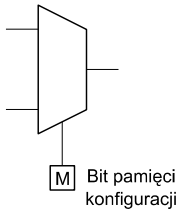
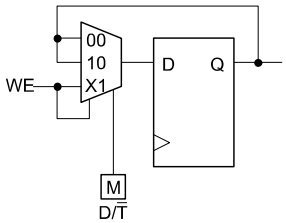
Blok funkcyjny i makrokomórka c. d.

Wypadałoby powiedzieć coś o multiplekserach, które będą się teraz często pojawiały na schematach logicznych układów. Są one multiplekserami konfigurowalnymi przy użyciu bitu pamięci konfiguracji. Multipleksery pojawiające się na schematach nie są przełączane podczas normalnej pracy układu. Konfiguruje się je tylko podczas programowania.



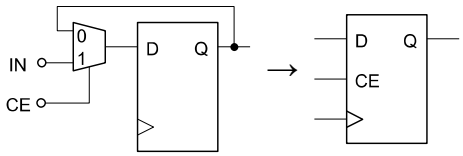
Na schemacie logicznym makrokomórki pojawiła się bramka XOR dzięki której realizuje się operację polaryzacji sygnału. Konfigurując multiplekser możemy zdecydować czy sygnał przechodzący przez bramkę XOR z Product Term Allocatora na wejście przerzutnika będzie negowany czy nie.

Przerzutnik w makrokomórce można ominąć co umożliwia pracę makrokomórki w trybie kombinacyjnym lub rejestrowym. Przerzutnik jest konfigurowalny: może działać jako przerzutnik typu D lub typu T. Z przerzutnika typu D można w bardzo łatwy sposób zrobić przerzutnik T. Potrzebny jest do tego 3-wejściowy multiplekser i pętla sprzężenia zwrotnego:



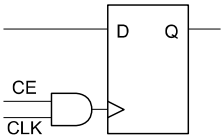
Podanie stanu wysokiego na bit sterujący sprawia, że przerzutnik będzie działał jako przerzutnik typu D. Podanie stanu niskiego zmienia tryb pracy na typ T. Podczas projektowania układu nie trzeba przejmować się sposobem wyboru przerzutnika. Tym zajmie się środowisko, w którym będziemy konfigurować układ.

Istotne jest wejście CE (Clock Enable) pozwalające na przełączenie się przerzutnika. Jeżeli to wejście jest nieaktywne, to przerzutnik podtrzymuje swój stan. Wejście CE realizuje się dzięki podłączeniu na wejście przerzutnika multipleksera. Multiplekser steruje się sygnałem CE:

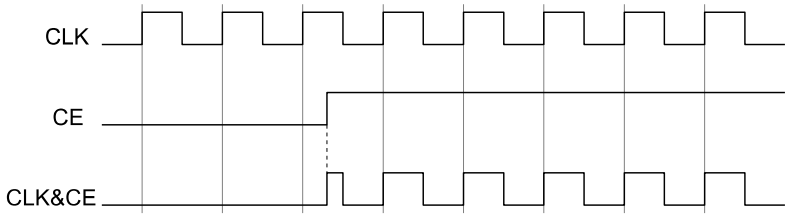


Podanie na wejście CE stanu niskiego sprawia, że przerzutnik będzie się przełączał z każdym narastającym zboczem zegarowym podtrzymując swój stan poprzedni. Takie rozwiązanie można zastosować tylko dla przerzutnika D.

Złym rozwiązaniem jest bramkowanie sygnału zegarowego:

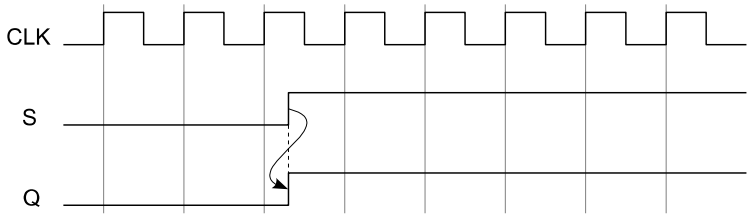


Należy zdawać sobie sprawę z tego, że każda manipulacja przy sygnale synchronizującym jest niebezpieczna. Idea synchronizacji pracy układu cyfrowego polega na tym, że momenty zbocza narastającego sygnału zegarowego są momentami przełączania się przerzutników w układzie. Jeżeli zastosowalibyśmy bramkowanie sygnału zegarowego, to może pojawić się problem w postaci narastającego zbocza zegarowego, które może pojawić się w nieodpowiednim momencie czasowym:

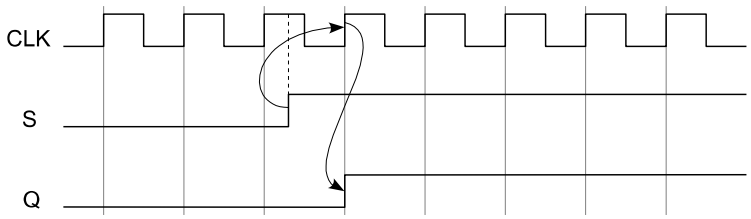


Narzędzia syntezy wyświetlają w takim wypadku komunikat ostrzegawczy „Gated Clock”. Zegary bramkowane mogą „popsuć” projekt, ponieważ bramkowanie sygnału zegarowego powoduje opóźnienia sygnału wynikające z czasów propagacji układów logicznych. Poza tym sygnałów zegarowych nie można bramkować.

Do każdej makrokomórki doprowadzone są sygnały asynchronicznego ustawiania i kasowania. Jeżeli sygnał Set działa w sposób asynchronicznie, to pojawienie się wartości aktywnej na nim wymusza natychmiastową zmianę stanu Q:



W przypadku pracy synchronicznej zmiana stanu Q nastąpi przy najbliższym zboczu zegarowym przełączającym przerzutnik:



Nazwy „Set” i „Reset” są tutaj używane niewłaściwie, ponieważ według reguły sygnały „Set” i „Reset” działają w sposób synchroniczny, a asynchronicznie działają sygnały „Preset” i „Clear”.

Gdy przyjrzymy się układowi połączeń, możemy stwierdzić że wszystkie sygnały sterujące (CLK, CE, S, R) mogą pochodzić z różnych źródeł. Mogą to być albo sygnały globalne (GCK – globalny zegarowy oraz GSR) lub lokalne w postaci obliczonych termów (PTCK – zegar, PTS – set, PTR – reset). Sygnał CE wyznacza się wyłącznie lokalnie; jest on produktem termu PTCE.

Z makrokomórki wychodzą dwa sygnały: sygnał logiczny kierowany równocześnie do bloku we/wy oraz do globalnej matrycy połączeniowej oraz sygnał PTOE (Product Term OE) sterujący buforem 3-stanowym w bloku we/wy.

Alokator termów

Zwiększa on elastyczność konfiguracji układu. Pozwala nam na decydowanie ile linii termów może być podłączonych do jednej bramki OR.

Przeważnie wykorzystywany jest do zwiększania ilości termów podłączonych do jednej bramki OR. Idea pracy

alokatora jest taka, że nieużywane termy z sąsiednich makrokomórek są w obrębie alokatora sumowane i eksportowane w dół lub w górę i dołączane w makrokomórce docelowej do bramki OR poprzez dodatkową bramkę OR. Sumowanie takie daje efekt akumulacji termów podłączonych do jednej bramki OR. Alokator termów może działać w trzech trybach: import termów, eksport termów „do góry" oraz eksport „w dół".

Zajmiemy się schematem alokatora termów. Symbolami prostokątów są oznaczone multipleksery, które decydują o tym czy dany sygnał dołączany będzie do wyjścia górnego czy dolnego. 3-wejściowa bramka OR na górze jest bramką zbiorczą, która łączy eksportowane termy z danej makrokomórki. Suma eksportowanych termów jest wysyłana w górę lub w dół. Suma ta składa się z trzech składników: termów które doszły do danej makrokomórki z góry, termów które doszły z dołu oraz termów obliczonych w danej makrokomórce. Każdy z 5-ciu termów obliczonych w danej makrokomórce może być przy pomocy multipleksera podłączony do 5-wejściowej bramki OR.

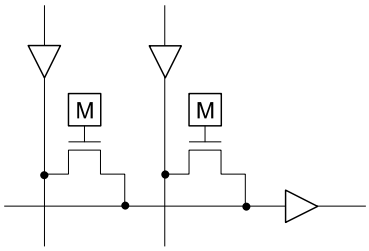
Importowanie termów może się odbywać z góry lub z dołu. Jeżeli importuje się jakieś termy, to nie dokłada się do nich termów pochodzących z danej makrokomórki. Termy importowane są sumowane ze sobą i wchodzą jako szóste wejście bramki OR (zaraz za szarą ramką).

Ze schematu widać, że sygnały sterujące są obliczane na poszczególnych termach. Term pierwszy może realizować funkcję Set lub Reset. Term drugi decyduje o polaryzacji sygnału wychodzącego z makrokomórki. Trzeci generuje sygnał PTC. Czwarty może realizować dwie funkcje: sygnału Reset lub Set. Ostatni term realizuje funkcję PTOE. Używanie termów jest ograniczone. Niemożliwa jest jednoczesna praca z pięcioma sygnałami sterującymi.

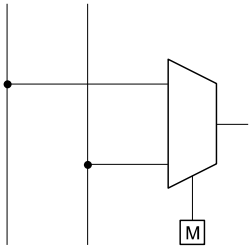
Importowanie i eksportowanie termów obarczone jest kosztami czasowymi. Opóźnienie termów importowanych t_{PTA}=0,7 ns. Dodaje się go do parametru określającego czas propagacji sygnału t_{LOGI}=1,0 ns. Term ,który jest podłączony do właściwej makrokomórki wnosi opóźnienie 1 ns. Każdy stopień importu to dodatkowe 0,7 ns (na każdy stopień alokatora). Przejście przez dwa stopnie alokatora daje opóźnienie 1,4 ns. Maksymalny stopień opóźnienia dodatkowego opóźnienia wynosi $8 \times t_{PTA}$.

Globalna matryca połączeń

Jest to niezbędny układ umożliwiający wzajemną komunikację poszczególnych bloków ze sobą. Wszystkie sygnały wejściowe i sprzężenia zwrotne przechodzą przez tą matrycę. Matryca ta musi działać szybko, by nie było mowy o wprowadzaniu dodatkowych opóźnień sygnałów. Pierwsze matryce połączeniowe zbudowane były w oparciu o punkty programowalne. Sygnały wejściowe mogły się znaleźć na liniach wyjściowych gdy punkty programowalne otwierały odpowiednie tranzystory:



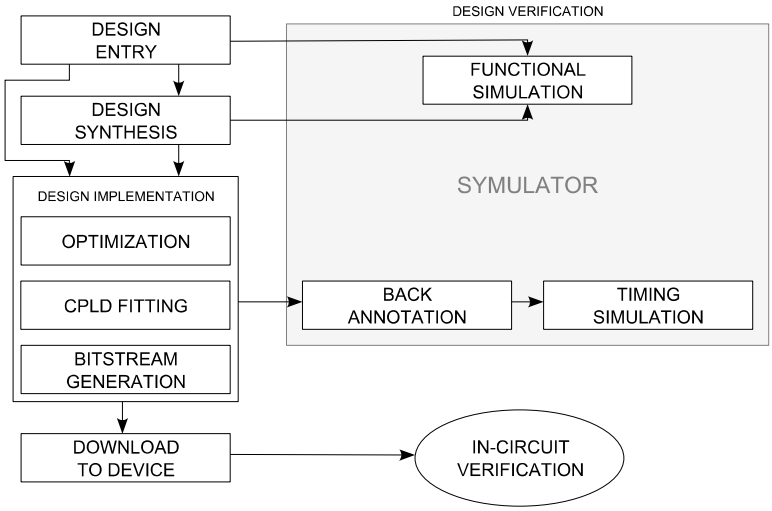
To było wolne rozwiązanie, ponieważ miała na to wpływ rezystancja kanałowa tranzystora oraz pojemność pasożytnicza, które razem wprowadzały opóźnienie. Lepszym rozwiązaniem jest zastosowanie matrycy Fastconnect:



Działanie oparte jest tu na multiplekserach. Tutaj jeden punkt programowalny decydujący o tym do którego z wejść jest podłączone wyjście matrycy. Multipleksery w technologii CMOS są łatwe do implementacji. Rozwiązanie jest szybsze.

Praca w środowisku Xilinx ISE

Poruszanie się w środowisku projektowania układów logicznych jest sporą sztuką z racji na złożony proces projektowy. Projektant układu cyfrowego powinien wiedzieć z jakimi etapami implementacji będzie miał do czynienia i co w poszczególnych etapach będzie się działo.



Zaczynamy od wprowadzenia opisu projektu (Design Entry). Można to porównać do pisania kodu programu w jakimś edytorze. Specyfikowanie projektu może się odbywać poprzez rysowanie schematu lub używanie języka opisu sprzętu. Kolejnym krokiem jest synteza projektu (Design Synthesis). W szczególności opis projektu za pomocą języków opisu sprzętu wymaga syntezy. Tekst opisu VHDL należy przełożyć (zsyntezować) na kategorie układów cyfrowych. W niektórych przypadkach można krok syntezy pominąć np. gdy opisujemy schemat wykorzystując elementy prymitywne (elementy, które nie wymagają syntezy, bo są w danej architekturze elementami podstawowymi).

Po syntezie przechodzimy do implementacji (Design Implementation) mającej na celu przygotowanie danych konfiguracyjnych, które zostaną przekazane do sprzętu i według tego strumienia będzie konfigurowany układ. W implementacji mamy krok optymalizacji (Optimalization) mający na celu uproszczenie wprowadzonego przez nas opisu. W układach CPLD jest jeszcze krok „Fitting" który polega na „wstawieniu" projektu do danej architektury układów CPLD. Na końcu mamy krok generacji strumienia bitowego, który posłuży do konfiguracji układu (Bitstream Generation). Powstaje gotowy strumień, który należy przesłać do urządzenia (Download to device).

Równolegle do tej ścieżki biegnie ścieżka symulacyjna. Mamy do czynienia z różnymi rodzajami symulacji. Symulacja funkcjonalna (Functional Simulation) przeprowadzana jest przed rozpoczęciem procesu implementacji projektu. Dokonywana jest symulacja naszego opisu układu przed lub po przeprowadzeniu procesu syntezy projektu. Symulacja ta nie uwzględnia zależności czasowych. Zależności czasowe stają się znane dopiero po przełożeniu projektu na daną architekturę układów. Wzbogacenie projektu o opis zależności czasowych oznaczony jest jako „Back Annotation". Jest to uzupełnienie opisu funkcjonalnego o parametry czasowe, które są dostępne po implementacji fizycznej układu. Wtedy możemy przejść do procesu symulacji czasowej (Timing Simulation) która pokazuje, czy wszystkie zależności czasowe są spełnione. Po przesłaniu projektu do urządzenia możemy korzystać z różnych mechanizmów debugowania projektu pracującego w układzie (In-Circuit Verification). Potrzebne do tego jest użycie specjalnych próbników logicznych umożliwiających śledzenie wykonywania się projektu w praktyce.

W środowisku ISE całością zarządza Project Manager, czyli powłoka z interfejsem użytkownika, która uruchamia kolejne etapy syntezy i implementacji. Sprawia on najwięcej kłopotów, ponieważ nie jest to powłoka pewnie działająca. Oprogramowanie wykonujące syntezę to moduły które na podstawie plików wejściowych, zawierających schematy układów lub opis sprzętu generują pliki wyjściowe, które będą służyć do konfiguracji układu.