynamicznie wymiennych rdzeniach w połączeniu z syntezą wysokiego poziomu dla układów sterowania i implementacji algorytmów obliczeniowych.

5. Literatura

[1] C. Claus, B. Hang, M. Hubner C. Schmutzler J. Becker W. Stechele "An XDL-based busmacro generator for customizable communications interfaces for dynamically and partially reconfigurable systems" RC Education, Porto Allegre, Brasil, May 2007
[2] A Ehliar, D. Liu "Thinking Outside the Flow: Creating Customized Backend Tools for Xilinx Based Designs" FPGA World 2007, Stockholm, Sweden, September 2007
[3] Xilinx "XAPP 150: Virtex Series Configuration Architecture User Guide" v.1.5 09.2000
[4] Xilinx "XAPP 290: Two Flows for Partial Reconfiguration: Module Based or Differ-ence Based" v4.0 9.09.2004
[5] Xilinx "XAPP 430: Spartan-3 Advanced Configuration Architecture"

Artykuł recenzowany