Laboratorium do przedmiotu „Projektowanie Układów ASIC” dla studiów inżynierskich

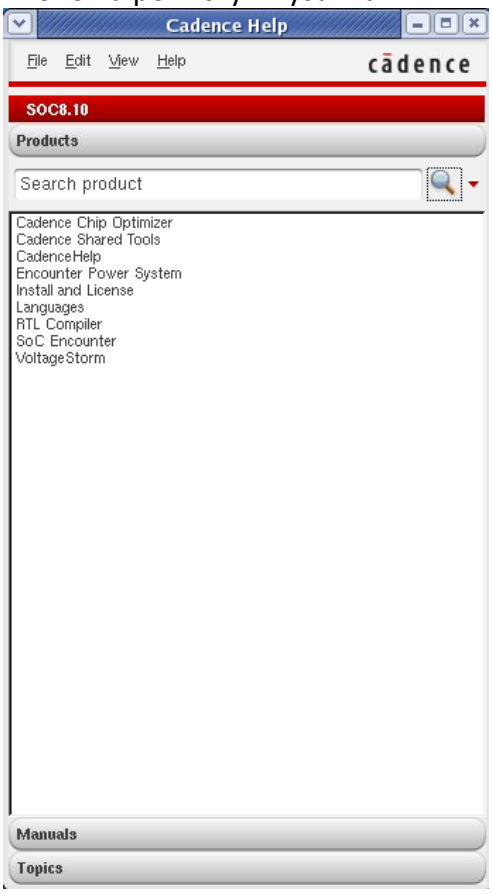
1) Organizacja laboratorium.

Wstęp. W ramach zajęć laboratoryjnych przewidziane jest wykonanie 3 prostych ćwiczeń, których zakończeniem będzie gotowy projekt rdzenia cyfrowego układu scalonego w technologii 130nm z wykorzystaniem biblioteki komórek standardowych oraz komórek I/O. Jako narzędzia projektowe wykorzystany zostanie pakiet CADENCE wydanie 2009/2010. Biblioteki komórek zostały zaprojektowane dla technologii UMC 130nm przez dyplomantów w ramach prac magisterskich. Ćwiczenia muszą być wykonane kolejno, gdyż zakończenie poprzedniego ćwiczenia daje umiejętności niezbędne do wykonania kolejnego.

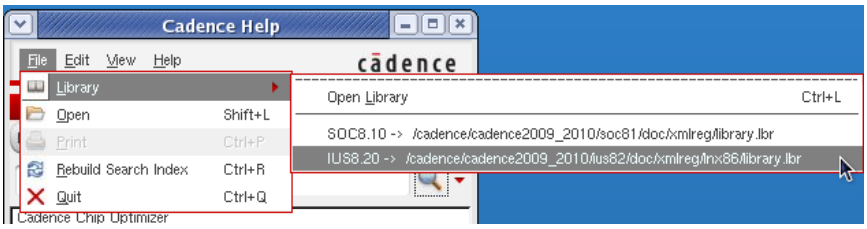
Zasady zaliczenia laboratorium. Trzeba wykonać, każde ćwiczenie przewidziane w programie laboratorium. Na zakończenie ćwiczenia i potwierdzenie jego wykonania należy wypełnić protokół dołączony do opisu ćwiczenia i wysłać jako załącznik do poczty email na adres wskazany przez prowadzącego ćwiczenia. Ocena końcowa stanowi średnią ocen z poszczególnych ćwiczeń.

2) Wiadomości ogólne.

Pakiet oprogramowania Cadence zainstalowany jest w katalogu: ***/cadence/cadence2009_2010***. Aby było można z niego skorzystać należy w terminalu uruchomić polecenie, które konfiguruje środowisko: ***source/cadence/cad9_10.sh*** lub ***source/cadence/cad9_10.csh*** w zależności od używanego shella. Następnie należy utworzyć katalog roboczy i wejść do niego i tam uruchamiać niezbędne pakiety oprogramowania. Utworzenie katalogu (o nazwie ***test_lab1***): ***mkdir test_lab1*** przejście do katalogu: ***cd test_lab1*** uruchomienie programu (np. symulacji cyfrowej) ***nclaunch –new &*** Uruchomienie dokumentacji pakietu Cadence następuje poprzez wydanie polecenia: ***cdnshelp &***. Pomoc Cadence otwiera się w oknie graficznym, jak to na przykład przedstawiono na poniższym rysunku:



U góry okna pomocy podana jest nazwa pakietu, którego pomoc została wczytana. W naszym przypadku jest to pakiet SOC8.10. Jeśli chcemy wczytać pomoc innego pakietu należy wybrać menu ***File/ Library*** i z listy pakietów wybrać stosowny. Na poniższym rysunku wybrany został pakiet IUS8.20.



W przypadku gdy na liście nie ma dostępnego pakietu należy go wczytać poprzez polecenie ***Ctrl+L*** i wybrać plik: ***/cadence/cadence2009_2010/(nazwa szukanego pakietu)/doc/xmlreg/library.lbr*** Po wczytaniu plików pomocy dotyczących pożądanego pakietu można poprzez wybór jednej z zakładek: ***Products / Manuals / Topics*** przeglądać dokumenty. Podwójne kliknięcie w wybrany temat pomocy otwiera boczne okno z treścią pomocy. Alternatywnie, poprzez kliknięcie ikony  na pasku zadań, treść pomocy można otworzyć jako dokument PDF. Pomoc jest w języku angielskim.

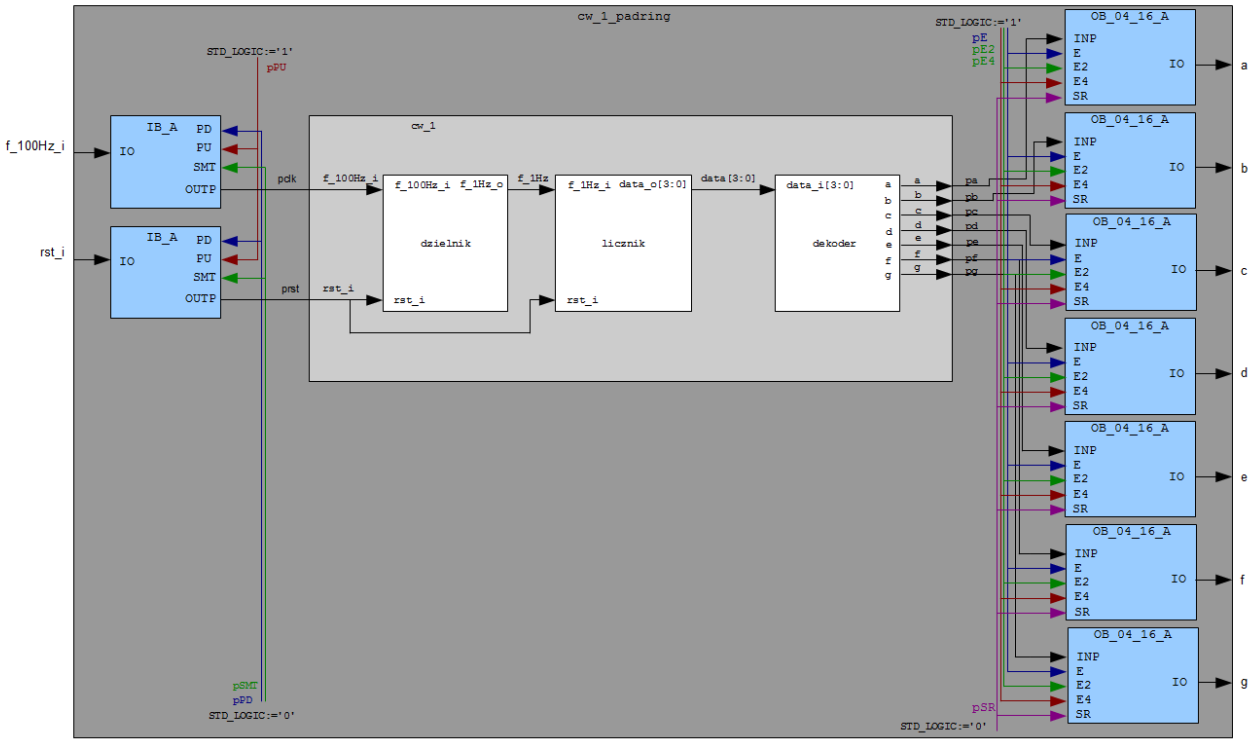
Laboratorium do przedmiotu „Projektowanie Układów ASIC” dla studiów inżynierskich Ćwiczenie 1: Opis VHDL i symulacja prostego projektu układu cyfrowego

Nieobowiązkowe ale zalecane: Dla osób chcących głębiej poznać symulator SimVision można skorzystać z pomocy Cadence uruchamiając program ***cdnshelp***. Następnie należy wybrać pakiet ***IUS8.2*** (***File/ Library/ OpenLibrary/ cadence/ cadence2009_2010/ ius82/ doc/ xmlreg/ library.lbr***). Kolejno należy wybrać ***NC-VHDL/ A Library of Tutorials and Examples/ Incisive Enterprise Simulator Tutorial with SimVision***. W tym tutorialu zawarty jest przykład projektu wyjaśniający obsługę symulatora SimVision.

W ramach ćwiczenia należy wykonać opis w języku VHDL i symulację prostego układu cyfrowego składającego się z 5 następujących bloków entity:

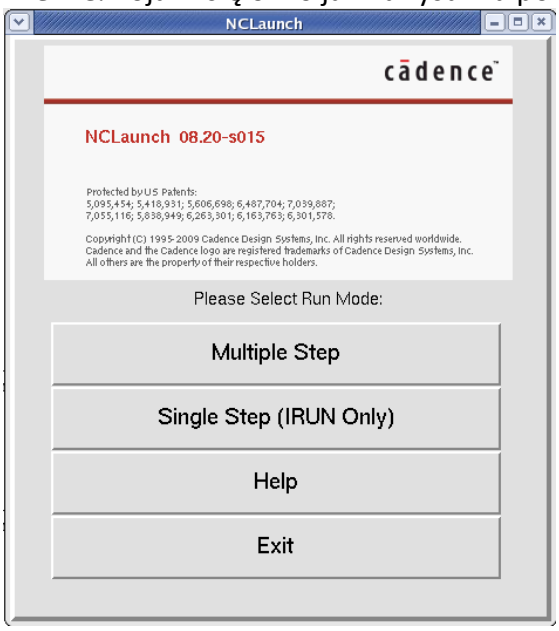
- **cw_1_padring** - blok integrujący blok cw_1 oraz bufory wejściowe (IB_A) i wyjściowe (OB_04_16_A).
    - **cw_1** – blok integrujący w sobie 3 bloki,
        - **dzielnik** – blok synchronicznego dzielnika częstotliwości z asynchronicznym resetem, podział przez 100,
        - **licznik** – blok synchronicznego licznika z asynchronicznym resetem zliczający w zakresie od 0 do 9,
        - **dekoder** – kombinacyjny blok dekodera wyświetlacza siedmiosegmentowego ze wspólną katodą (zapalanie segmentów stanem logicznej jedynki).

Bloki mają być połączone zgodnie z poniższym rysunkiem, przy czym należy koniecznie zastosować **nazwy sygnałów i bloków identyczne** jak na rysunku. Dodatkowo należy stworzyć blok entity o nazwie **cw_1_testbench** w którym należy osadzić blok ***cw_1_padring*** i wygenerować sygnały testujące wykorzystane w symulacji. Sygnał testujący ***reset_i*** powinien być aktywny do 35ms, sygnał ***f_100Hz_i*** powinien być wygenerowany jako przebieg prostokątny o częstotliwości 100Hz. Bloki buforów zostaną wczytane z wcześniej przygotowanych plików vhdl – należy je tylko osadzić. W bloku **cw_1_padring** należy dodatkowo połączyć odpowiednie wejścia sterujące buforów do sygnałów logicznych jak na rysunku. Każdy z bloków (**cw_1,dzielnik,licznik,dekoder**) należy zapisać w osobnym pliku tekstowym o nazwie takiej jak nazwa bloku z rozszerzeniem **.vhdl**. Do wykonania opisu należy wykorzystać dowolny edytor tekstowy, np. ***gedit***, ***vi***, ***emacs*** lub ***nedit***.

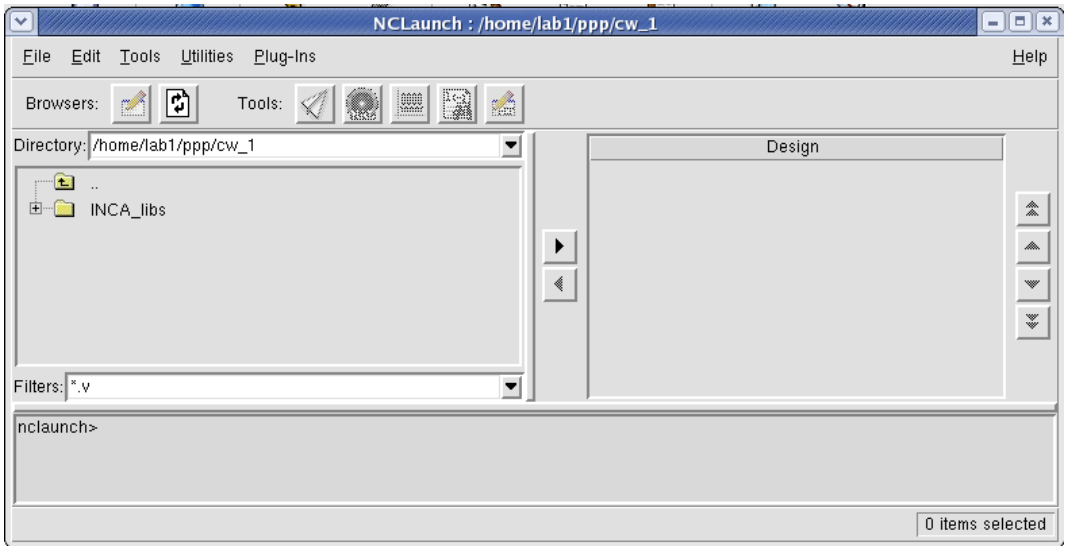


W celu sprawdzenia poprawności składni należy przeprowadzić kompilację pliku poleceniem: ***ncvhdl –messages –V93 nazwa_pliku.vhdl*** W przypadku błędu kładni pojawi się komunikat informujący o linii i kolumnie w której jest prawdopodobny błąd. W przypadku wielu błędów poprawki należy rozpoczynać od pierwszego błędu!

Po doprowadzeniu do poprawności składniowej wszystkich plików należy uruchomić symulację poleceniem ***nclaunch –new &*** Pojawi się okno jak na rysunku poniżej:



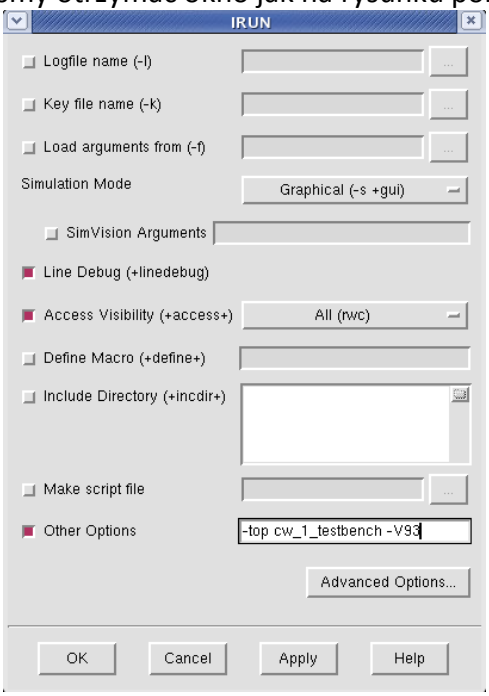
W tym momencie można wybrać dwa rodzaje symulacji: wykonywana etapowo (***Multiple Step***) lub jednokorokowo (***IRUN Only***). Dla celów naszego ćwiczenia wybieramy symulację jednokrokową. Po naciśnięciu pola ***Single Step*** powinno pojawić się okno jak na rysunku poniżej, przedstawiające zestaw plików wykorzystywanych podczas symulacji:



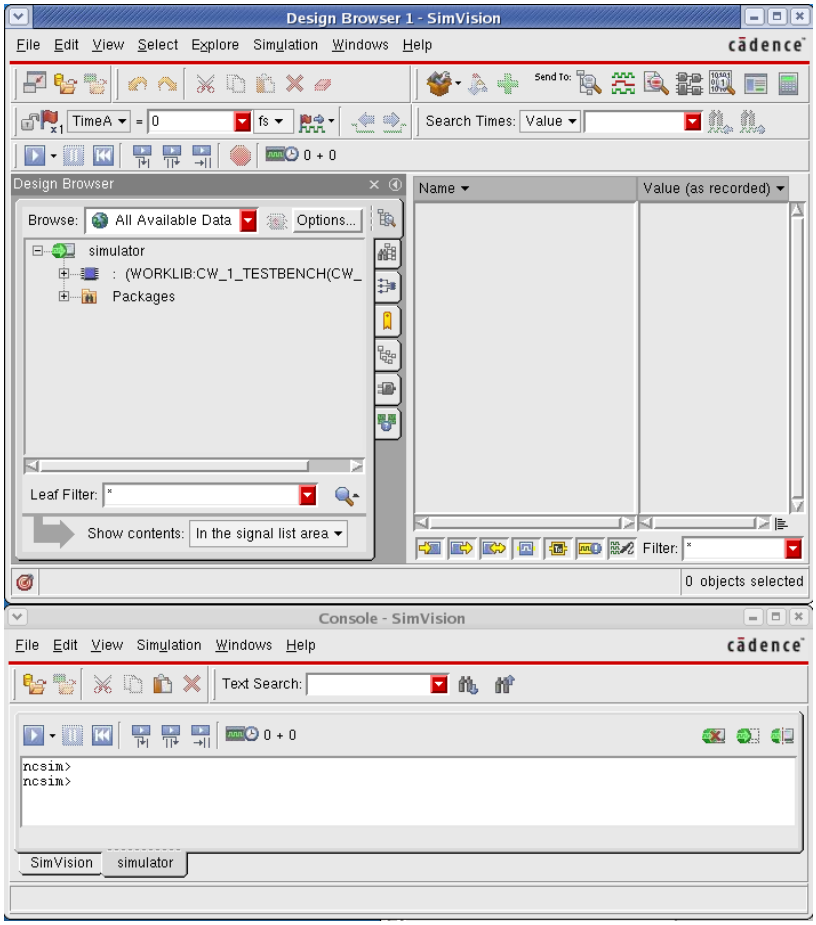
Domyślnie ustawiony jest filtr wyświetlania plików Verilog (z rozszerzeniem .v). W celu przeglądania innych plików należy w polu ***Filters*** zmienić wpis na .vhdl. Następnie należy dodać wszystkie pliki VHDL do projektu:

- dzielnik.vhdl
- licznik.vhdl
- dekoder.vhdl
- cw_1.vhdl
- cw_1_padring.vhdl
- cells_io_typical.vhdl (z katalogu /techfiles/umc130_ksme_lab/IO/VHDL)

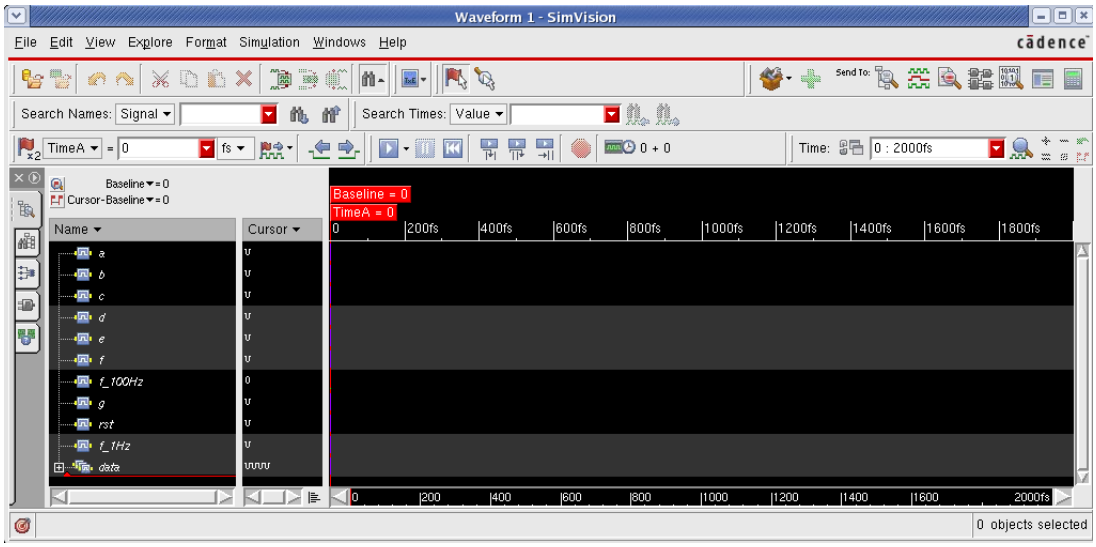
(zaznaczyć myszką i kliknąć strzałkę w prawo ▶). Następnie wybieramy menu ***Tools/ IRUN*** i zaznaczamy pole ***Other Options*** i wpisujemy obok ***–top cw_1_testbench –V93***. Dopisanie tej opcji wskazuje symulatorowi który z bloków jest najwyżej w hierarchii projektowej oraz włącza składnię języka VHDL w wersji 1993. Po tym kroku powinniśmy otrzymać okno jak na rysunku poniżej.



Kilkamy ***OK*** i o ile nie mamy błędów składni językowej automatycznie uruchamia się symulator ***SimVsion*** w postaci 2 okien zatytułowanych: ***Design Browser*** oraz ***Console*** jak to przedstawiono na poniższym rysunku.



Aby wykonać symulację i oglądnąć jej wyniki na wykresie należy „wysłać" wyselekcjonowane sygnały na wykres a następnie wykonać symulację. Wysyłanie sygnałów na wykres wykonuje się poprzez ich zaznaczenie w zakładce ***Design Browser*** a następnie kliknięcie na przycisk . Poprzez klikanie na symbol „**+**" przy nazwie naszego projektu można zmieniać hierarchię projektu i obserwować sygnały z dowolnego miejsca w projekcie. Przykładowo na rysunku poniżej do wykresu dodano wszystkie sygnały bloku ***cw_1_tesbench*** oraz wewnętrzne ***data*** i ***f_1Hz***.



Symulację uruchamia się strzałką w prawo , warto jednak przed uruchomieniem ograniczyć zakres czasu o ile zwiększony zostanie czas symulacji poprzez kliknięcie na strzałkę w dół (obok strzałki symulacji). Przed symulacją warto również wybrać jednostki czasu odpowiednie dla badanego projektu. Wykonuje się to poprzez wybór z rozwijanego menu: . Wyzerowanie symulacji wykonuję się poprzez kliknięcie w , można wówczas dokonać ponownej symulacji. Aby zmienić widoczną skalę czasu

(powiększyć lub zmniejszyć zakres czasowy obserwowanych sygnałów) można przeciągać kursor czasu znajdujący się na dole wykresu. W przypadku gdy się okaże że projekt badanego układu jest błędny należy wyedytować pliki źródłowe VHDL nanieść poprawki i ponownie uruchomić symulację. Aby jednak nie wykonywać wszystkich czynności od początku samą symulację w takim przypadku można uruchomić ponownie poprzez polecenie ***Simulation/ Reinvoke Simulator.../ Yes***. Symulator ***SimVision*** posiada szereg dodatkowych funkcji, które tu ze względu na brak miejsca i czasu nie będą szczegółowo omówione. Omówione zostaną natomiast dwie najważniejsze: poszukiwanie w kodzie źródłowym miejsca sterowania sygnałem oraz oglądanie schematów blokowych projektowanych bloków.

Poszukiwanie miejsc sterowania sygnałem Przykładowo, chcemy znaleźć w kodzie źródłowym miejsce w którym sygnał ***f_1Hz*** zmienia swoją wartość z 1 na 0. Aby to wykonać należy:

a) wykonać symulację z sygnałem ***f_1Hz*** umieszczonym na wykresie,
b) kliknąć myszką na sygnał ***f_1Hz***,
c) kliknąć na symbol  , który uruchamia funkcję poszukiwania zmian wybranego wcześniej sygnału, należy kliknąć tyle razy aż znajdziemy się w punkcie czasowym który chcemy przeanalizować,
d) wybrać z menu ***Explore/ Go To/ Cause***, pojawi się wówczas okno ***Source Browser***, w którym można po lewej stronie klikać na przypadki sterowania wybranym sygnałem, równocześnie po prawej stronie pojawia się plik źródłowy ze strzałką wskazującą na konkretną linię kodu powodującą zmianę badanego sygnału,
e) okno z kodem źródłowym służy wyłącznie do przeglądania kodu, jeśli chcemy dokonać zmian w pliku źródłowym możemy uruchomić edytor poleceniem ***Edit/ Edit File***, domyślnym edytorem systemowym jest edytor vi, jeśli ktoś chce zmienić ten edytor na inny powinien zrobić to w menu ***Edit/ Preferences/ Source Browser/ Editor Command***, wpisanie w tą linię np. polecenia ***xterm -e nedit -line %L %F*** uruchamia edytor ***nedit*** i skacze od razu do rozpatrywanej linii,
f) po zmianach w plikach źródłowych należy ponownie uruchomić symulację poleceniem ***Simulation/ Reinvoke Simulator.../ Yes***.

Przeglądanie schematów blokowych projektu

Przeglądanie schematów polega na wybraniu bloku który chcemy oglądać a następnie „wysłaniu" tego bloku do programu ***Schematic Tracer***. Na przykład chcemy oglądnąć pełen schemat całego badanego układu łącznie z testebenchem. Aby to wykonać należy:

a) w oknie ***Design Browser*** zaznaczamy ***WORKLIB:CW_1_TESTBENCH***,
b) wysyłamy zaznaczony blok do programu wykreślania schematu blokowego poprzez kliknięcie na przycisk  ,
c) otwiera się okno ***Schematic Tracer***, w którym poprzez zaznaczanie obiektu na schemacie i klikanie na ikony  możemy wchodzić głębiej lub wyżej do hierarchii projektu.

Załącznik: Wzór protokołu:

Ćwiczenie 1 – laboratorium PASIC, studia inżynierskie, sem. 7		
L.p.	**Nazwa / opis**	**Wartość**
1	Data wykonania	
2	Nazwisko i imię	
3	Zrzut okna symulacji z umieszczonymi wszystkimi sygnałami bloku głównego, czas symulacji 15ms.	
4	Plik: cw_1_testbench.vhdl	
5	Plik: cw_1_padring.vhdl	
6	Plik: cw_1.vhdl	
7	Plik: dzielnik.vhdl	
8	Plik: licznik.vhdl	
9	Plik: dekoder.vhdl	

Ćwiczenie opracowali:
Bogdan Pankiewicz i Dawid Lewańczyk,
Gdańsk, listopad 2010

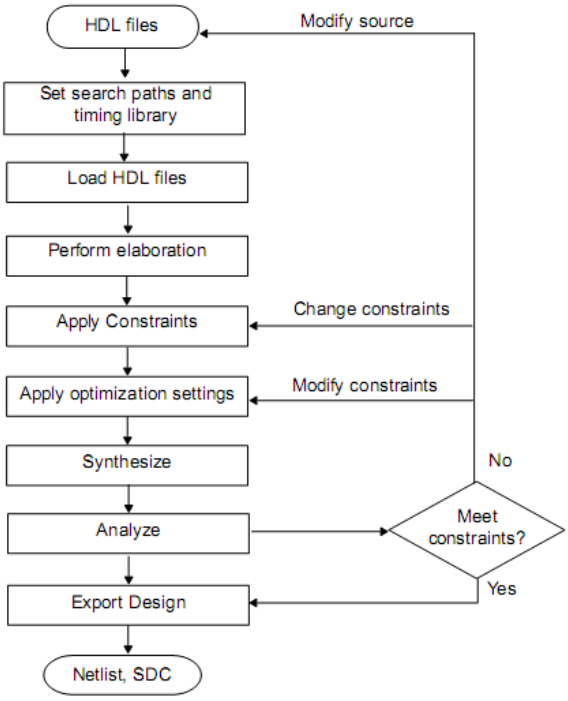
Laboratorium do przedmiotu „Projektowanie Układów ASIC” dla studiów inżynierskich Ćwiczenie 2: Synteza logiczna i symulacja po syntezie prostego projektu układu cyfrowego

Pomoc dla pakietu SOC8.1 - synteza: Dostęp do pliku pomocy użytkowika syntez era ***RTL Compiler*** jest dostępna w pliku PDF: ***/cadence/cadence2009_2010/soc81/doc/rc_start/rc_start.pdf*** oraz ***rc_user/rc_user.pdf***. Alternatywnie można uruchomić pomoc poleceniem ***cdnshelp &*** a następnie wybrać bibliotekę SOC8.10/ RTL Compiler.

W ramach ćwiczenia należy wykonać syntezę prostego projektu opisanego wcześniej w języku VHDL w ramach ćwiczenia nr 1 a następnie należy wykonać symulację układu po syntezie.

Do syntezy, w ramach pakietu CADENCE 2009/2010 należy użyć programu ***RTL Compiler***. Uruchomienie tego programu następuje z poziomu terminala poprzez polecenie:
rc lub ***rc –gui*** jeśli chcemy uruchomić syntezę z dodatkowym oknem w trybie graficznym. Po uruchomieniu programu możemy wydawać polecenia w trybie tekstowym. Pomoc ogólną można uzyskać używając polecenia ***help***. Pomoc dotyczącą konkretnego polecenia można uzyskać poleceniem ***help nazwa_polecenia***.

Typowy, ogólny przebieg projektowania układu cyfrowego z użyciem programu ***RTL Complier*** przedstawiony jest na rysunku poniżej. Jest to rysunek zaczerpnięty z dokumentacji. Przebieg takiego projektowania jest ogólny w tym sensie, że niezależnie od tego jakiego szczegółowego przebiegu projektowania się użyje, kroki z przebiegu ogólnego zawsze się powtarzają. Ten przebieg ogólny jest przebiegiem podstawowym bez używania bardziej zaawansowanych możliwości syntezera takich jak np. synteza DFT lub Low Power.



Dla celów niniejszego ćwiczenia przeprowadzimy syntezę w podstawowej postaci, osoby zainteresowane pełnymi możliwościami syntezera mogą uzupełnić swoje wiadomości w dokumentacji.

Pierwszym krokiem jest ustawienie ścieżek i bibliotek. Wszelkie ustawienia w programie wykonuje się poprzez nadanie określonym atrybutom wartości. W celu ustawienia ścieżki gdzie będą poszukiwanie pliki projektu ustawiany atrybut ***hdl_serch_patch***. Np. jeśli zamierzamy nasze pliki

źródłowe umieścić w podkatalogu ***hdl*** katalogu roboczego powinniśmy w programie ***RTL Compiler*** wpisać polecenie:

set_attribute hdl_search_path ./hdl /

Znak ***/*** na końcu polecenia oznacza nadanie atrybutu do korzenia projektu. Warto ustawić ścieżkę pod którą poszukiwane będą biblioteki komórek standardowych:

set_attribute lib_search_path /techfiles/umc130_ksme_lab/lib /

Aby sprawdzić wartość ustawionego wcześniej atrybutu można użyć polecenia

get_attribute nazwa_poszukiwanego_atrybutu.

Na komputerach laboratorium w sali 308 katalog ***/techfiles*** zawiera różne pliki technologiczne podzielone na producentów i twórców plików technologicznych i bibliotecznych. Ze względu na poufny charakter tych plików domyślny użytkownik ma dostęp wyłącznie do plików ogólnie dostępnych. W katalogu ***/techfiles/umc130_ksme_lab*** umieszczone są pliki bibliotek używanych w czasie laboratorium. Katalog ten ma kilka podkatalogów:

- ***lib*** w którym umieszczone są opisy komórek standardowych w formacie ***Liberty***, w podkatalogu tym umieszczone są 3 pliki opisujące komórki standardowe typowe, szybkie i wolne,
- ***techLEF*** zawierający ogólny plik technologiczny LEF, jest to plik opisujący parametry warstw metalu służących do połączeń pomiędzy komórkami standardowymi,
- ***cellsFEF*** zawierający m.in. opis kształtów i miejsc wyprowadzeń komórek standardowych,
- ***verilog*** oraz ***VHDL*** zawierające opisy komórek w formatach Verilog i VHDL i służące do symulacji projektów, opisy te są dla warunków typowych jak i graniczne najszybsze i najwolniejsze,
- ***IO*** w którym umieszczone są pliki jw. tyle że dotyczące komórek wejścia – wyjścia.

W programie RC-Compiler możliwe jest uruchamianie poleceń w postaci skryptów. W tym celu należy użyć polecenia ***include nazwa_skryptu***. Program w czasie pracy rejestruje wszystkie wydane polecenia do domyślnego pliku ***rc.cmd***, rejestruje również wszystkie komunikaty konsoli do domyślnego pliku ***rc.log***. Aby kolejnego dnia uruchomić wszystkie wcześniej wykonane polecenia wystarczy uruchomić ***include rc.cmdxx*** gdzie ***xx*** jest ostatnim numerem logu.

Wracając do ćwiczenia, wszystkie pliki HDL należy umieścić w podkatalogu ***hdl*** i ustawić atrybuty dla plików HDL i bibliotek tak jak to przedstawiono w powyższych przykładach poleceń. Następnie wczytujemy biblioteki poleceniem:

set_attribute library {cells_typical.lib cells_best.lib cells_worst.lib } /

W tym miejscu można dodatkowo wczytać pliki LEF technologii oraz komórek. Zmienia to automatycznie model obliczeniowy opóźnień układu z modelu bazującego na obciążeniach sieci na model PLE skutkując dokładniejszą oceną opóźnień modelu po syntezie.

set_attribute lef_library {../techLEF/tech8m2t/tech8m2t.lef ../cellsLEF/cells.lef} /

Jeśli wczytujemy pojedynczy plik biblioteczny wówczas nie potrzeba stosować nawiasów klamrowych {}. Dla wielu plików przyjmowana jest nazwa biblioteki jak pierwszego pliku a później dołączane są kolejne pliki. Następnie musimy zapobiec uproszczeniu komórek wejściowych (IB_A), które z racji spełniania roli wyłącznie buforów zostaną usunięte w procesie optymalizacji. Wykonujemy to poprzez wpisanie polecenia:

set_attribute preserve true IB_A

oraz

set_attribute preserve true OB_04_16_A.

W kolejnym kroku wczytujemy pliki źródłowe kodu HDL. Ponieważ są to pliki wykonane w ćwiczeniu nr 1 ich wczytanie będzie następujące:

read_hdl -vhdl {cw_1_padring.vhdl cw_1.vhdl dekoder.vhdl dzielnik.vhdl licznik.vhdl}

Pliki źródłowe nie mogą mieć zakończeń linii typu Windows/DOS bo to powoduje błędy wczytania. Parametr –vhdl oznacza że wczytywane pliki są w języku VHDL, domyślnie syntezer wczytuje pliki Verilog. Kolejnym krokiem jest elaboracja projektu polegająca na: zbudowaniu struktury danych, wstawieniu rejestrów do projektu, przeprowadzeniu optymalizacji na wysokim poziomie (np. usuwanie nieużywanych fragmentów kodu) i sprawdzeniu poprawności semantycznej. Elaborację wykonuje się poprzez polecenie ***elaborate***. Po elaboracji w oknie graficznym (o ile ***RTL Compiler*** był uruchomiony z opcją ***–gui***) można

przeglądać schemat blokowy projektu oraz jego kod HDL. Przemieszczanie się po hierarchii projektu następuje poprzez rozwijane znakiem „+" menu w zakładce ***Hierarchy***.

Kolejnym krokiem projektowym jest nałożenie ograniczeń (ang. constrains). Ograniczenia można podzielić na 3 podstawowe kategorie:

- warunki pracy,
- przebiegi zegarowe,
- ograniczenia czasowe wyprowadzeń I/O.

Nałożenie ograniczeń można wykonać na kilka sposobów:

- wpisanie manualne w oknie programu RTL Compiler,
- wczytanie pliku z ograniczeniami,
- wczytanie pliku z ograniczeniami w formacie SDC.

W ramach niniejszego ćwiczenia nałożone zostaną podstawowe ograniczenia czasowe. Zdeklarowanie sygnału, który jest zegarem o nazwie CLK, okresie 3500 (wszystkie jednostki czasowe w programie są w ps) i przypisany do portów o nazwie rozpoczynającej się ***f_*** wykonuję się poprzez polecenie:

define_clock -name CLK -period 3500 [find / -port f_*]

Dodatkowo na pozostałe wyprowadzenia nadajemy opóźnienie w dotarciu sygnału o wartości 100ps.

external_delay -output 100 -clock CLK [find / -port a]
external_delay -output 100 -clock CLK [find / -port b]
external_delay -output 100 -clock CLK [find / -port c]
external_delay -output 100 -clock CLK [find / -port d]
external_delay -output 100 -clock CLK [find / -port e]
external_delay -output 100 -clock CLK [find / -port f]
external_delay -output 100 -clock CLK [find / -port g]
external_delay -input 100 -clock CLK [find / -port rs*]

Polecenie ***find*** działa na hierarchii projektu i może służyć do automatyzowania skryptów. Zawsze, jeśli nie jest się pewnym wyników działania tego polecenia możne je uruchomić niezależnie w konsoli programu. W kolejnym kroku wykonywana jest synteza. Plecenie jest następujące:

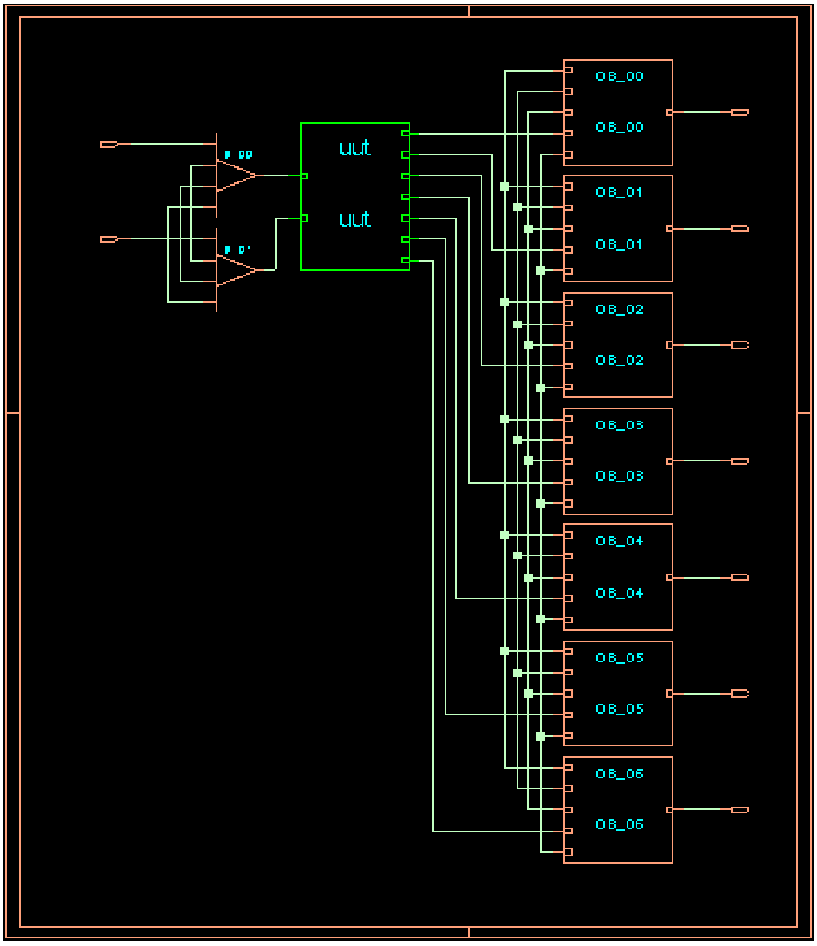
synthesize -to_mapped

Po syntezie otrzymujemy listę połączeniową wykorzystującą komórki standardowe. W oknie graficznym programu możemy teraz obejrzeć schemat zbudowany w oparciu o komórki fizyczne a nie tak jak poprzednio z wykorzystaniem ogólnych bramek logicznych.

Końcowym krokiem jest sporządzenie raportów i wygenerowanie końcowej listy połączeń.

report timing
report area
report timing > ./rpt/timing_umc.rpt
report area > ./rpt/area_umc.rpt
write_hdl > ./out/cw_1_umc.v
write_sdf > ./out/cw_1_umc.sdf
write_sdc > ./out/cw_1_umc.sdc

Jako wynik powyższych poleceń, w katalogach ***rpt*** oraz ***out*** stworzone zostaną raporty oraz wyjściowa lista połączeniowa (katalogi te muszą wcześniej istnieć). Dodatkowo w katalogu ***out*** wypisywany jest zbiór ograniczeń projektowych w formacie SDC, który to posłuży później do wczytania podczas implementacji projektu. Rezultatem pierwszej ćwiczenia powinien być wynik jak na rysunku poniżej:



Druga część ćwiczenia ma na celu wykonanie symulacji układu po syntezie. Symulację przeprowadza się identycznie jak poprzednio – należy jednak do programu ***NCLaunch*** wczytać pliki: ***/techfiles/umc130_ksme_lab/verilog/cells_typical_corrected.v*** – będący biblioteką w języku Verilog komórek standardowych,
./out/cw1_umc.v – będący listą połączeniową projektu otrzymaną po syntezie oraz ***./hdl/cw_1_testbench.vhdl*** – będący testbenchem uprzednio przygotowanym dla symulacji funkcjonalnej. Dla niniejszej symulacji należy zwiększyć częstotliwość zegara do 100MHz oraz przyspieszyć deaktywację resetu. Na wykresie wyjściowym widoczne będą opóźnienia powodowane przez bramki i przerzutniki użyte fizycznie przez syntezer. Uwaga: pliki w prawym oknie programu ***NCLaunch*** muszą występować w kolejności j.w.

Załącznik: Wzór protokołu:

Ćwiczenie 1 – laboratorium PASIC, studia inżynierskie, sem. 7		
L.p.	**Nazwa / opis**	**Wartość**
1	Data wykonania	
2	Nazwisko i imię	
3	Zrzut okna graficznego syntezera – po syntezie, hierarchia główna projektu ***cw_1_padring***	
4	Zrzut okna graficznego syntezera – po syntezie, rozwiniety bloku ***cw_1***	
5	Zrzut okna graficznego syntezera – po syntezie, rozwiniety bloku ***licznik***	
6	Zrzut okna graficznego syntezera – po syntezie, rozwiniety bloku ***dzielnik***	
7	Zrzut okna graficznego syntezera – po syntezie, rozwiniety bloku ***dekoder***	

Ćwiczenie opracowali:
Bogdan Pankiewicz i Dawid Lewańczyk,
Gdańsk, listopad 2010

Laboratorium do przedmiotu „Projektowanie Układów ASIC” dla studiów inżynierskich
Ćwiczenie 3: Implementacja rdzenia układu scalonego

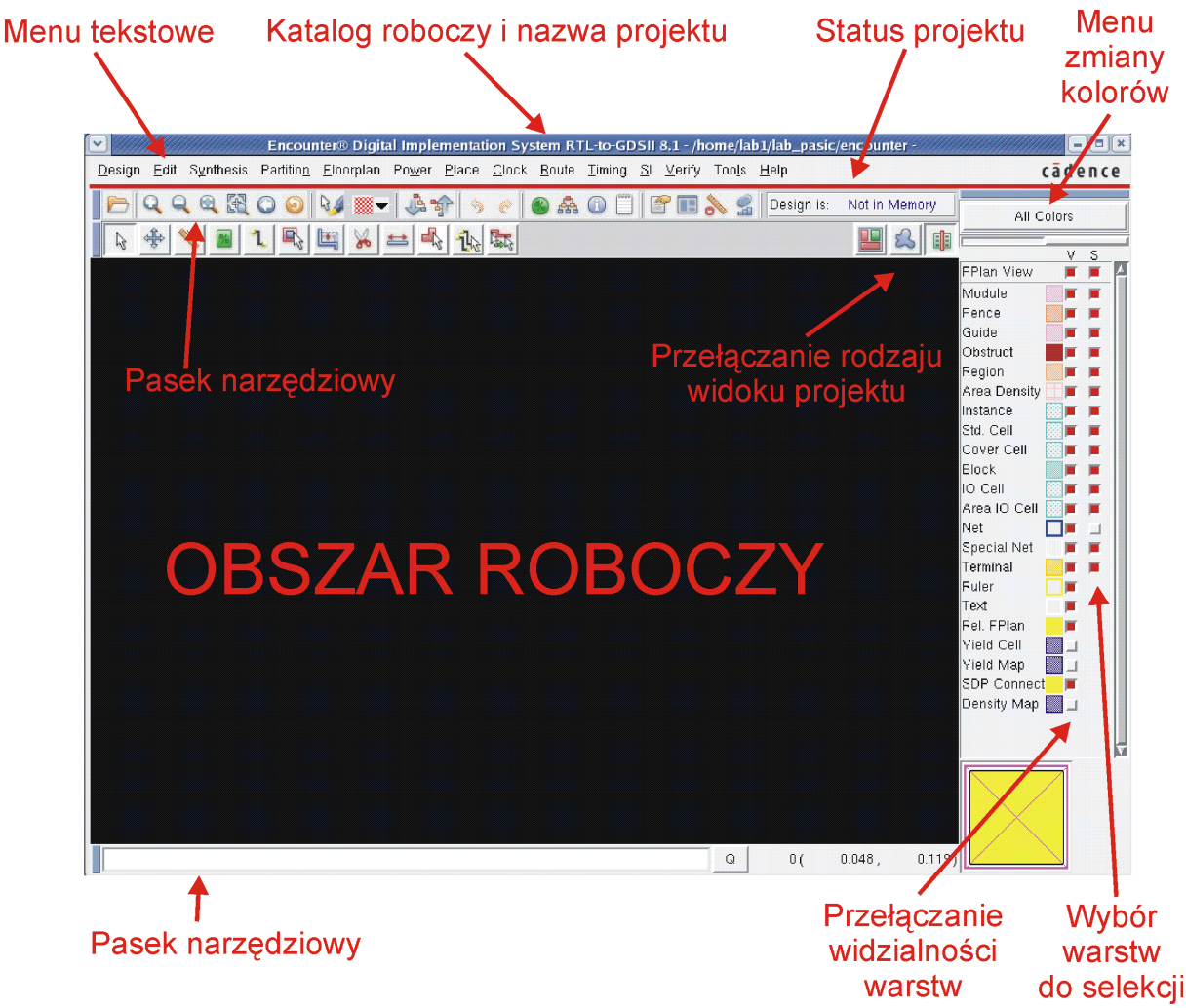
Pomoc dla pakietu SOC8.1 - implementacja: Dostęp do pliku pomocy użytkowania programu SOC Encounter dostępny jest w pliku PDF ***/cadence/cadence2009_2010/soc81/soceUG/soceUG.pdf***. Alternatywnie można skorzystać z pełniejszego pakietu pomocy poprze uruchomienie programu ***cdnshelp*** i wybranie biblioteki ***SOC8.10*** a następnie produktu ***SOC Encounter***. Chętni mogą również samodzielnie wykonać workshop umieszczony w katalogu ***/cadence/cadence2009_2010/soc81/share/fe/gift/tutorials/dtmf/work_fe***.

W ramach niniejszego ćwiczenia zostanie wykonana implementacja rdzenia układu scalonego wraz z wyprowadzeniami I/O. Implementacja wykonana w ramach ćwiczenia dotyczyć będzie projektu otrzymanego po syntezie w ćwiczeniu drugim. W celach porządkowych należy utworzyć katalog roboczy dla celów implementacji np. ***mkdir encounter*** a następnie należy wejść do tego katalogu poleceniem ***cd encounter***. Uruchomienie programu ***SOC Encounter*** następuje po wydaniu polecenia ***velocity***. Uruchomiony program ma 2 okna – konsolę służącą do wydawania poleceń tekstowych jak również okno graficzne. Kolejność wykonania projektu jest następująca:

- import wyników syntezy i bibliotek,
- planowanie ułożenia elementów (Floorplaning),
- trasowanie linii zasilających,
- rozmieszczanie komórek standardowych,
- wstępne trasowanie połączeń,
- ekstrakcja RC i generacja modelu obciążeniowego,
- analiza spełnienia czasów ustalania (setup),
- optymalizacja czasowa ze względu na czas setup,
- synteza drzewa zegarowego,
- analiza czasów hold,
- optymalizacja czasowa,
- trasowanie szczegółowe,
- analiza przesłuchów sygnałów,
- naprawa sieci z przesłuchami,

Poza przedstawionymi powyżej, podstawowymi krokami implementacji w rzeczywistości wykonuje się często dodatkowe kroki takie jak np.: analiza rozpraszania mocy i elektromigracji na liniach zasilających, analiza efektu anteny, reorganizacja łańcucha skanowania i inne.

Po uruchomieniu programu ***SOC Encounter*** (polecenie ***velocity***) otworzy się okno jak na rysunku poniżej. Na rysunku wyjaśniono znaczenie ważniejszych elementów interfejsu użytkownika.



Przyciski myszki mają następujące funkcje:
Lewy klawisz (oznaczany dalej w tekście skrótem LKM):

- klik – zaznaczanie/podświetlanie obiektów oraz wyświetlanie właściwości obiektów,
- Shift + klik – zaznaczanie/odznaczanie obiektów dodatkowych,
- podwójne kliknięcie – otwarcie formularza ***Attribube Editor***.

Środkowy klawisz myszki (oznaczany dalej w tekście skrótem ŚKM):

- klik na obiekcie otwiera dodatkowe menu kontekstowe.
- Prawy klawisz myszki (oznaczany dalej w tekście skrótem PKM):
- klik i przeciągnięcie – zoom wyświetlanego obszaru,
- Shift + klik – przesuwanie wyświetlanego obszaru.

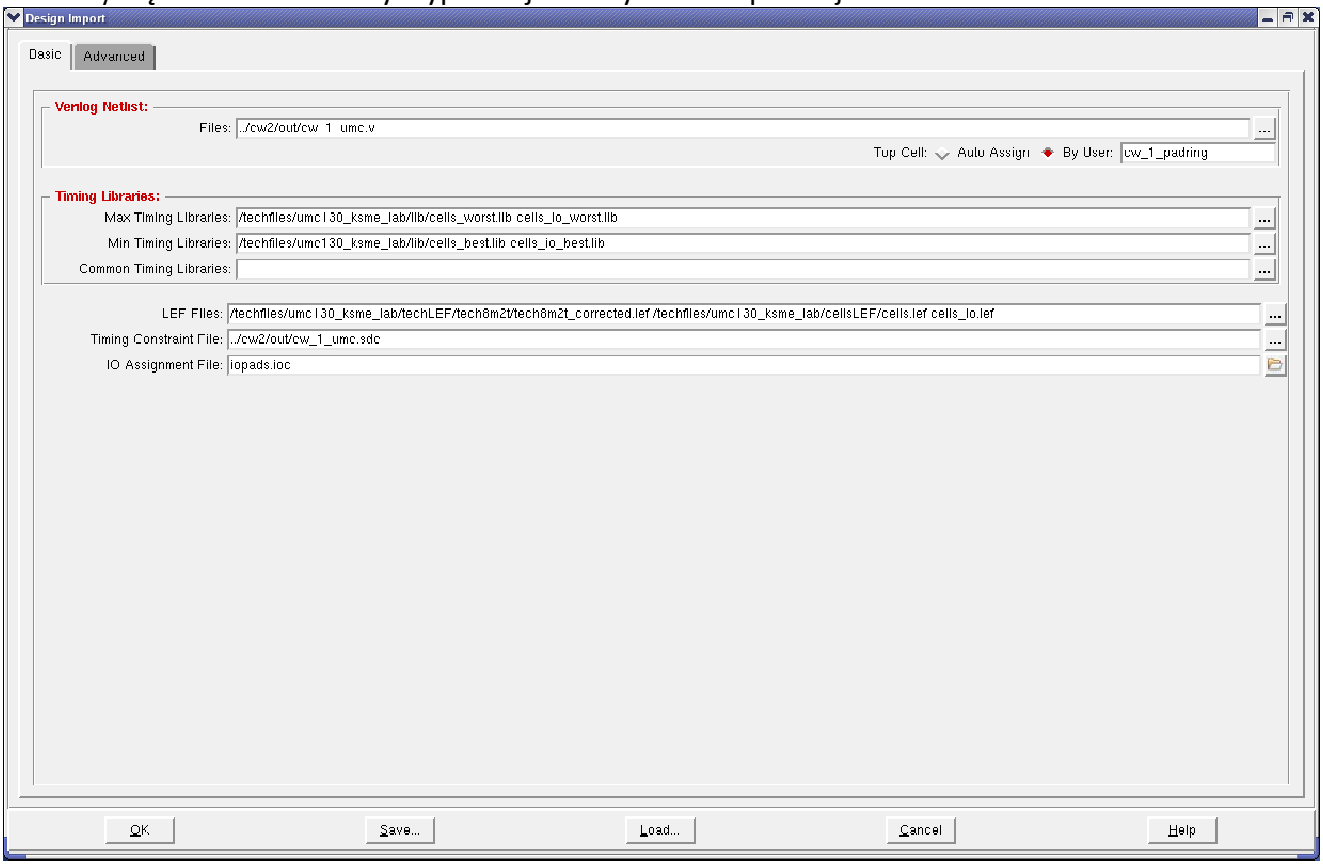
Polecenia można wydawać poprzez ich wybór z menu myszką, przez wydanie polecenia w oknie terminala oraz poprzez klawisze skrótów. Poniżej przedstawione są niektóre domyślne klawisze skrótów:

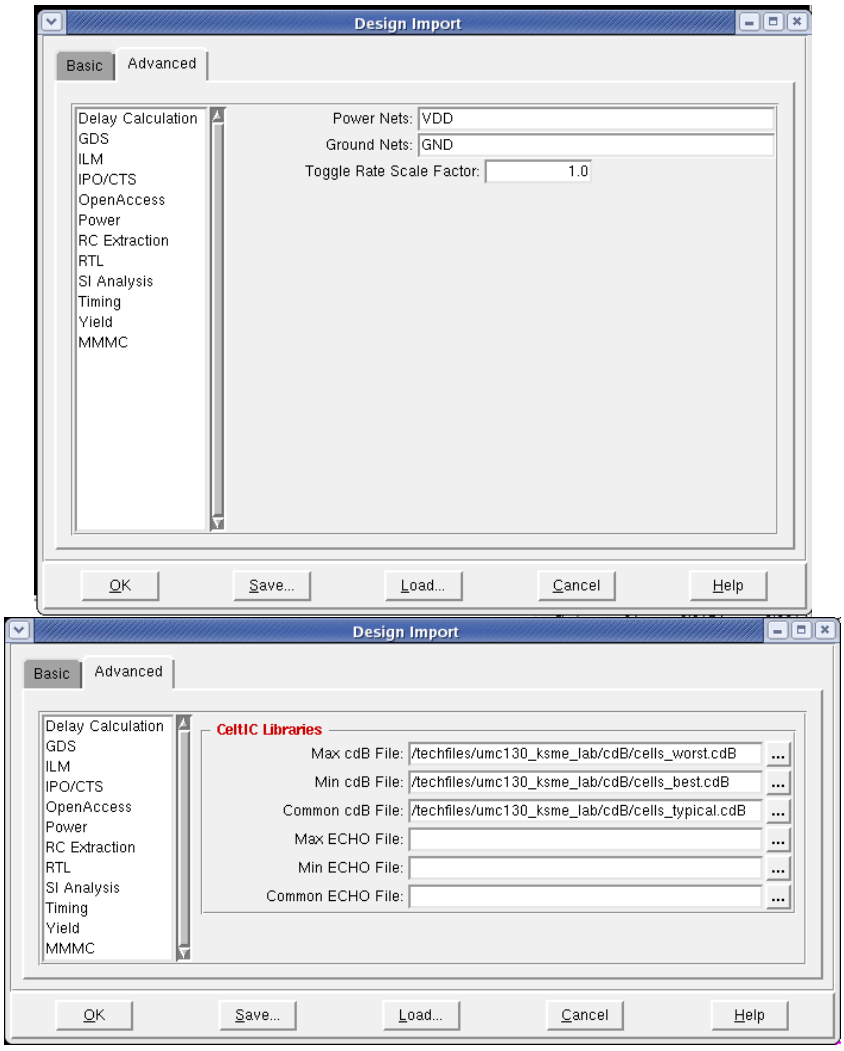
Klawisz	Opis działania
b	Otwiera formularz przypisywania klawiszy
f	Powiększa obraz do widoku całego rdzenia
g	Przesuwa hierarchię znaczonego obiektu do góry
G	Przesuwna hierarchię zaznaczonego obiektu w dół
k	Tworzy linijkę
K	Usuwa linijkę
q	Otwiera edytor atrybutów zaznaczonego obiektu

v	Wyświetla atrybut zaznaczonego obiektu
u	Polecenie undo – cofa ostatnio wykonaną czynność
U	Cofa polecenie undo
z	Powiększa wyświetlane obiekty 2 krotnie
Z	Zmniejsza 2 krotnie powiększenie
Strzałki	Przesuwa widok w kierunku strzałki
Ctrl-a	Otwiera formularz wyrównania w pionie/poziome
Ctrl-d	Deselekcja wybranej wcześniej instancji
Delete	Kasuje wybraną instancję
Shift	Umożliwia wybór wielu elementów
Spacja	Zmiana wyboru elementów nałożonych na siebie
n	Zmiana wyboru elementów nałożonych na siebie (Auto Query)
p	Zmiana wyboru poprzednich nałożonych elementów na siebie (Auto Query)

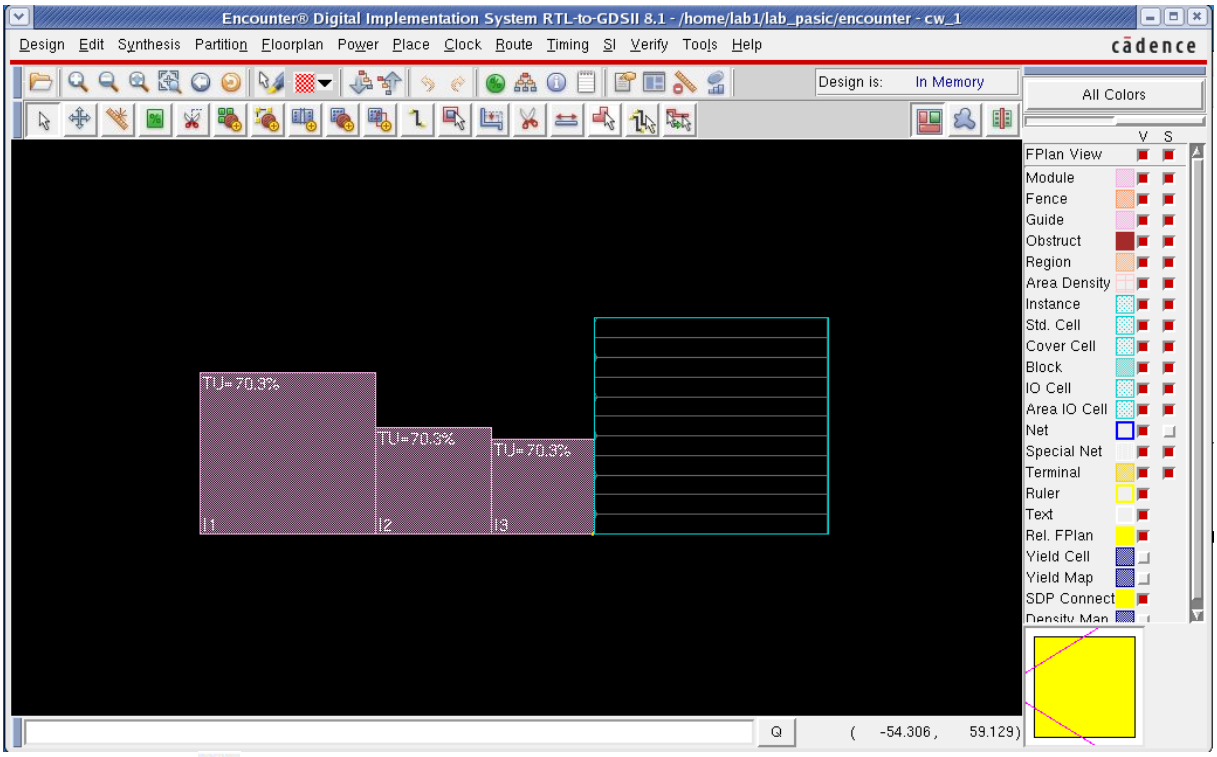
1) Import projektu

Pierwszą czynnością jest wczytanie wyników syntezy oraz bibliotek niezbędnych do przeprowadzenia implementacji. W tym celu należy wybrać menu ***Design/Import Design***. Otworzy się okno które należy wypełnić jak na rysunkach poniżej.





Po wypełnieniu wszystkich zakładek ***Basic/Advanced*** jak na powyższych rysunkach kilkamy ***OK*** i obserwujemy w konsoli komunikaty. Nie powinno być żadnych błędów (ostrzeżenia są dopuszczalne). Odwrotna kolejność wczytania plików LEF powoduje błąd importu projektu. W przypadku błędów należy uruchomić program ponownie i ponownie wczytać projekt. Aby za każdym razem nie wypełniać na nowo okna ***Design Import*** można konfigurację okna zapamiętać klikając przycisk ***Save...*** Status projektu powinien się zmienić na ***In Memory*** a na ekranie powinien pojawić się kwadrat obrazujący miejsce na przyszły rdzeń układu scalonego. Obok tego kwadratu u dołu po lewej stronie zgromadzone są komórki, które należy ułożyć w kwadracie. Początkowo, ze względu na ustawiony wysoki domyślny próg widoczności komórek, może nie być nic widoczne (zależy to od konkretnego projektu i użytych bloków). Zmianę progu można wykonać poprzez menu ***Design/ Preference/ Display*** i zmianę parametru ***Min. Floorplan Module Size:*** ze 100 na 1. Po wykonaniu tych czynności i zmianie powiększenia powinniśmy mieć okno graficzne jak na rysunku poniżej.



Kliknięcie na  uruchamia przeglądnie elementów projektu w postaci hierarchicznej. Możliwe są również inne czynności – po najechaniu na dowolny przycisk paska narzędziowego po chwili pojawia się skrócony opis jego funkcji. Najechanie LKM na obiekt (np. I1, I2 lub I3) i pojedyncze klikniecie wskazuje niebieską linią z jakimi innymi modułami dany się łączy. Podwójne kliknięcie LKM powoduje otworzenie edytora atrybutów.

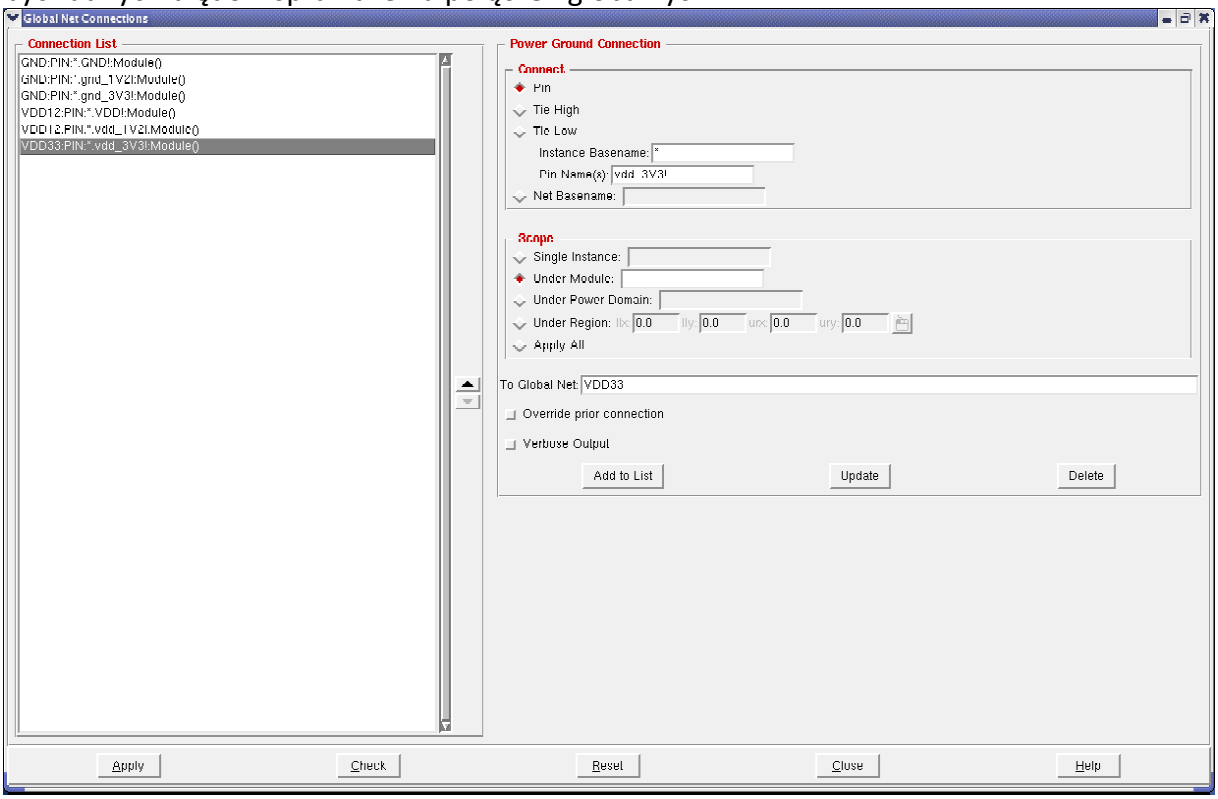
Kolejnym krokiem jest wykonanie wstępnego planu rozłożenia elementów rdzenia oraz zaplanowanie kształtu i procentowego wykorzystania powierzchni rdzenia (ang. floorplaning). W tym celu należy wybrać menu ***Floorplan/ Specify Floorplan***. Otworzy się okno, gdzie możemy wypełnić i zaplanować kształt rdzenia. Tu zmienimy ***Core utilization*** na 80%, ***Ratio (H/W)*** na 0.6, klikamy na ***Core Margins by: Core to IO Boundary*** i wpisujemy wszystkie 4 marginesy na równe 12 a następnie klikamy LKM na ***OK*** - otrzymamy zmieniony kształt i powierzchnię przyszłego rdzenia. Marginesy ustawione powyżej są miejscem oddzielającym rdzeń układu od wyprowadzeń IO i zazwyczaj są tu umieszczane pierścienie rozprowadzające zasilanie układu scalonego. Po ustaleniu wymiarów rdzenia można zdefiniować w nim obszary zabronione dla umieszczania elementów (komórek standardowych i bloków macro) jak też zabronione dla prowadzenia ścieżek połączeniowych. Można też samodzielnie ręcznie przesuwać elementy na obszar
rdzenia w celu ich umieszczenia. Aby umieścić blokadę miejsca do umieszczania elementów należy kliknąć na  , następnie LKM kliknąć w 2 przeciwległych rogach prostokąta obrazującego miejsce zabronione. Kształt takiej blokady można zmieniać poprzez kliknięcie na  a następnie przeciąganie miejsca zabronionego uprzednio utworzonego. Po najechaniu na  a następnie podwójne kliknięcie LKM na blokadę otwiera się edytor atrybutów blokady i możemy zmienić jej właściwości np. spowodować blokadę tylko 50% powierzchni.

2) Zapis i odtworzenie bieżącego stanu projektu

W każdej chwili można zapisać jak i odtworzyć bieżący stan projektu. Aby to wykonać należy wybrać menu ***Design/ Save Design As/ SoCE*** a następnie wpisać nazwę pliku pod którym ma być zapamiętany projekt. Odtworzenie wykonuje się poprzez menu ***Design/ Restore/ SoCE*** a następnie wybór nazwy pliku z zapisanym uprzednio projektem.

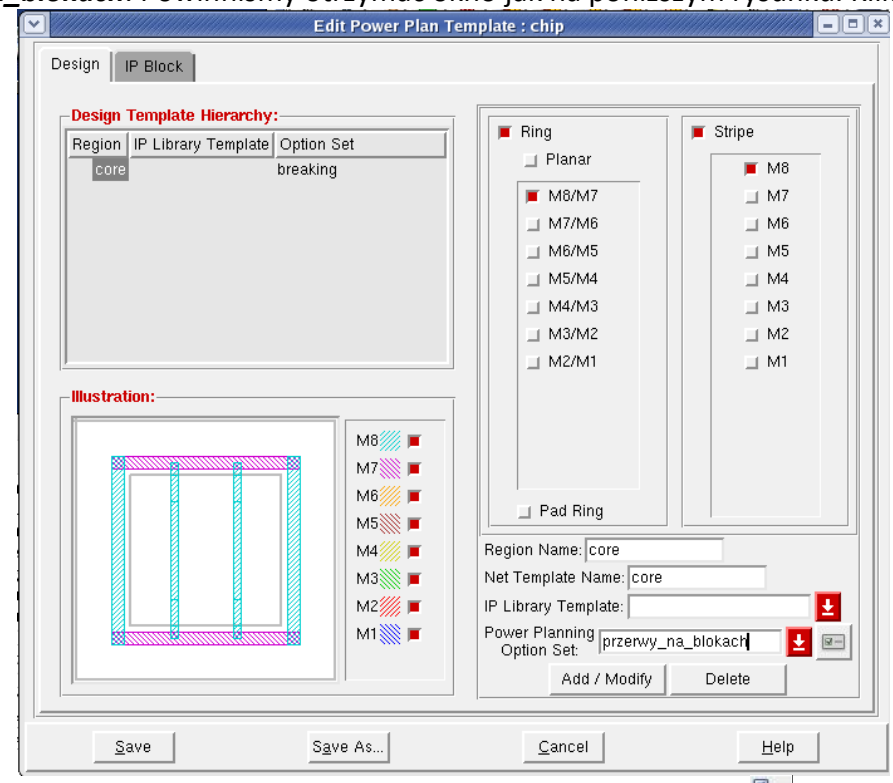
3) Projektowanie linii zasilających

Kolejnym krokiem jest zaplanowanie linii zasilających naokoło rdzenia jak i w samym rdzeniu. W tym celu wybieramy menu ***Floorplan/ Connect Global Nets***. Najpierw należy wprowadzić listę połączeń globalnych - w naszym przypadku będzie to wskazanie aby wyprowadzenia VDD! komórek standardowych (PIN VDD!) zostały połączone do sieci globalnej VDD12 oraz wyprowadzenia GND! komórek standardowych do sieci globalnej GND. W tym celu musimy wpisać ***VDD!*** do pola ***Pin Name(s)*** oraz ***VDD12*** do pola ***To Global Net*** i kliknąć LKM na ***Add to List***. Tą samą czynność należy powtórzyć dla wpisów ***GND!*** i ***GND***, podobnie podłączamy **vdd_1V2!** do **VDD12** oraz **vdd_3V3!** do **VDD33**,po tej operacji powinniśmy mieć okno jak na rysunku poniżej. Następnie kilkamy LKM na ***Apply*** i następnie ***Check***. W konsoli nie powinno być żadnych błędów sprawdzenia połączeń globalnych.

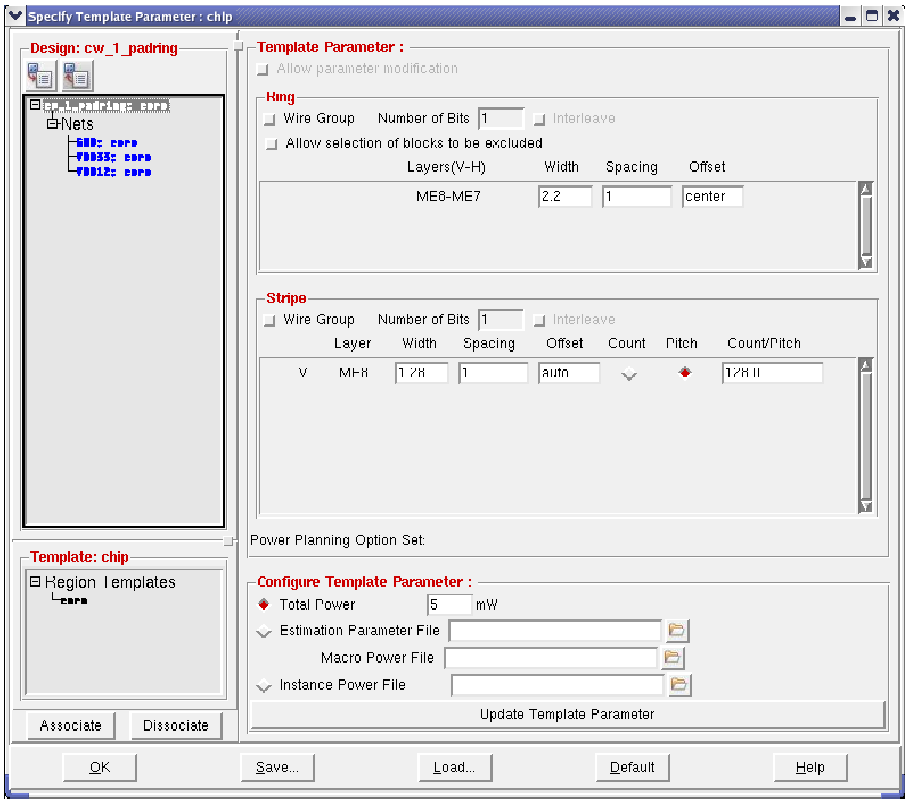


Kolejnym krokiem jest synteza połączeń linii zasilających. Wybieramy menu ***Power/ Power Planning/ Synthesize Power Plan***, w oknie wpisujemy ***Total Average Power*** 15mW, ***IR Drop Threshold*** 0.05V a następnie zaznaczmy ***Use template to create power plan*** oraz zaznaczmy ***Design***. Następnie kilkamy na klawisz tworzenia nowego planu mocy  i powinno się otworzyć kolejne okno edycji zasilania. Zaznaczy opcję ***Ring/ M8/M7***, następnie zaznaczamy ***Stripe/M8***. W oknie ***Illustration*** widzimy automatycznie efekty naszych wyborów. W pole ***Region Name*** wpisujemy ***core*** a następnie klikamy na ***Add/Modify*** a później na ***Save As..*** i podajemy nazwę ***chip***. Okno ustawień powinno się zamknąć a w dalszym ciągu powinno być widoczne okno ***Synthesize Power Plan***. W ten sposób utworzyliśmy szablon ustawień

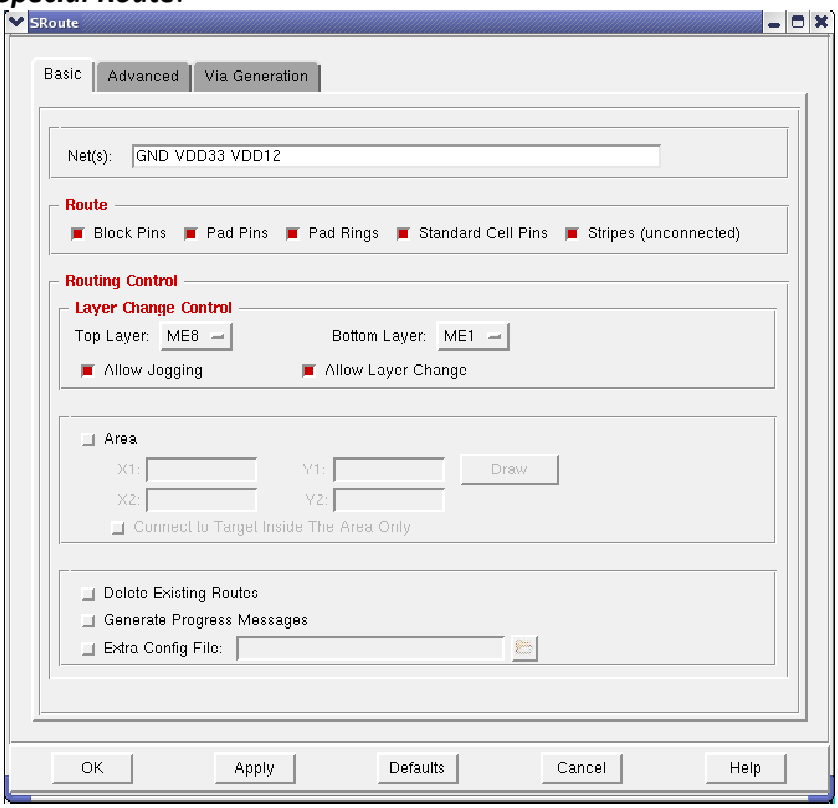
trasowania linii zasilających, który jeszcze nie jest użyty a dopiero przygotowywany do zastosowania. Zanim jednak go zastosujemy dokonamy w nim dalszych modyfikacji. Kilkamy menu ***Power/ Edit Power Planning Option...*** otwiera się okno dodatkowych ustawień trasowania linii zasilających. Następnie wybieramy ***Object: Stripe*** i w sekcji ***Stripe Breaking*** zaznaczamy ***Omit stripes inside block rings***. Działanie opcji wyjaśniane jest na bieżąco graficznie w okienku powyżej. Następnie klikamy powyżej na ***Add/Modify***, w polu ***Power Planning Option*** Set wpisujemy ***przerwy_na_blokach*** a następnie klikamy ***Add/Modify*** i ***Save***. Teraz chcemy aby opcja „***przerwy_na_blokach***" został użyta w naszym szablonie nazwanym wcześniej „***chip***". W tym celu na otwartym oknie ***Synthesize Power Plan*** upewniamy się, że jest wybrana opcja ***Design*** oraz nazwa chip i klikamy ***open*** - . Ponownie otwiera się okno edycji i w polu ***Power Planing Option Set*** wybieramy czerwoną strzałką ***przerwy_na_blokach***. Powinniśmy otrzymać okno jak na poniższym rysunku. Klikamy ***Save***.



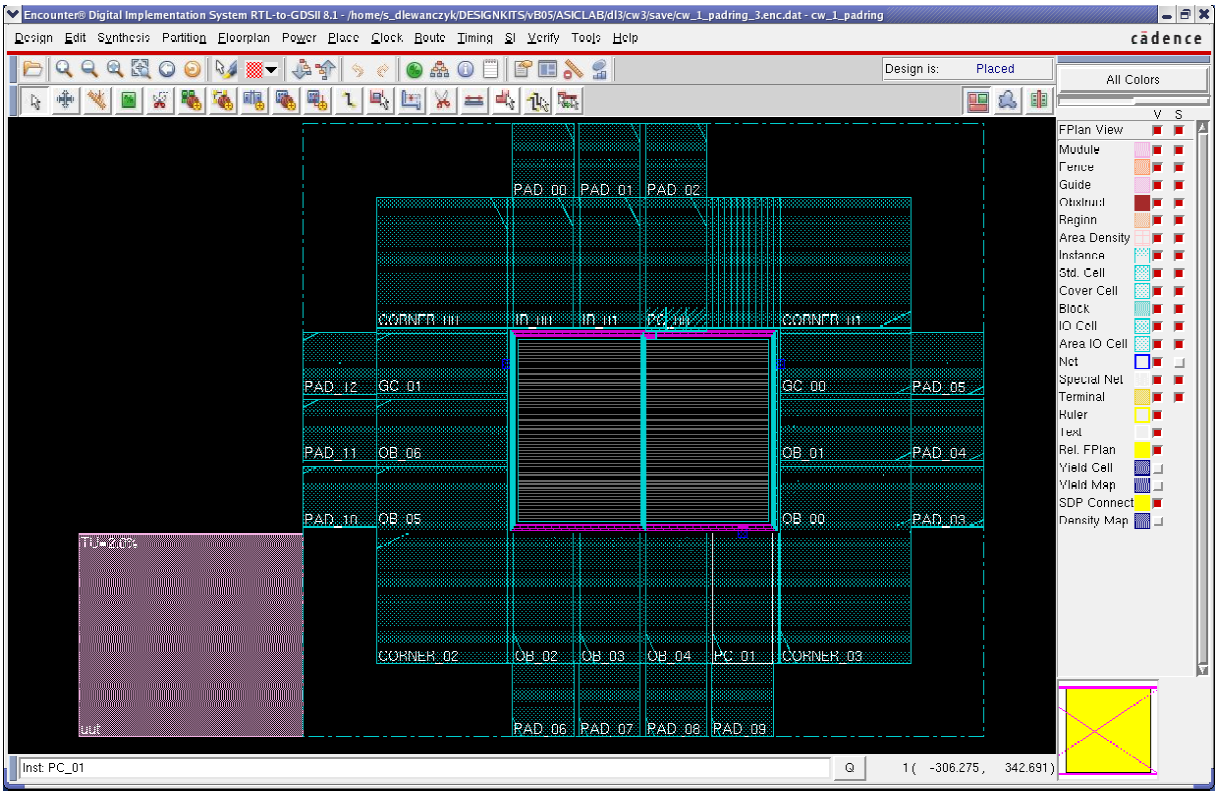
Następnie ustawiamy ostatnie parametry poprzez kliknięcie na ikonę . Otwiera się okno w którym najpierw wybieramy LKM projekt ***cw_1:core*** a następne wypełniamy parametry jak na rysunku poniżej i klikamy ***OK***.



W ten sposób został zakończony etap przygotowywania szablonu. Oby go zastosować w oknie ***Synthesize Power Plan*** (które cały czas powinno być otwarte) klikamy ***Apply*** i ***OK***. Po tym kroku w obszarze roboczym powinien pojawić się ring zasilający oraz paski połączeniowe. Następnie wykonujemy trasowanie połączeń linii zasilających poleceniem menu ***Route/ Special Route***.



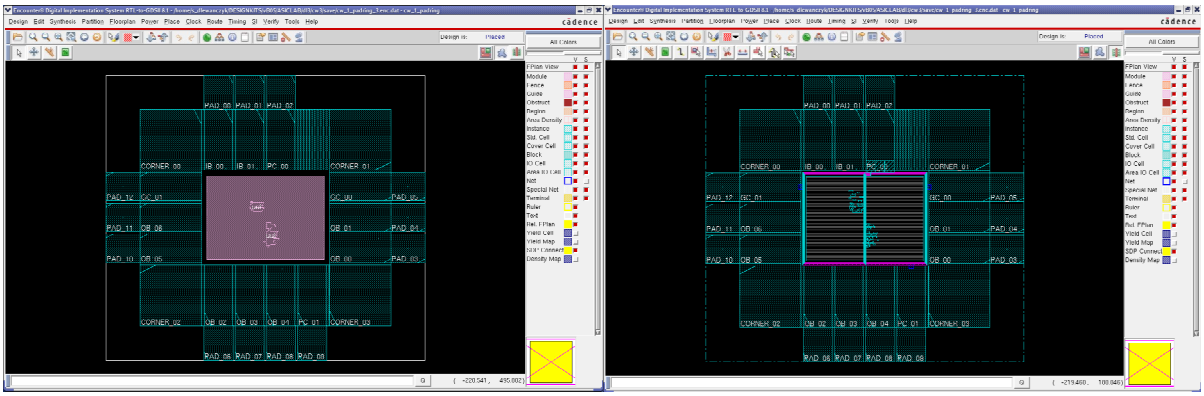
W oknie konfiguracji zaznaczamy wszystkie ***Pad Pins*** i ***Pad Rings*** i klikamy ***OK***, powinniśmy otrzymać wynik jak na poniższym rysunku.



W tym miejscu można wykonać pierwsze weryfikacje poprawności dotychczasowego projektu. Testowanie poprawności połączeń zasilania (specjalnych) wykonuje się poprzez menu ***Verify/ Verify Connectivity*** a następnie zaznaczenie ***Special Only*** i odznaczenia ***Antenna***. Klikamy ***OK*** i w oknie konsoli powinniśmy otrzymać raport (powinniśmy otrzymać ostrzeżenia tylko o niepodłączonych jeszcze sieciach buforów). Podobnie można sprawdzić geometrię, menu ***Verify/ Verify Geometry*** i ***OK***.

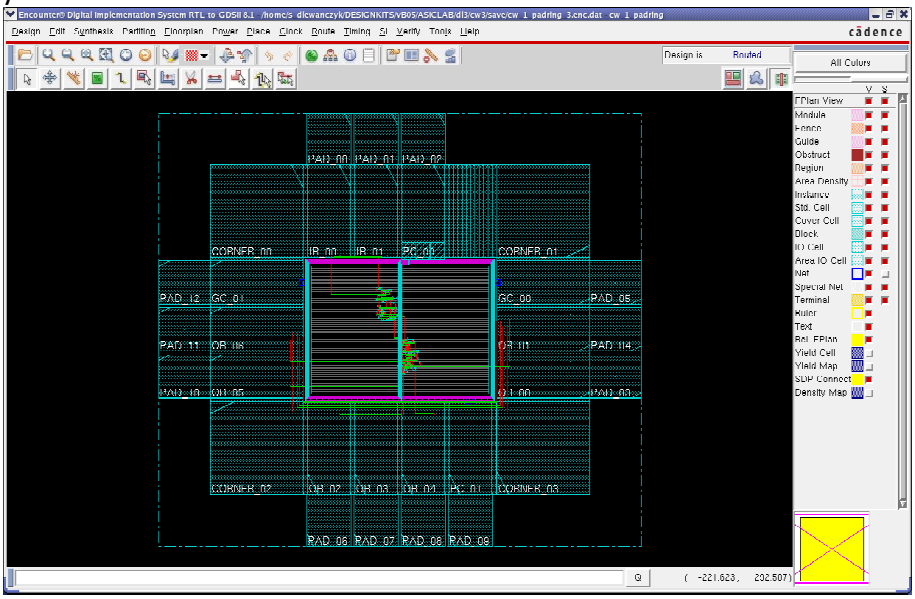
4) Rozmieszczanie komórek standardowych (ang. placement)

W naszym projekcie jest już wstępnie wykonany projekt kształtu rdzenia i planowana jego utylizacja. W celu rozmieszenia komórek standardowych dokonujemy najpierw zapis ustawień. Wybieramy menu ***Place/ Specify/ Placement Blockage*** i zaznaczamy ***M1***, ***M2*** i ***M8*** co spowoduje, że pod paskami zasilającymi program nie będzie rozmieszczał komórek. Następnie wybieramy menu ***Place/ Standard Cells..*** i powinno otworzyć się okno ***Place***. Zaznaczamy ***Run Full Placement*** a następnie klikamy LKM na ***Mode***. Otwiera się okno dodatkowych ustawień, w polu ***List of Modes*** wybieramy z ***Placement*** a następnie w zakładce ***Placement*** możemy wybrać tryby. Dla naszych celów pozostawiamy opcje domyślnie klikamy ***OK***. Rozpoczyna się rozmieszczanie elementów i po kilkunastu sekundach nasz projekt powinien być rozmieszczony. Po rozmieszczeniu komórek przyciskami  można przełączać widoki. Środkowy widok „Ameboa” jest przedstawieniem granic pomiędzy poszczególnymi blokami projektu hierarchicznego, widok „Physical” przedstawia rzeczywisty kształt ścieżek i komórek standardowych. Po procesie rozmieszczania komórek obszary robocze w widokach „Ameboa” i Physical powinny wyglądać podobnie jak to zaprezentowano poniżej.



5) Trasowanie połączeń (ang. routing)

Do trasowania połączeń zasilających używany jest program ***Sroute***. Połączenia standardowe można wykonać używając ***Traial Route*** lub ***Nano Route***. Pierwszy z ruterów jest ruterem prostym mogącym sobie poradzić z niewielkimi projektami, natomiast drugi jest złożonym ruterem na bieżąco optymalizującym połączenia. Dla celów naszego ćwiczenia wykorzystamy ***Trial Route*** wybierając z menu ***Route/ Trial Route*** a następnie ***OK***. To powinno skutkować wykonaniem połączeń pomiędzy komórkami standardowymi co można zaobserwować w widoku „Physical".

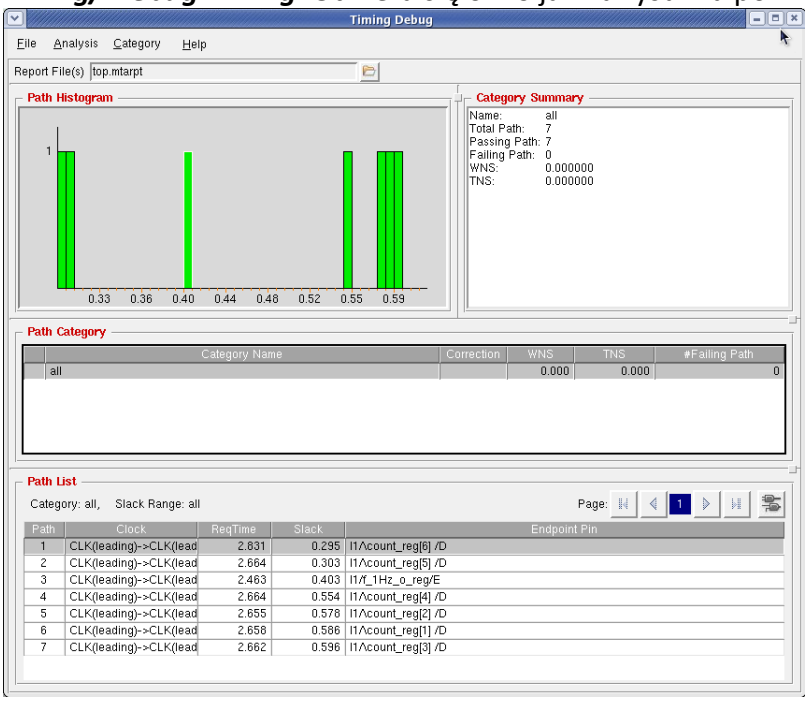


6) Ekstrakcja pojemności i rezystancji ścieżek, generacja modelu obciążeniowego oraz obliczanie opóźnień

Wykonywanie estymacji czasów propagacji oraz ich analiza jest kilkuetapowa. Ekstrakcja RC i generacja modelu obciążeniowego to etapy wstępne do obliczenia opóźnień. Ekstrakcja RC wykonywana jest poprzez menu ***Timing/ Extract RC*** , tu zaznaczmy wszystkie pliki z nazwą domyślą i kilkamy ***OK***. Następnie wykonujemy generację modelu obciążeniowego menu ***Timing/ Generate/ Wire-Load Model*** pozostawiamy opcje domyślne i klikamy ***OK***. Kolejnym krokiem jest obliczenie opóźnień, wykonujemy to poprzez menu ***Timing/ Calculate Delay***, ponieważ jeszcze nie było generacji drzewa zegarowego zaznaczmy ***Ideal Clock*** i następnie klikamy ***OK***. To kończy obliczenia opóźnień, które są zapisane do pliku typu „.sdf".

7) Analiza czasów ustalania (ang. setup)

Analizator czasowy wykorzystywany w programie ***SOC Encounter*** jest analizatorem typu ***MMMC*** (ang. multi – mode, multi - conditions) co oznacza, że można za jego pomocą analizować pracę projektowanego układu w kilku trybach jego pracy (np. normalna praca, praca DFT, praca w trybie o zmniejszonym poborze mocy i.t.p.) i w różnych warunkach zarówno zewnętrznego środowiska (temperatura pracy i napięcie zasilające) jak i dla różnych przebiegów procesu technologicznego (proces szybki, wolny lub typowy). Projektowany w ramach niniejszego laboratorium układ licznika – dekodera ma tylko jeden tryb pracy, natomiast możemy go analizować w różnych warunkach pracy i dla różnych przebiegów technologicznych. Określenie trybów i warunków symulacji wykonuje się poprzez menu ***Timing/ Analysys Condition/Specify Operating Condition/ PVT***. Są tam dostępne 2 zakładki ***min*** i ***max*** dla których podświetlone są warunki analizy. Klikamy ***OK*** i przechodzimy do analizy czasowej poprzez menu ***Timing/ Analyze Timing...*** tu wybieramy ***Pre-CTS*** (analiza przed generacją drzewa zegarowego) i ***Setup*** (analiza czasów ustalania) i kilkamy ***OK***. Po wykonaniu analizy w katalogu ***timingReports*** mamy stworzone pliki tekstowe z raportami analizy czasowej. Można te pliki przejrzeć niezależnie od systemu ***SOC Encounter*** ale lepiej jest użyć menu ***Timing/ Debug Timing***. Otwiera się okno jak na rysunku poniżej.



Bardzo ważne są dwie wartości oznaczane skrótami ***WNS*** (ang. worst negative slack) i ***TNS*** (ang. total negative slack). Wartości równe 0 dla parametrów ***WNS*** i ***TNS*** oznaczają, że zaprojektowany układ spełnia ograniczenia czasowe wczytane z pliku SDC (który był wcześniej wygenerowany w syntezie na podstawie ograniczeń wprowadzonych przez projektanta). ***WNS*** jest największym niespełnieniem ograniczeń czasowych z pliku ***SDC*** natomiast ***TNS*** jest sumą wszystkich niespełnień czasowych. Na histogramie prezentowane są liczby ścieżek po stronie dodatniej i ujemnej, przy czym na osi poziomej jest wartość niespełnienia czasowego. Wszystkie słupki z ujemnymi czasami oznaczają niespełnienie wymagań czasowych i są automatycznie kolorowane na czerwono. W polu ***Path List*** okna ***Timing Debug*** można wybrać dowolną ścieżkę i poprzez podwójne kliknięcie prześledzić dokładnie jej właściwości. W oknie głównym ***SOC Encounter*** obszar roboczy dzieli się na 2 części, górny widok topografii układu i dolny w którym można wybierać ścieżki i jej segmenty. Wybrane elementy z dolnego okna są automatycznie podświetlane w oknie

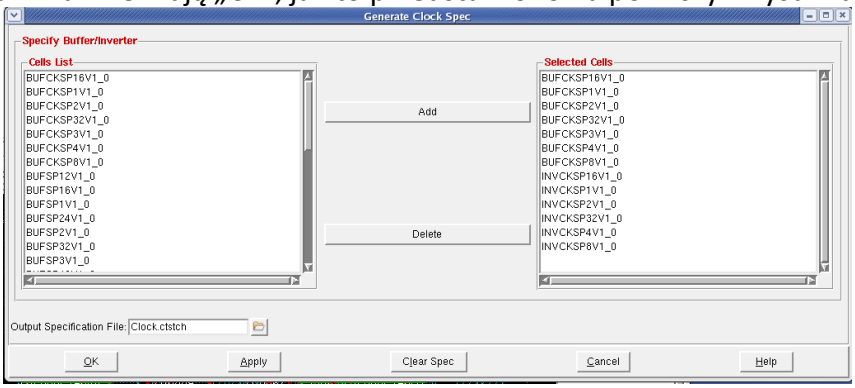
widoku. Możliwe jest przeglądanie zaznaczonych ścieżek we wszystkich rodzajach widoków ale widok „Ameboa" umożliwia najbardziej wyraźne śledzenie ścieżki. Znak O na ścieżce oznacza wyjście komórki standardowej natomiast znak X oznacza wejście komórki standardowej.

8) Wykonanie optymalizacji czasowej w celu naprawy czasów ustalania (ang. setup)

W przypadku gdy projekt nie spełnia wymagań czasowych można wykonać optymalizację czasową. Optymalizacja czasowa polega zazwyczaj na wstawianiu buforów i inwerterów w ścieżki sygnałowe, zmiany siły wyjść komórek standardowych oraz klonowanie instancji komórek standardowych. Z tego względu dobrze jest planować układ z utylizacją mniejszą niż 100% bo inaczej tracimy swobodę optymalizacji. Optymalizacja czasowa wykonywana jest poprzez menu ***Timing/ Optimize***, następnie należy wybrać ***Pre-CTS***, i ***Setup*** po kliknięciu na ***Mode*** i wybraniu ***Optimization*** można ustalić dodatkowe parametry np. duży wysiłek ***Effort High***. Następnie kilkamy ***OK*** i czekamy na wykonanie optymalizacji. Po optymalizacji możemy ponownie dokonać analizy opóźnień poprzez polecenie ***Timing/ Debug Timing***.

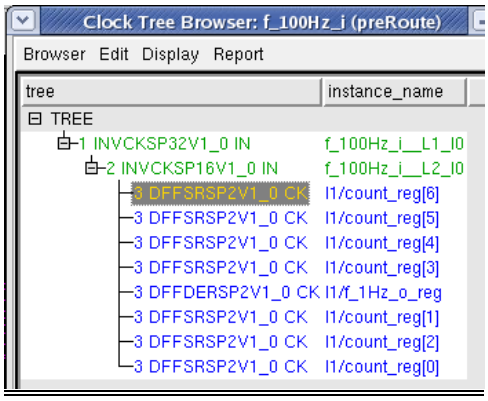
9) Generacja drzewa zegarowego

Do obecnej chwili w projekcie nie było poprowadzonej ścieżki zegarowej a wszelkie symulacje czasowe zakładały dojście zegara do wszystkich przerzutników bez jakichkolwiek opóźnień. Takie założenie powoduje pogorszenie możliwości zachowania czasów ustalania natomiast poprawę czasów podtrzymania (ang. hold). W celu utworzenia drzewa zegarowego wybieramy z menu ***Clock/ Design Clock***. Otwiera się okno ***Synthesize Clock Tree*** którym klikamy na ***Gen Spec...*** Otwiera się kolejne okno w którym musimy wybrać inwertery i bufory, które w nazwie mają „CK", jak to przedstawiono na poniższym rysunku.



Są to specjalnie zaprojektowane komórki o wyrównanych czasach narastania i opadania sygnału na ich wyjściu i szczególnie przydatne dla prowadzenia sygnału zegara. Klikamy ***OK.*** i wracamy do poprzedniego okna. Po naciśnięciu ***Mode*** a następnie ***CTS*** można ustalić dodatkowe parametry generacji drzewa zegarowego. Tu pozostawiamy domyślne wpisy, klikamy ***Cancel*** a następnie w oknie ***Synthesize Clock Tree*** klikamy na ***Apply*** i ***OK***. To powoduje wygenerowanie drzewa zegarowego a w katalogu ***clock_report*** utworzonych jest szereg plików tekstowych z raportami dotyczącymi utworzonego właśnie drzewa zegarowego. Po wygenerowaniu drzewa zegarowego można oglądnąć rezultaty za pomocą ***Clock/ Display/ Display Clock Tree***, zaznaczamy ***All Clocks***, ***Pre-Route*** oraz ***Display Clock Phase Delay***. To spowoduje różne zakolorowanie instancji do których dochodzą sygnały zegara z różnymi opóźnieniami (jeśli opóźnienia pomiędzy poszczególnymi instancjami będą duże, jeśli nie wszystkie zostaną zakolorowane jednakowo na niebiesko). Można również zobaczyć drzewo zegarowe poprzez menu ***Clock/ Clock Tree Browser***. Wybieramy nazwę sygnału zegara ***f_100Hz_i***, następnie ***Select***, ***Pre-Route*** i ***Apply***. Otwiera się okno jak na

rysunku poniżej, w którym można zaobserwować drogę sygnału zegara od wejścia do poszczególnych przerzutników.



10) Analiza czasów podtrzymania (ang. hold)

Tą analizę wykonujemy podobnie do poprzedniej, zaznaczamy tylko opcję czasów hold zamiast setup. Wybieramy menu ***Timing/ Analyze Timing***, wybieramy opcję ***Post-CTS*** oraz ***Hold*** i klikamy ***OK***. Podsumowanie analizy mamy na konsoli a szczegóły można przebadać poleceniem ***Timing/ Debug Timing*** (tu, po otworzeniu się okna ***Timing Debug*** trzeba ponownie załadować plik analizy poleceniem ***File/Load Report File***).

11) Wykonanie optymalizacji czasowej w celu naprawy czasów podtrzymania i/lub ustalania

Podobnie jak poprzednio, w przypadku gdy projekt nie spełnia wymagań czasowych można wykonać optymalizację czasową. Klikamy menu ***Timing/ Optimize***, następnie należy wybrać ***Post-CTS***, i ***Setup*** oraz ***Hold*** po kliknięciu na ***Mode*** i wybraniu ***Optimization*** można ustalić dodatkowe parametry np. duży wysiłek ***Effort High***. Następnie kilkamy ***OK*** i czekamy na wykonanie optymalizacji. Po optymalizacji możemy ponownie dokonać analizy i debugowania opóźnień poprzez polecenia ***Timing/ Analyze Timing*** oraz ***Debug Timing***.

12) Zapobieganie i analiza przesłuchów(ang. SI – signal integrity)

Każdy projekt układu cyfrowego niskoskalowanego musi być przeanalizowany pod kontem przesłuchów międzyścieżkowych powodujących zmianę rzeczywistych opóźnień występujących na ścieżkach projektowanego układu. Najpierw zostanie wykonane trasowanie połączeń z uwzględnieniem naprawy efektu anteny, SI oraz poprawy parametrów czasowych. Wybieramy menu ***Route/ NanoRoute/ Route...*** a następnie zaznaczamy opcje ***Fix Antenna***, ***Timing Driven*** oraz ***SI Driven*** i klikamy ***OK***. W celu analizy przesłuchów mysi być wykonana szczegółowa ekstrakcja RC. Wybieramy menu ***Timing/ Analysis Condition/ Specify RC Extraction Mode***, wybieramy ***Detail*** oraz ***Coupled RC*** i klikamy ***OK***. Następnie w klikamy ***SI/ Run CeltIC Crosstalk Analysis...***, wybieramy ***Mode*** - ***Naitve***, ***Effort*** - ***High*** oraz ***Process*** - ***130nm*** i klikamy ***OK***. W konsoli można obserwować pracę analizatora przesłuchów, na koniec otrzymujemy podsumowanie i jeśli lista ***Noise nets to be fixed:*** jest niepusta występuje w projekcie problem niespełnienia parametrów czasowych ze względu na przesłuchy. W takim przypadku można oglądnąć problematyczne sieci poprze menu ***SI/ Display Noise Net*** oraz dokonać naprawy przesłuchów poprzez menu ***SI/ Fix Crosstalk***.

13) Analiza poboru mocy

Można oszacować moc pobieraną przez układ w sposób szczegółowy lub uproszczony, statystyczny bazujący na prawdopodobieństwie przełączania sieci. Do pierwszego rodzaju analizy niezbędna jest wcześniej wykonana szczegółowa symulacja czasowa na sygnałach dokładnie takich jak później spodziewanie w rzeczywistej pracy układu. Uproszczona analiza mocowa zakłada natomiast przybliżone prawdopodobieństwo przełączania się sieci w stosunku do sygnału zegara. W naszym przypadku wykonana zostanie analiza uproszczona. Najpierw ustawiamy parametry analizy poprzez menu ***Power/ Power Analysis/ Set Power Analysis Mode***. Pozostawiamy opcje domyślne i klikamy ***OK***. Następnie wybieramy menu ***Power/ Power Analysis/ Run Power Analysis...*** W zakładce ***Basic*** wpisujemy ***Input Activity - 0.1***, ***Dominant Frequency - 300MHz*** i ***Flop Activity - 0.1***, następnie w zakładce ***Activity*** wpisujemy ***Global Activity - 0.3*** i klikamy ***Apply*** i ***OK***. Zostanie wykonana analiza mocowa a na konsoli zostaną przedstawione jej wyniki. Równocześnie bardziej szczegółowe wyniki są umieszczane w pliku tekstowym ***cw_1.rpt***. Domyślną jednostką mocy jest mW.

Załącznik: Wzór protokołu:

Ćwiczenie 1 – laboratorium PASIC, studia inżynierskie, sem. 7		
L.p.	**Nazwa / opis**	**Wartość**
1	Data wykonania	
2	Nazwisko i imię	
3	Zrzut obszaru roboczego programu SOC Encounter po wykonaniu analizy poboru mocy w widoku „Physical”	
4	Całkowity pobór mocy zaprojektowanego układu [mW]	

Ćwiczenie opracowali:
Bogdan Pankiewicz i Dawid Lewańczyk,
Gdańsk, listopad 2010

Powyższy materiał dydaktyczny przygotowano z wykorzystaniem systemu pomocy CDNSHELP firmy Cadence, oraz prac dyplomowych magisterskich realizowanych w Katedrze Systemów Mikroelektronicznych, ETI, PG w latach 2009 - 2010.