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 techniki komórek standardowych. Jako narzędzie projektowe wykorzystany zostanie pakiet CADENCE wydanie 2017/2018. Biblioteka komórek standardowych to biblioteka zaprojektowana dla technologii UMC 130nm przez dyplomanta w ramach pracy magisterskiej. Ćwiczenia muszą być wykonane kolejno, gdyż zakończenie poprzedniego ćwiczenia daje umiejętności niezbędne do wykonania kolejnego. Wykonanie ćwiczenia rozszerzonego umożliwia dodanie wyprowadzeń I/O do rdzenia układu scalonego.

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) Konto i logowanie.

Każdy ze studentów zajęć otrzyma konto, którego login (nazwa użytkownika) ma następujący format:

`s_xyyy`

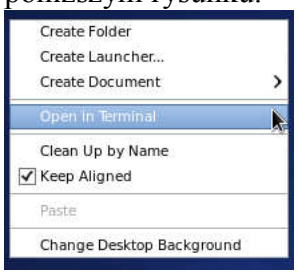
gdzie: **x** – pierwsza litera imienia, **yyy** – nazwisko bez polskich znaków diakrytycznych. Po włączeniu komputera w sali 308 lub 337 należy wybrać uruchomienie systemu LINUX a następnie zalogować się z użyciem powyższego loginu i hasła przydzielonego przez prowadzącego.

UWAGA: login należy wpisywać wyłącznie małymi literami w przeciwnym razie system pozwoli na zalogowanie ale błędnie zostaną przydzielone prawa i nie będzie można wykonać ćwiczeń laboratoryjnych.

Po pierwszym logowaniu należy zmienić hasło z przydzielonego odgórnie na własne. Aby to zrobić należy w terminalu wydać polecenie:

`passwd`

a następnie postępować zgodnie z poleceniami przekazywanymi w terminalu. Sam terminal uruchamia się poprzez przyciśnięcie prawego klawisza myszki i wybranie polecenia **Open in Terminal** jak na poniższym rysunku.



System kont w laboratorium jest wspólny dla wszystkich komputerów, tj. tym samym loginem i hasłem można posługiwać się na wszystkich komputerach. System plików użytkownika jest natomiast unikalny na każdym z komputerów. Z tego względu należy na każdych zajęciach zajmować to samo stanowisko lub też kopiować pliki pomiędzy komputerami z wykorzystaniem polecenia **scp**. Przykładowo jeśli z komputera **fpgalab5** z konta użytkownika **s_xyyy** chcemy skopiować wszystkie pliki z katalogu **lab1** na bieżący komputer do bieżącego katalogu należy w terminalu wydać polecenie:

scp -r s_xyyy@fpgalab5:/home2/s_xyyy/lab1 .

Należy zwrócić uwagę na to, że znak kropki na końcu polecenia oznacza katalog bieżący. System plików użytkowników na wszystkich komputerach w sali 308 umieszczony jest w katalogu ***/home2***.

3) Podstawowe informacje o pakiecie CADENCE.

Pakiet oprogramowania Cadence zainstalowany jest w katalogu: ***/eda/cadence/2017_2018***. Aby było można z niego skorzystać należy w terminalu uruchomić polecenie, które konfiguruje środowisko:

source /eda/cadence/2017-2018/digital_c64.csh

Następnie należy utworzyć katalog roboczy i wejść do niego i tam uruchamiać niezbędne pakiety oprogramowania. Przejście do katalogu głównego użytkownika (z dowolnego innego katalogu) wykonuje się poleceniem:

cd

utworzenie katalogu (o nazwie ***test_lab1***):

mkdir test_lab1

przejście do katalogu:

cd test_lab1

uruchomienie programu (np. symulacji cyfrowej)

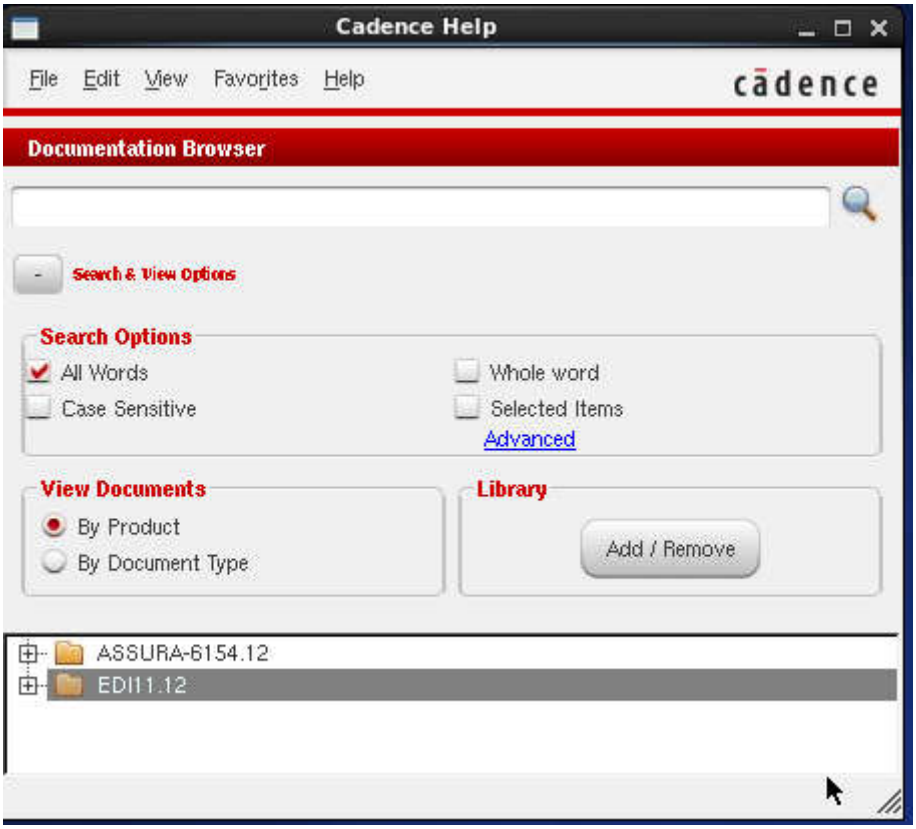
nclaunch -new &

Znak ”&” na końcu polecenia wydawanego w terminalu oznacza, że dane polecenie uruchomione jest w tle a terminal został zwolniony i można go wykorzystać do wydawania kolejnych poleceń.

Uruchomienie dokumentacji pakietu Cadence następuje poprzez wydanie polecenia:

cdnshelp &

Pomoc Cadence otwiera się w oknie graficznym, jak to na przykład przedstawiono na poniższym rysunku:



W dolnej części okna podane są pakiety pomocy które zostały wczytane. W powyższym przypadku jest to pakiet ASSURA oraz EDI. Jeśli chcemy wczytać pomoc dodatkowego pakietu należy wybrać myszką

przycisk ***Add/Remove*** w oknie ***Library*** a następnie wybrać przycisk ***Add..*** i z listy pakietów wybrać stosowny katalog zgodnie z formatem poniżej:

/eda/cadence/2017-2018/RHELx86/(nazwa szukanego pakietu)/doc.

Po wczytaniu plików pomocy dotyczących pożądanego pakietu można je wyświetlać sortując wg produktu lub typu dokumentu a następnie poprzez rozwijanie znaku + w dolnym oknie można wybrać pożądany element. Podwójne klikniecie w wybrany temat pomocy otwiera boczne okno z treścią pomocy.

Alternatywnie, poprzez kliknięcie ikony  na pasku zadań, treść pomocy można otworzyć jako dokument PDF. W przypadku problemów z otworzeniem pliku pdf należy skonfigurować aplikację, która służy do wyświetlania tego rodzaju plików poprzez wybranie menu ***Edit/Settings*** zakładka ***Applications***, klikamy przycisk ***Add*** a następnie w kolumnie ***Ext*** wpisujemy ***pdf*** a w kolumnie ***Application*** wpisujemy ***firefox***. Po naciśnięciu przycisku OK ponowne wybranie ikony  powinno spowodować wyświetlenie wybranego pliku w przeglądarce ***Mozilla Firefox***. Pomoc jest w języku angielskim.

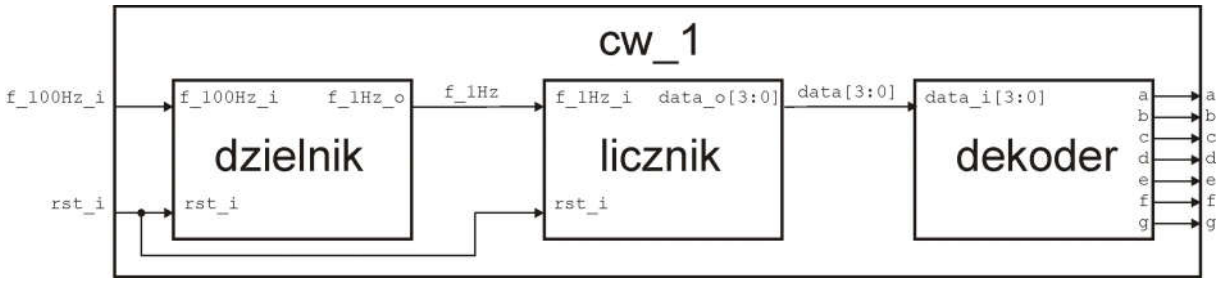
Laboratorium do przedmiotu „Projektowanie Układów ASIC” dla studiów inżynierskich
Ćwiczenie 1: Opis VHDL i symulacja prostego projektu układu cyfrowego

Nieobowiązkowe ale zalecane: Dla osób chcących głębiej poznać symulator Incisive/SimVision można skorzystać z pomocy Cadence uruchamiając program ***cdnshelp***. Następnie należy wybrać pakiet ***Incisive_15.20*** a następnie kolejno należy wybierać:

View Documents – By Document Type
rozwinąć znak ”+” w dokumentach z kategorii ***15.20***
rozwinąć znak ”+” w dokumentach z kategorii ***Getting Started***
podwójnie kliknąć na ***Tutorial for Mixed Verilog and VHDL with SimVision***

W tym tutorialu zawarty jest przykład projektu wyjaśniający obsługę symulatora Incisive/SimVision.

W ramach ćwiczenia należy wykonać opis w języku VHDL (lub Verilog) i symulację prostego układu cyfrowego składającego się z 4 następujących bloków entity:
cw_1 – blok integrujący w sobie 3 pozostałe 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*** 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. Każdy z bloków należy zapisać w osobnym pliku tekstowym o nazwie takiej jak nazwa bloku z rozszerzeniem **.vhdl**. (lub **.v** dla plików Verilog). Do wykonania opisu należy wykorzystać dowolny edytor tekstowy, np. ***gedit***, ***vi*** lub ***emacs***.

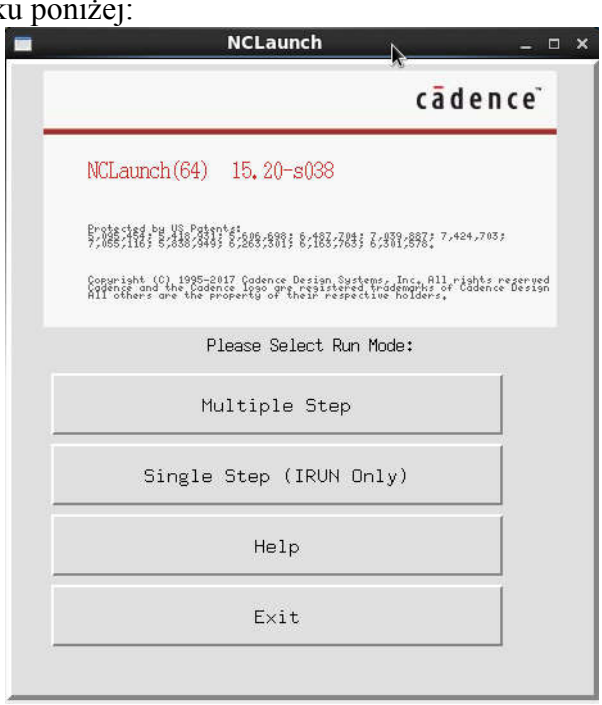


W celu sprawdzenia poprawności składni należy przeprowadzić kompilację pliku poleceniem:
ncvhdl –messages –V93 nazwa_pliku.vhdl
lub dla kodu Verilog:
ncverilog nazwa_pliku.v

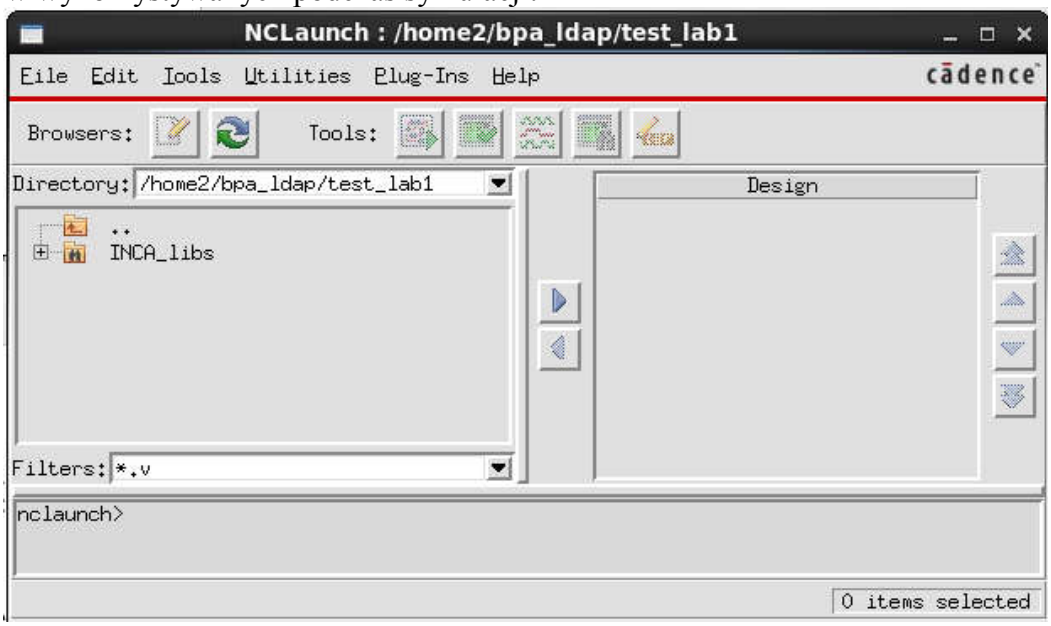
W przypadku błędu składni pojawi się komunikat informujący o linii i kolumnie w której jest prawdopodobny błąd. W przypadku wielu błędów poprawki należy rozpoczynać od pierwszego błędu!
Po doprowadzeniu do poprawności składniowej wszystkich plików należy uruchomić symulację poleceniem

nclaunch -new &

Pojawi się okno jak na rysunku poniżej:

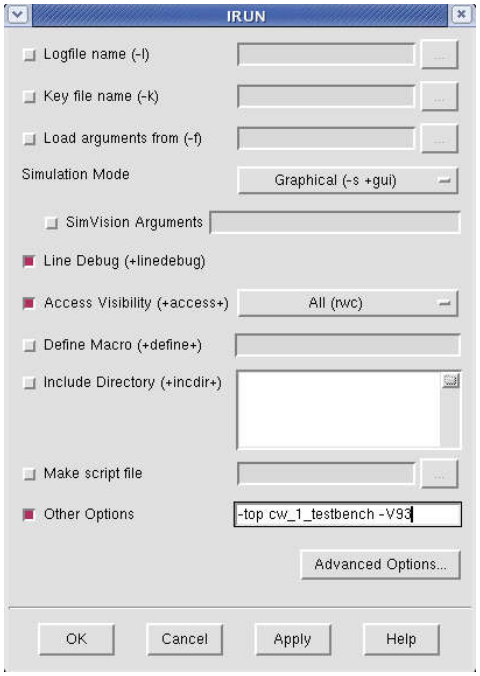


W tym momencie można wybrać dwa rodzaje symulacji: wykonywana etapowo (***Multiple Step***) lub jednokorokowo (***IRUN Only***). Dla celów naszego ćwiczenia wybieramy symulację jednokrokową. Po naciśnięciu pola ***Single Step*** powinno pojawić się okno jak na rysunku poniżej, przedstawiające zestaw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

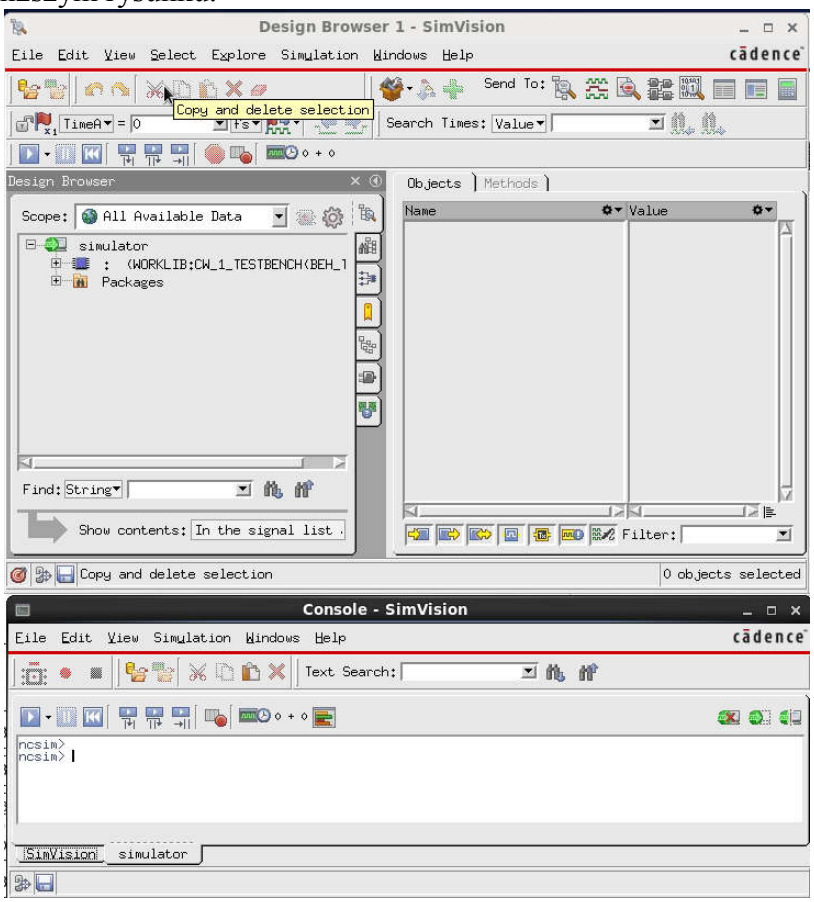


Domyślnie ustawiony jest filtr wyświetlania plików Verilog (z rozszerzeniem .v). W celu przeglądania innych plików należy w polu ***Filters*** zmienić wpis na .vhdl. Następnie należy dodać wszystkie pliki VHDL do projektu (zaznaczyć myszką i kliknąć strzałkę w prawo ). Następnie wybieramy menu ***Tools/IRUN*** i zaznaczamy pole ***Other Options*** i wpisujemy obok ***-top cw_1_testbench -***

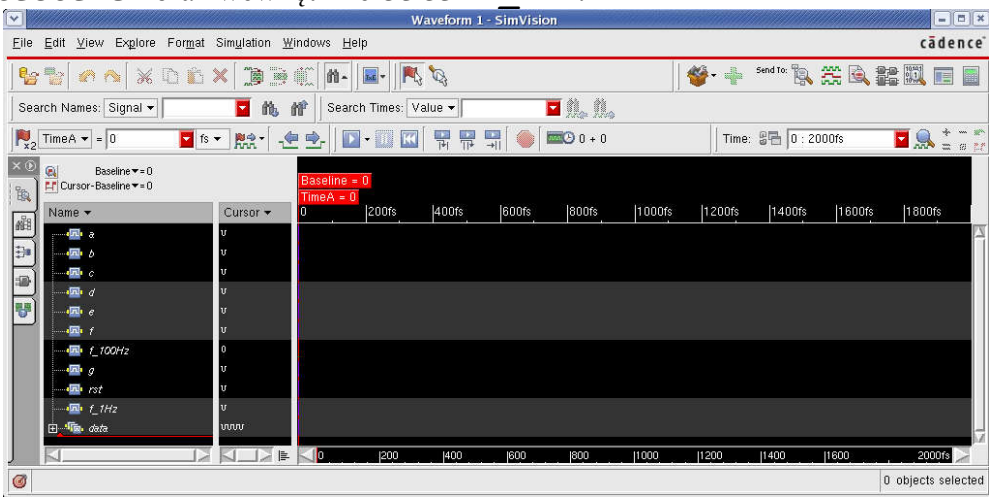
V93. Dopisanie tej opcji wskazuje symulatorowi który z bloków jest najwyżej w hierarchii projektowej oraz włącza składnię języka VHDL w wersji 1993. Po tym kroku powinniśmy otrzymać okno jak na rysunku poniżej.



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/Source Browser***, 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 ***gedit +%L %F*** uruchamia edytor ***gedit***,
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 bloku głównego, czas symulacji 15s.	
4	Plik: cw_1_testbench.vhdl	
5	Plik: cw_1.vhdl	
6	Plik: dzielnik.vhdl	
7	Plik: licznik.vhdl	
8	Plik: dekoder.vhdl	

Ćwiczenie opracował Bogdan Pankiewicz, Gdańsk, wrzesień 2010
Poprawki: październik 2011, wrzesień 2014, lipiec 2018

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

Pomoc dla pakietu GENUS 17.11 (synteza układów scalonych): dostęp do pliku pomocy użytkownika syntezera ***Genus*** jest dostępny w pliku PDF: ***/eda/cadence/2017_2018/RHELx86/GENUS_17.11.000/doc/genus_start/genus_start.pdf*** oraz ***genus_user_legacy/genus_user_legacy.pdf***. Alternatywnie można uruchomić pomoc poleceniem ***cdnshelp &*** a następnie wybrać bibliotekę GENUS 17.11.

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.

UWAGA:

Aby poniższe przykłady poleceń można było zastosować bezpośrednio bez potrzeby modyfikacji nazw ścieżek należy:

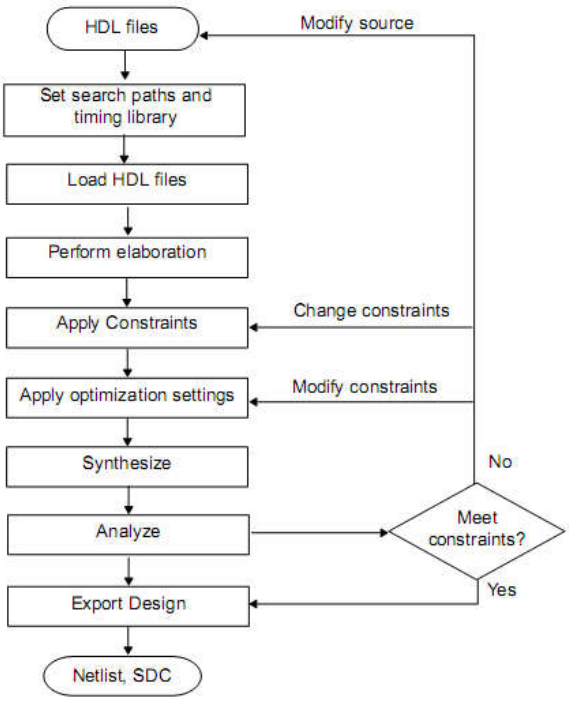
- utworzyć katalog ***lab2*** a następnie do niego wejść,
- utworzyć podkatalogi ***hdl***, ***rpt*** oraz ***out***,
- aby wykonać 2 punkty powyższe, otwieramy terminal i wykonujemy polecenia kolejno:

```
cd
mkdir lab2
cd lab2
mkdir hdl
mkdir rpt
mkdir out
```

- należy do podkatalogu ***hdl*** skopiować przygotowane w ćwiczeniu nr 1 pliki VHDL,
- uruchomienie programu ***Genus*** powinno być wykonane z katalogu ***lab2***.

Do syntezy, w ramach pakietu CADENCE 2017/2018 należy użyć programu ***Genus***. Uruchomienie tego programu następuje z poziomu terminala poprzez polecenie:
genus lub ***genus -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 ***Genus*** przedstawiony jest na rysunku poniżej (rysunek zaczerpnięty z dokumentacji tego oprogramowania). Przebieg takiego projektowania jest ogólny w tym sensie, że stanowi podstawę do przebiegów szczegółowych, przedstawione poniżej etapy syntezy zawsze się powtarzają. Przebiegi bardziej zawansowane umożliwiają np. wykonanie syntezy typu 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 przeszukiwania plików HDL i bibliotek. Domyślnym miejscem przeszukiwania jest katalog z którego został uruchomiony program Genus. Wszelkie ustawienia w programie wykonuje się poprzez nadanie obiektom określonych atrybutów (wartości). Polecenie ustawiające atrybuty to ***set_db***. W celu ustawienia ścieżki gdzie będą poszukiwanie pliki projektu ustawiany atrybut ***init_hdl_serch_path***. Np. jeśli zamierzamy nasze pliki źródłowe umieścić w podkatalogu ***hdl*** katalogu roboczego powinniśmy w programie ***Genus*** wpisać polecenie:

set_db init_hdl_search_path ./hdl

Warto ustawić ścieżkę pod którą poszukiwane będą biblioteki komórek standardowych:

set_db init_lib_search_path /techfiles_ldap/umc_ksmi_lab/lib

Aby sprawdzić wartość ustawionego wcześniej atrybutu można użyć polecenia

get_db nazwa_poszukiwanego_atrybutu.

Informacje dotyczące poszczególnych poleceń/obiektów/atrybutów można uzyskać poprzez:

help nazwa_polecenia np. ***help get_db***

help -attribute *lib

help -attribute *search_path

Na komputerach laboratorium w sali 308 katalog ***/techfiles_ldap*** 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_ldap/umc_ksmi_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.

W programie ***Genus*** 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 ***genus.cmd***, rejestruje również wszystkie komunikaty konsoli do domyślnego pliku ***genus.log***. Aby kolejnego dnia uruchomić wszystkie wcześniej wykonane polecenia wystarczy uruchomić ***include genus.cmdxx*** gdzie ***xx*** jest ostatnim numerem logu. Możliwe jest również wczytanie poleceń w czasie uruchamiania programu: ***genus -file plik_polecen***.

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_db library cells_typical.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_db 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.

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.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 ***Genus*** był uruchomiony z opcją ***-gui***) można przeglądać schemat blokowy projektu (prawy klawisz myszki na głównym module i wybranie opcji ***Schematic View (Module)/in New...)*** oraz jego kod HDL (naciśnięcie symbolu  w dowolnym oknie, wybranie opcji ***HDL Viewer*** a następnie wczytanie pliku poprzez przycisk). Przemieszczanie się po hierarchii projektu następuje poprzez rozwijane znakiem „+” menu w zakładce ***Design Browser***.

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 Genus,
- wczytanie pliku z ograniczeniami,
- wczytanie pliku z ograniczeniami w formacie SDC.

W ramach niniejszego ćwiczenia nałożone zostaną podstawowe ograniczenia czasowe. Najbardziej rozpowszechnione jest wczytywanie plików SDC, jednak proste ograniczenia stosowane w tym

ćwiczeniu zostaną wprowadzone bezpośredni z konsoli narzędzia Genus. Najpierw sprawdzamy używane w bibliotekach jednostki czasowe gdyż to one są używane do podawania czasów:

```
report_units
```

Następnie deklarowany jest sygnał zegara CLK wraz z jego wymaganym minimalnym okresem 2ns i połączeniem go portu zewnętrznego rozpoczynającego się nazwą **f_** :

```
create_clock -name CLK -period 3 [get_ports f_*]
```

Dodatkowo nadajemy sygnałowi zegara niepewność w czasie dotarcia do przerzutników (tzw. clock skew):

```
set_clock_uncertainty 0.005 CLK
```

Sygnałom wyjściowym i wejściowym również nadane są ograniczenia oczekiwanego wygenerowania lub dotarcia sygnałów, dla czasu setup:

```
set_output_delay -clock CLK -max 0.1 [get_ports {a b c d e f g}]
```

oraz dla czasu hold:

```
set_output_delay -clock CLK -min 0.05 [get_ports {a b c d e f g}]
```

:

Dla sygnału wejściowego resetu podawane są opóźnienia sygnału na wejściu w stosunku do sygnału zegara (podobnie jak poprzednio dla wymagań setup i hold):

```
set_input_delay -clock CLK -max 0.1 [get_ports rs*]
set_input_delay -clock CLK -min 0.05  [get_ports rs*]
```

Polecenie **get_ports** 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, np. **get_ports {a b c r* *}**. W nawiasach klamrowych można podać listę wyszukiwanych elementów rozdzielonych spacją i zawierających tzw. "wildcards" w miejsce pojedynczego wyszukania.

W kolejnym kroku wykonywana jest synteza. Przed wydaniem poleceń syntezy można poprzez zmianę atrybutów sterować przebiegiem pracy. W ramach niniejszego ćwiczenia chcemy zachować hierarchię projektu po syntezie. Jedną z metod jej uzyskania jest ustawienie niskiego wysiłku syntezera, ustaiamy to poprzez polecenia:

```
set_db syn_generic_effort low
set_db syn_map_effort low
```

Synteza przeprowadzana jest poleceniami:

```
syn_generic
syn_map
```

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 (najeżdżamy na wybrany moduł i prawym klawiszem myszki wybieramy polecenie **Schematic_View(module)/In New...**).

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
write_design -innovus cw_1
```

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. Ostatnie polecenie zapisuje projekt w formacie przeznaczonym do wczytania w oprogramowaniu implementacyjnym.

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_ldap/umc_ksmi_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 500MHz oraz przyspieszyć deaktywację resetu do czasu 5ns. 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 2 - laboratorium PASIC, studia inżynierskie, sem. 7		
L.p.	**Nazwa / opis**	**Wartość**
1	Data wykonania	
2	Nazwisko i imię	
3	Zrzut okna graficznego syntezera – po elaboracji i po syntezie, hierarchia główna projektu ***cw_1***	
4	Zrzut okna graficznego syntezera – po elaboracji i po syntezie, rozwinięty blok ***licznik***	
5	Zrzut okna graficznego syntezera – po elaboracji i po syntezie, rozwinięty blok ***dzielnik***	
6	Zrzut okna graficznego syntezera – po elaboracji i po syntezie, rozwinięty blok ***dekoder***	
7	Zrzut okna symulacji z umieszczonymi wszystkimi sygnałami bloku głównego oraz podzielonym zegarem wewnętrznym – musi być widoczne opóźnienie sygnałów wyjściowych w stosunku do zegara.	
8	Odczytana z rysunku powyżej wartość opóźnienia sygnału ***f_1Hz*** w stosunku do ***f_100Hz***	

Ćwiczenie opracował Bogdan Pankiewicz, Gdańsk, wrzesień 2010, wrzesień 2014, lipiec 2018

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

Pomoc dla pakietu ***Innovus 17.11 - implementacja:*** Dostęp do pliku pomocy użytkowania programu Innovus dostępny jest w pliku PDF ***/eda/cadence/2017_2018/RHELx86/INNOVUS_17.11.000/doc/innovusUG/innovusUS.pdf***. Alternatywnie można skorzystać z pełniejszego pakietu pomocy poprze uruchomienie programu ***cdnshelp*** i wybranie pakietu ***Innovus 17.11*** a następnie ***User Guide / Innovus User Guide***.

W ramach niniejszego ćwiczenia zostanie wykonana implementacja rdzenia układu scalonego bez dołączania wyprowadzeń 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. Dla celów niniejszego ćwiczenia zakładamy katalog lab3 (równoległy do katalogu lab2) poprzez polecenie ***mkdir lab3*** (w katalogu w którym jest widoczny katalog lab2) a następnie należy wejść do tego katalogu ***cd lab3***. Uruchomienie programu ***Innovus*** następuje po wydaniu polecenia ***innovus***. Uruchomiony program ma 2 okna – konsolę służącą do wydawania poleceń tekstowych jak również okno graficzne. Kolejność wykonania projektu składa się z 5 głównych etapów i jest następująca:

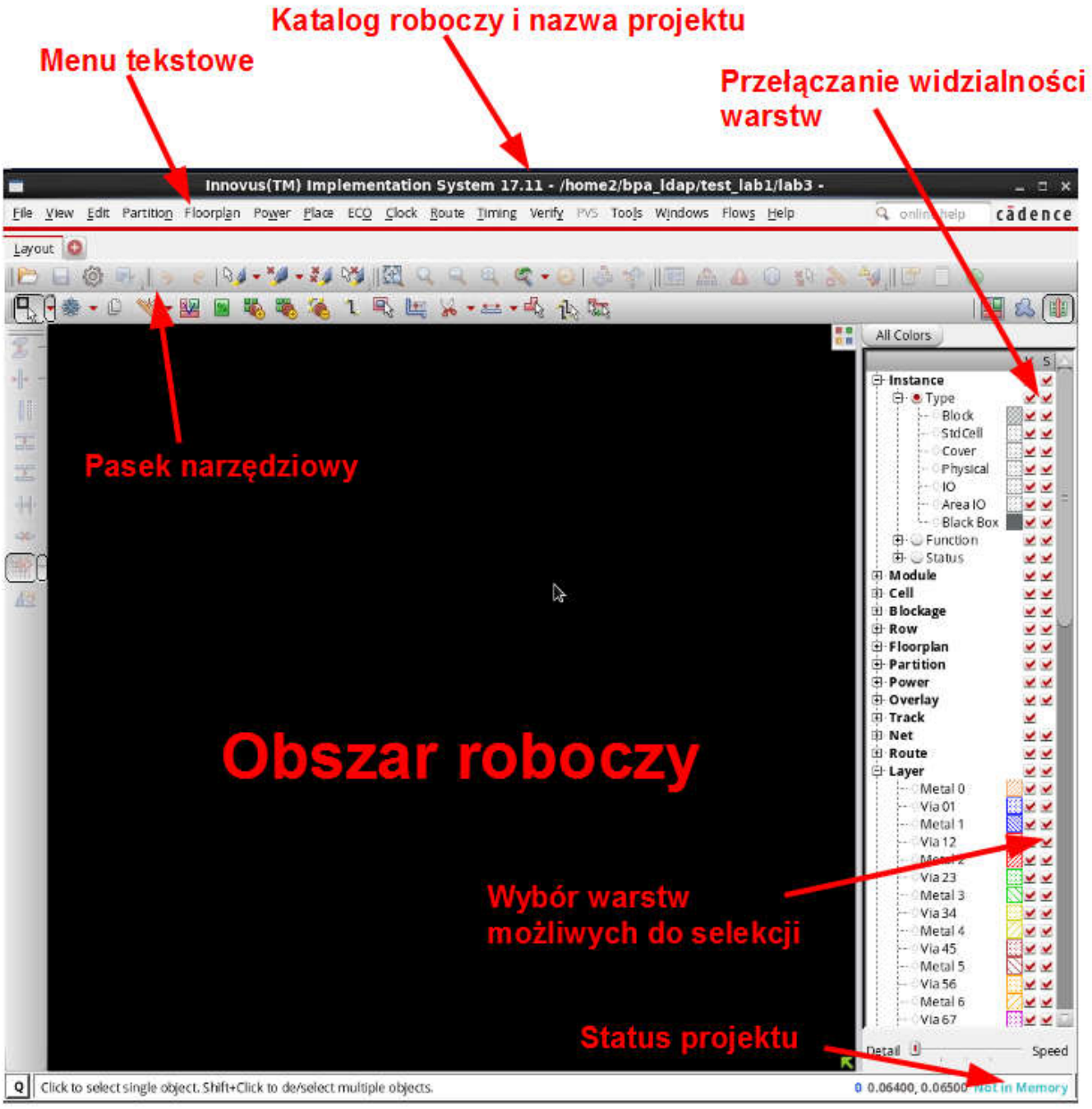
1) inicjalizacja projektu:
   - import wyników syntezy i bibliotek,
   - planowanie ułożenia elementów (Floorplaning),
   - ustawienie trybu pracy (Timing, Mode Setup),
2) implementacja przed utworzeniem ścieżki zegara (Pre-CTS Flow):
   - definicja łańcucha skanowania (Scan Definition)
   - wstawienie komórek JTAG i optymalizacja czasowa (JTAG placement and timing optimalisation),
3) implementacja po wstawieniu ścieżki zegarowej (Post-CTS Flow):
   - synteza drzewa zegarowego (Clock Tree Synthesis),
   - optymalizacja czasów hold (Post-CTS Hold Fixing),
4) implementacja po trasowaniu ścieżek (Postroute Flow):
   - trasowanie ścieżek (Routing),
   - naprawa czasów setup/hold po trasowaniu oraz testowanie i optymalizacja czasów związanych z przesłuchami (Postroute Base & SI Delays Setup/Hold Fixing),
5) Końcowe testowanie projektu (Signoff):
   - dodanie wypełnień metalu (Add Metal Fill),
   - weryfikacja fizyczna (Physical Verification),
   - testowanie czasowe (Timing Analysis).

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.

Pliki, które są potrzebne do wykonania implementacji obejmują:
- netlistę po syntezie w postaci pliku Verilog lub bazy OA (Open Acess),
- plik planu układu, można go stworzyć interaktywnie w Innovus lub zaimportować plik typu .fp lub DEF,
- plik specyfikacji drzewa zegarowego, jest generowany automatycznie z pliku .sdc,
- plik kolejności elementów łańcucha skanowania w formacie Tcl lub DEF,

- plik informacji o padach I/O,
- plik mapowania warstw GDS (do generacji kształtu masek przekazywanych do producenta),
- plik ograniczeń czasowych, tzw. plik .sdc,
- biblioteki czasowe w formacie .lib,
- biblioteki fizyczne w formacie LEF lub OA niezbędne do trasowania połączeń i rozmieszczania elementów,
- dodatkowe pliki do ekstrakcji takiej jak captable lub pliki Quantus QRC.

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



Przyciski muszki 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 – otworzenie formularza ***Attribube Editor***.

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

- klik wyśrodkowuje widziany obraz w punkcie kliknięcia.

Prawy klawisz myszki (oznaczany dalej w tekście skrótem PKM):

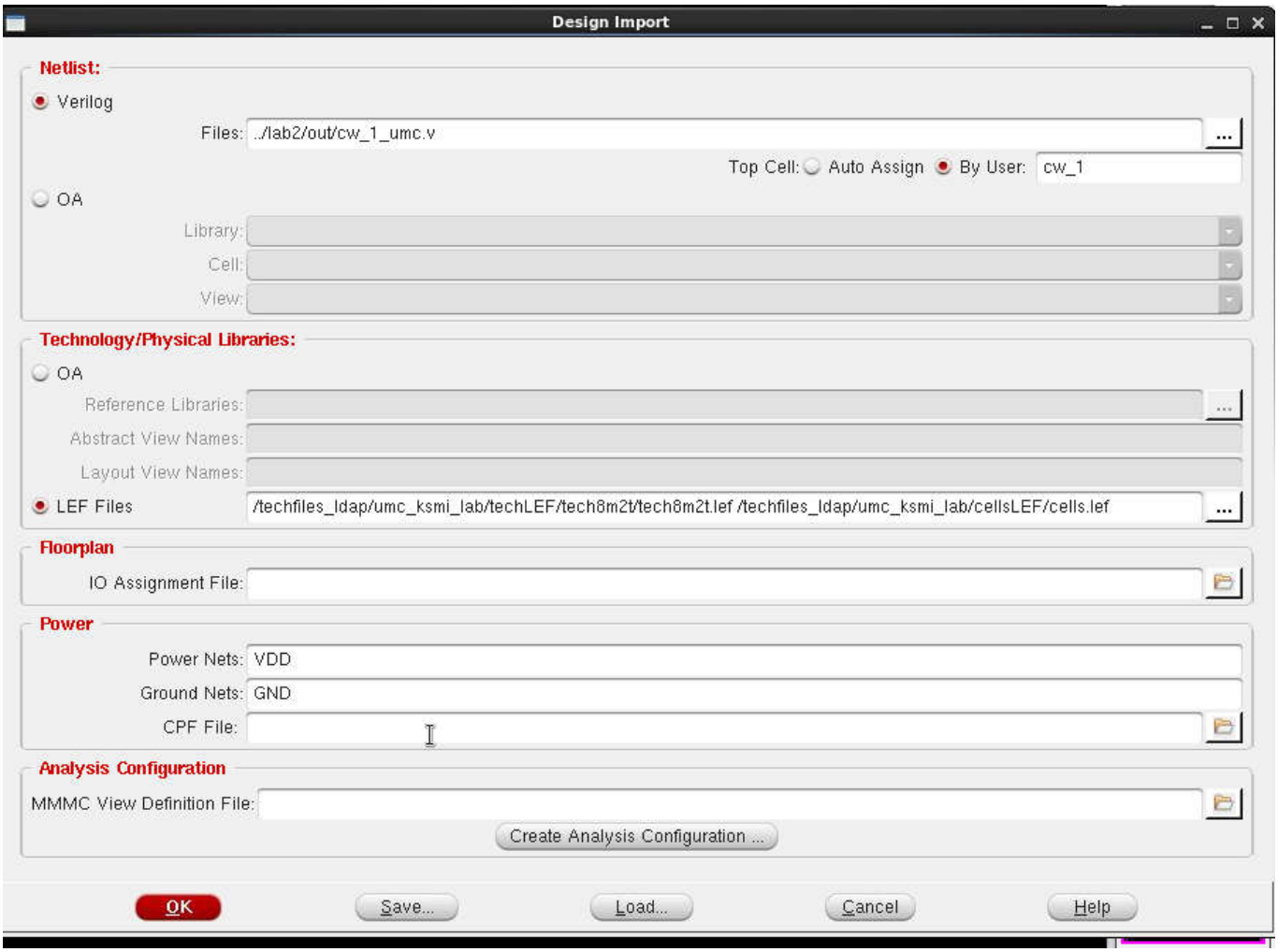
- klik i przeciągnięcie – zoom wyświetlanego obszaru,
- Shift + klik – przesuwanie wyświetlanego obszaru.

Polecenia można wydawać poprzez ich wybór z menu myszką, przez wydanie polecenia w oknie terminala oraz poprzez klawisze skrótów. Tabela zawierająca klawisze skrótów jest edytowalna a zmiany można wprowadzić poprzez menu ***View / Set Preference*** zakładka ***Design*** przycisk ***BindingKey***. Poniżej przedstawione są niektóre domyślne klawisze skrótów:

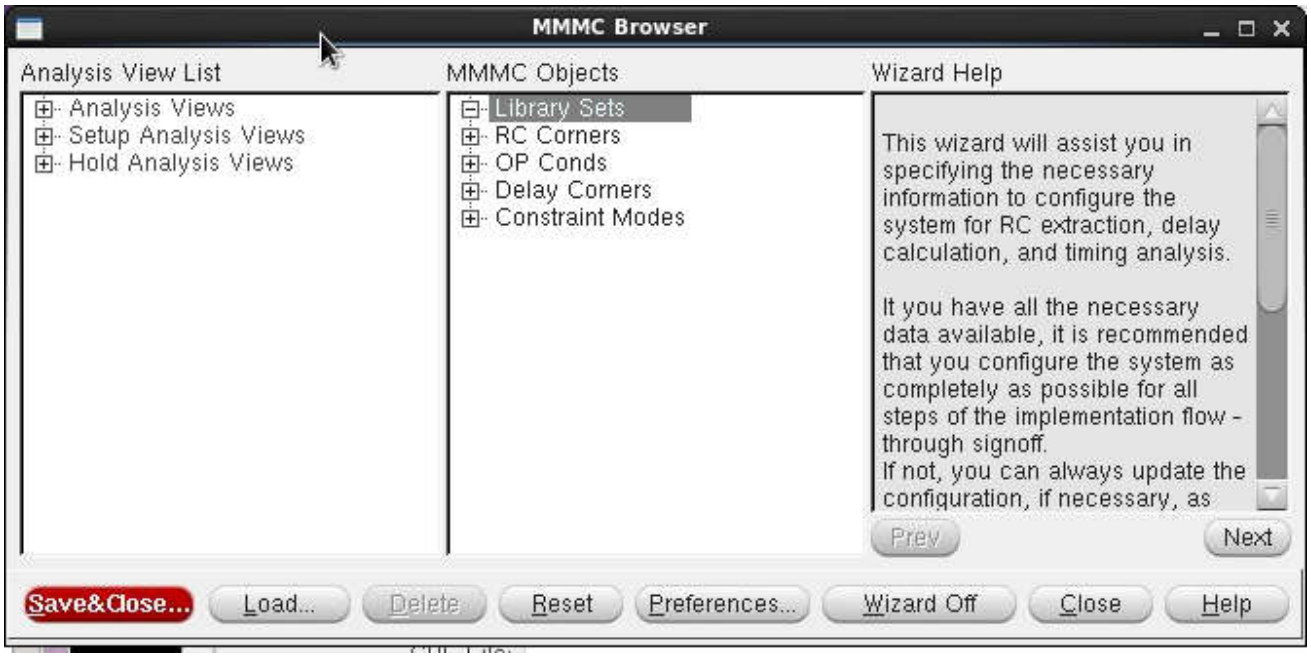
Klawisz	Opis działania
B	Otwiera formularz przypisywania klawiszy
F	Powiększa obraz do widoku całego rdzenia
G	Przesuwa hierarchię znaczonego obiektu do góry
Shift - G	Przesuwna hierarchię zaznaczonego obiektu w dół
K	Tworzy linijkę
Shift - K	Usuwa linijkę
Q	Otwiera edytor atrybutów zaznaczonego obiektu
V	Wyświetla atrybut zaznaczonego obiektu
U	Polecenie undo – cofa ostatnio wykonaną czynność
Shift - U	Cofa polecenie undo
Z	Powiększa wyświetlane obiekty 2 krotnie
Shift - 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 (z lewym klVDawiszem myszki)
Spacja	Zmiana wyboru elementów nałożonych na siebie

1) Import projektu

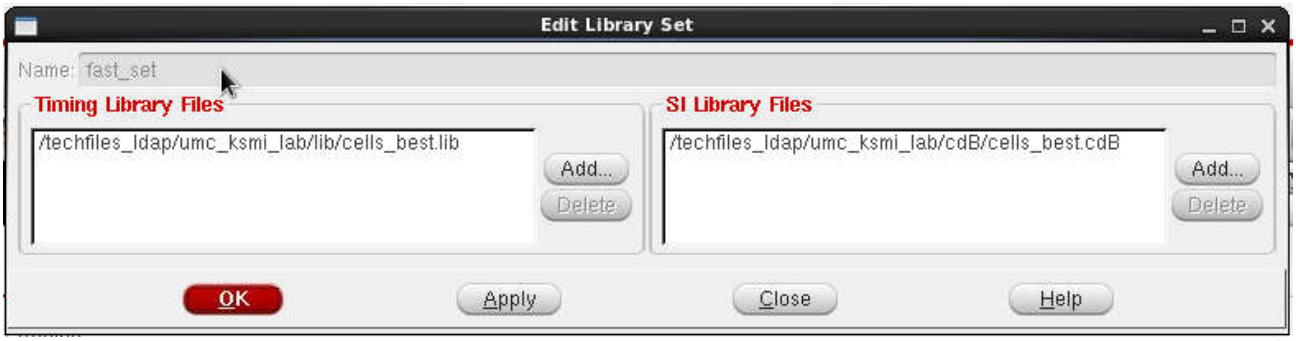
Pierwszą czynnością jest wczytanie wyników syntezy oraz bibliotek niezbędnych do przeprowadzenia implementacji. W tym celu należy wybrać menu ***File/Import Design***, otworzy się okno które należy wypełnić jak na rysunku poniżej.



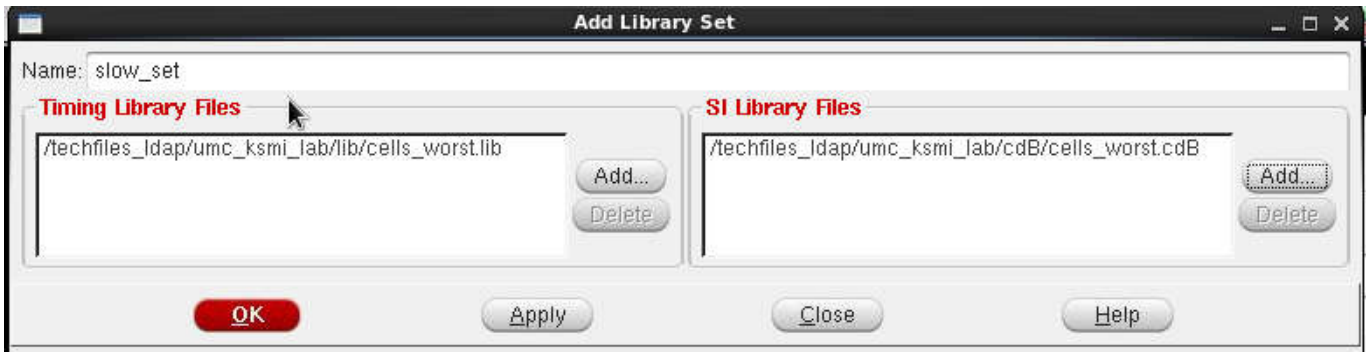
Następnie należy kliknąć na przycisk ***Create Analysis Configuration***. Otworzy się okno ***MMMC Browser***. Ten krok ma na celu skonfigurowanie wszystkich zestawów analiz czasowych projektowanego układu. Skrót MMMC oznacza Multi-Mode Multi-Condition. Widok okna przedstawiony jest poniżej.



Klikamy podwójnie na napis ***Library Sets*** i wprowadzamy informacje jak na poniższym rysunku a następnie potwierdzamy przyciskiem ***OK***.:

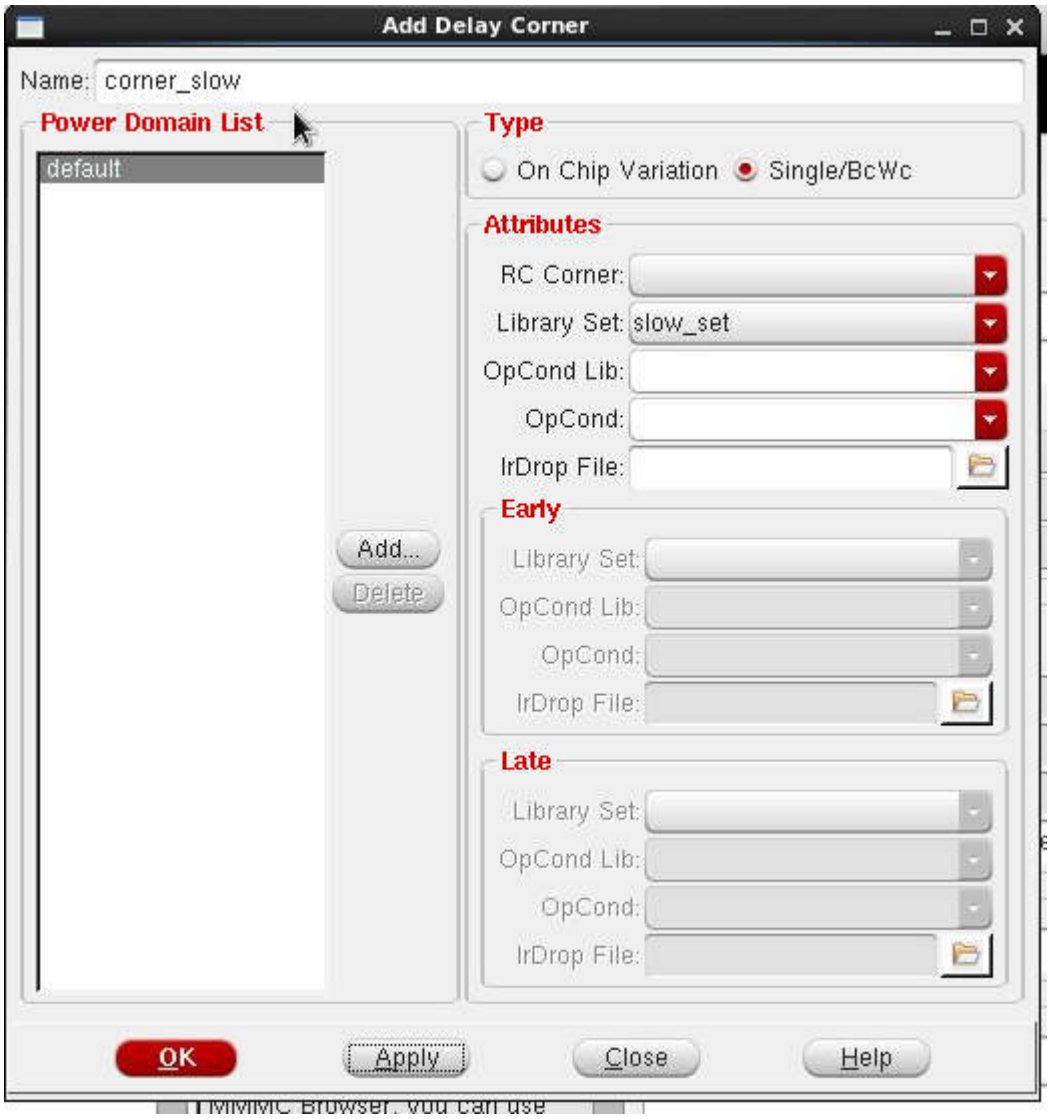


Ponownie klikamy na ***Library Sets*** i wprowadzamy informacje jak na poniższym rysunku:

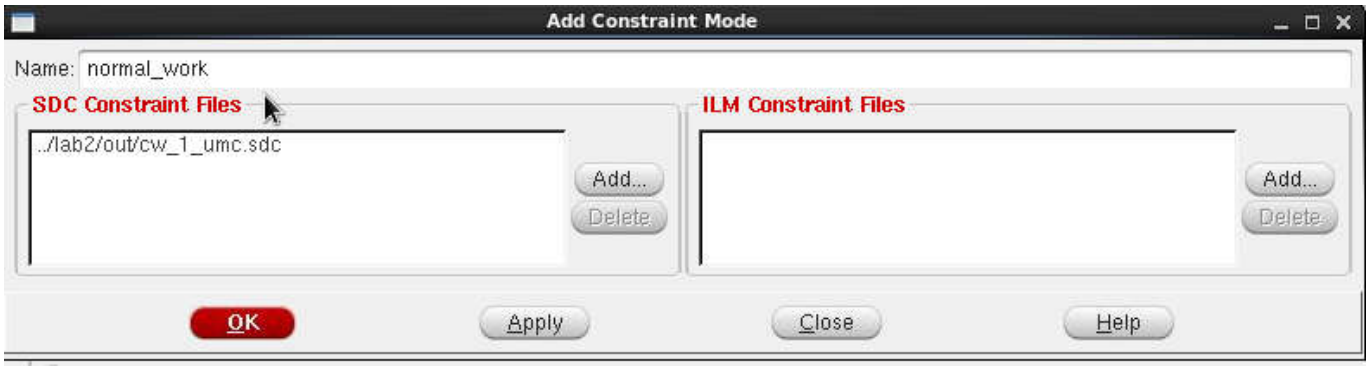


Klikamy podwójnie na napis ***Delay Corners*** i wprowadzamy informacje jak na poniższych dwóch rysunkach a następnie potwierdzamy przyciskiem ***OK***.:

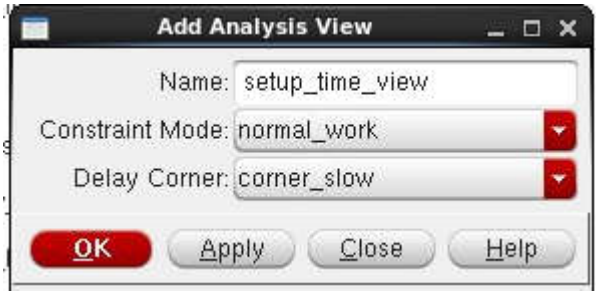

Add Delay Corner
Name: corner_fast
Power Domain List
default
Add...
Delete
Type
On Chip Variation
Single/BcWc
Attributes
RC Corner:
Library Set: fast_set
OpCond Lib:
OpCond:
IrDrop File:
Early
Library Set:
OpCond Lib:
OpCond:
IrDrop File:
Late
Library Set:
OpCond Lib:
OpCond:
IrDrop File:
OK
Apply
Close
Help

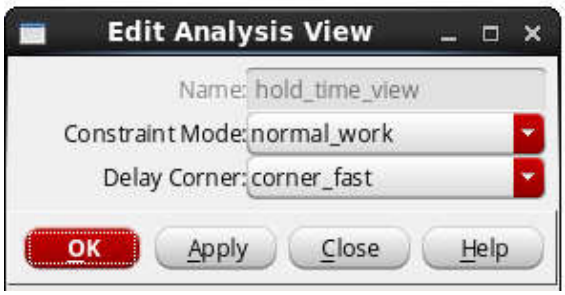


Klikamy podwójnie na napis ***Constraint Modes*** i wprowadzamy informacje jak na poniższym rysunku a następnie potwierdzamy przyciskiem ***OK***.:

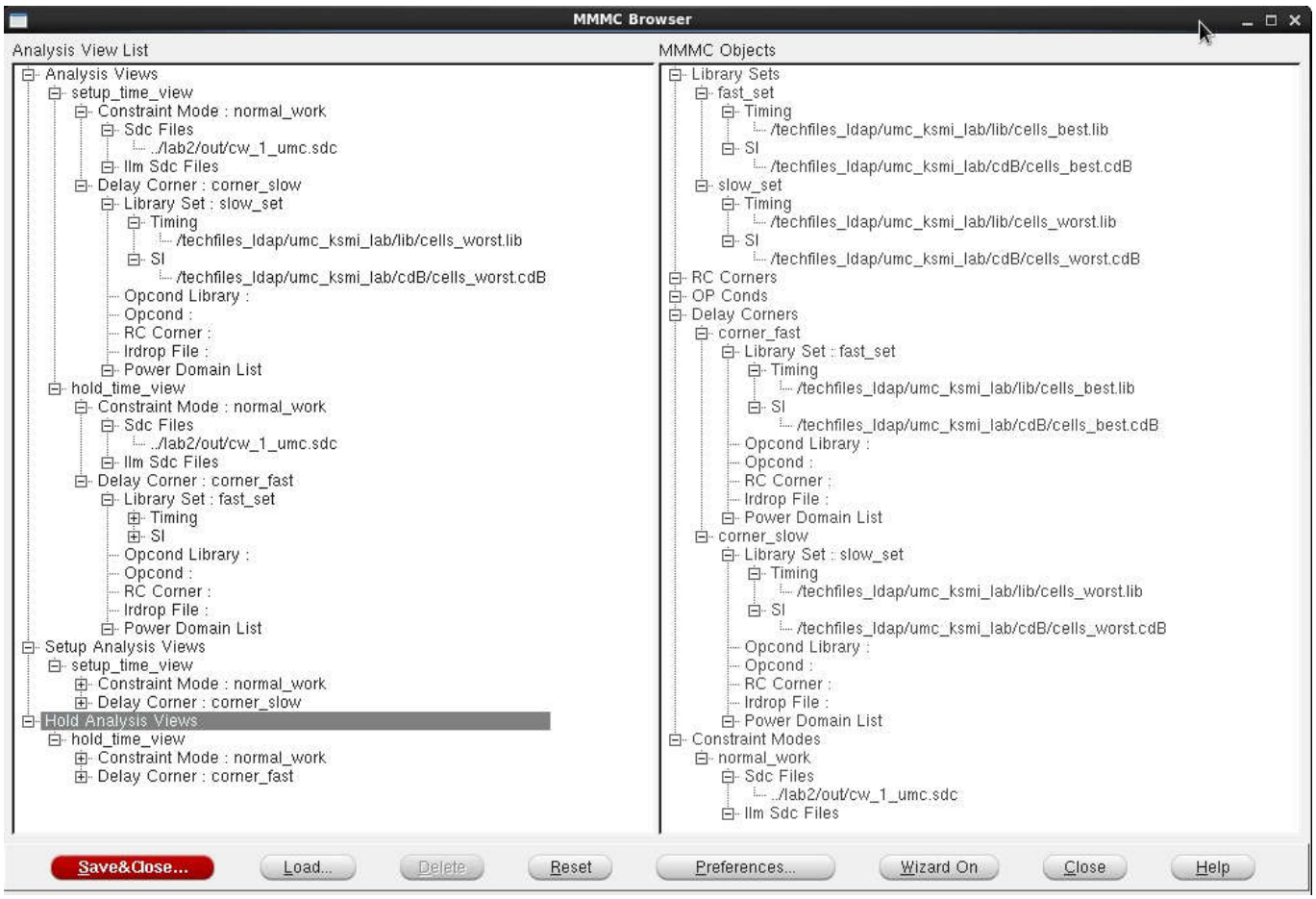


Klikamy podwójnie na napis ***Analysis View*** i wprowadzamy informacje 2 razy jak na poniższych rysunkach a następnie potwierdzamy przyciskiem ***OK***.:

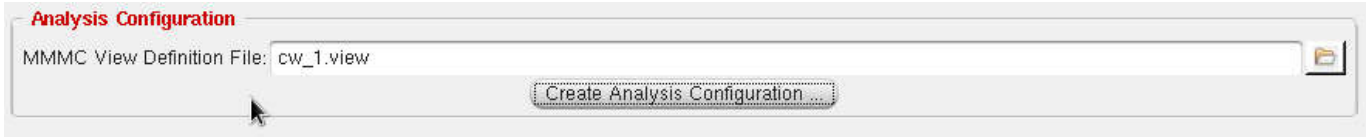




Ostatnim krokiem jest wskazanie które ***Analysis View*** ma być używane dla sprawdzania spełnienia czasu Hold a które dla sprawdzania spełnienia czasów Setup. W tym celu podwójnie klikamy na napis ***Setup Analysis View*** i wybieramy **setup_time_view** oraz klikamy na napis ***Hold Analysis View*** i wybieramy **hold_time_view**. Po powyższych operacja rozwinięte okno ***MMMC Browser*** powinno wyglądać jak na poniższym rysunku:

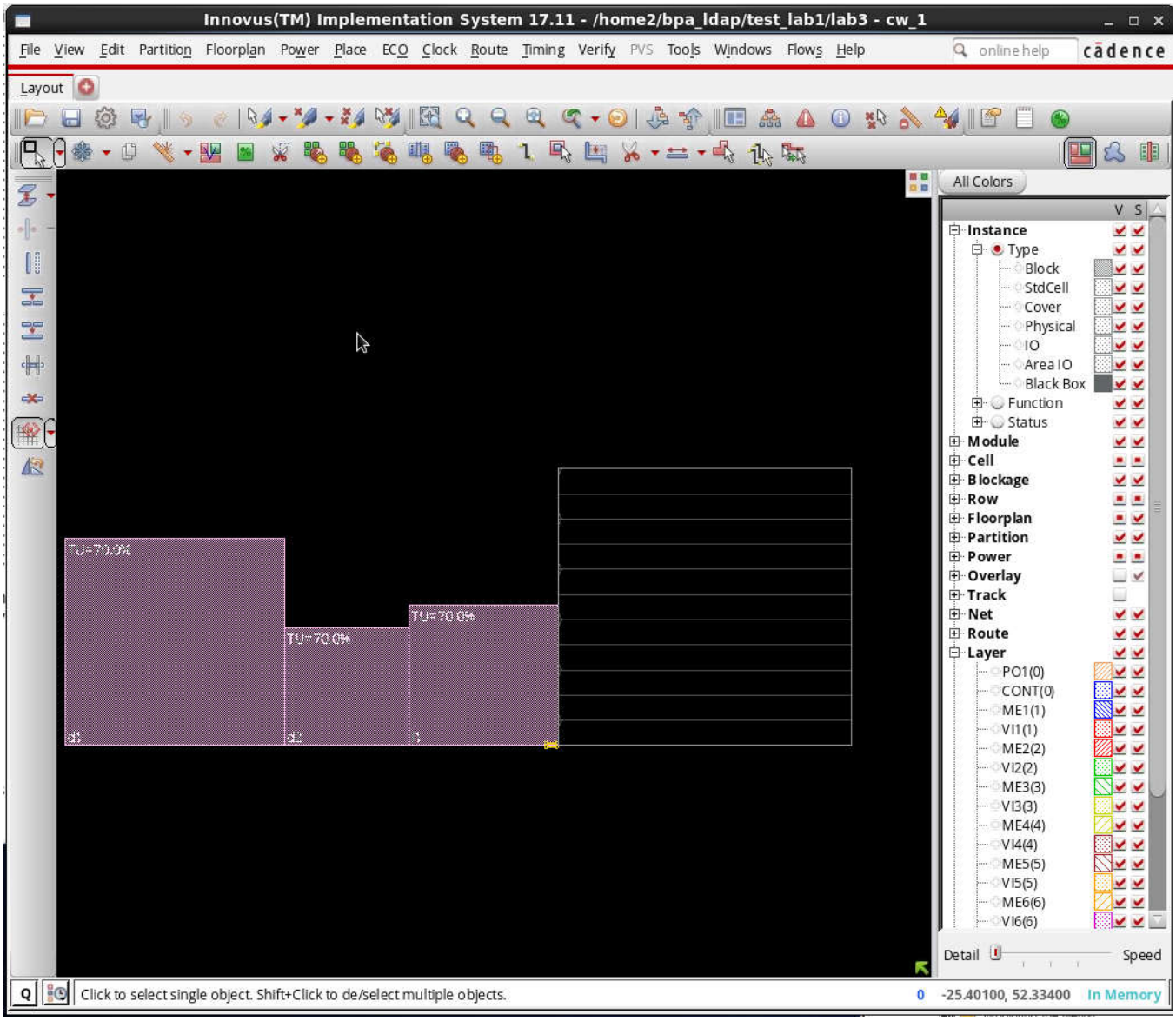


Po wypełnieniu wszystkich zakładek jak na powyższych rysunkach kilkamy ***Save&Close*** i nadajemy nazwę plikowi w którym zapisujemy konfigurację jako **cw_1.view**. W polu ***Analysis Configuration*** okna ***Design Import*** powinien pojawić się wpis nazwy pliku w którym zapisana jest konfiguracja analiza jak to przedstawiono poniżej:



Zanim klikniemy OK warto zapisać konfigurację importu, która może się w przyszłości okazać potrzebna. Klikamy więc ***Save...*** a następnie podajemy nazwę pliku **import_settings.globals**. Teraz możemy już przystąpić do wczytania projektu, klikamy ***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 ***View/Set Preference/Display*** i zmianę parametru ***Min. Floorplan Module Size:*** ze 100 na 1. Dodatkowo należy wybrać ***Florplan View*** przyciskiem:

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 (na rysunku powyżej są to: ***d1***, ***d2*** oraz ***l***) i pojedyncze kliknięcie 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 zmieniamy ***Core utilization*** na 0.5 (dla tego, małego przykładu mamy 50% wykorzystania powierzchni gdyż po dodaniu drzewa zegarowego może zabraknąć miejsca, dla dużych praktycznych projektów należy wpisać wyższą wartość ale zazwyczaj nie przekraczającą 80%), ***Ratio (H/W)*** na 0.6, klikamy na ***Core Margins by: Core to IO Boundary*** i wpisujemy wszystkie 4 marginesy na równe 10 a następnie klikamy na ***OK***. W wyniku otrzymujemy 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

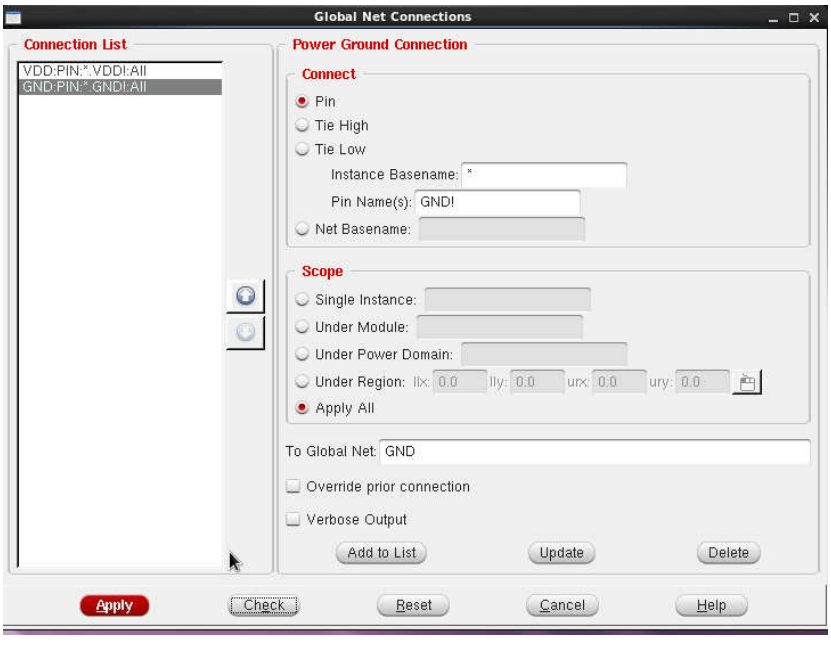
zabronione dla prowadzenia ścieżek połączeniowych. Można też samodzielnie ręcznie przesuwać elementy na obszar rdzenia w celu ich umieszczenia. Aby umieścić blokadę miejsca do umieszczania elementów należy kliknąć na  , (**`Create Placement Blockage`**) następnie LKM kliknąć w 2 przeciwległych rogach 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 **`File/Save Design...`** a następnie wybrać **`Data Type`** jako **`Innovus`** i podać nazwę pliku do zapisu. Odtworzenie wykonuje się poprzez menu **`File/Restore Design...`** 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 **`Power/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 VDD oraz wyprowadzenia GND! komórek standardowych do sieci globalnej GND. W tym celu musimy wpisać **`VDD!`** do pola **`Pin Name(s)`** oraz **`VDD`** do pola **`To Global Net`** przy zaznaczonym przycisku **`Apply All`** i kliknąć LKM na **`Add to List`**. Tą samą czynność należy powtórzyć dla wpisów **`GND!`** i **`GND`**, 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 wykonanie pierścieni zasilających wokół rdzenia układu scalonego. W tym celu wybieramy menu **`Power/Power Planning/Add Ring...`** a następnie w oknie **`Add Rings`** należy podać dodać nazwy sieci zasilania i masy. W tym celu klikamy na symbol  i dodajemy nazwy

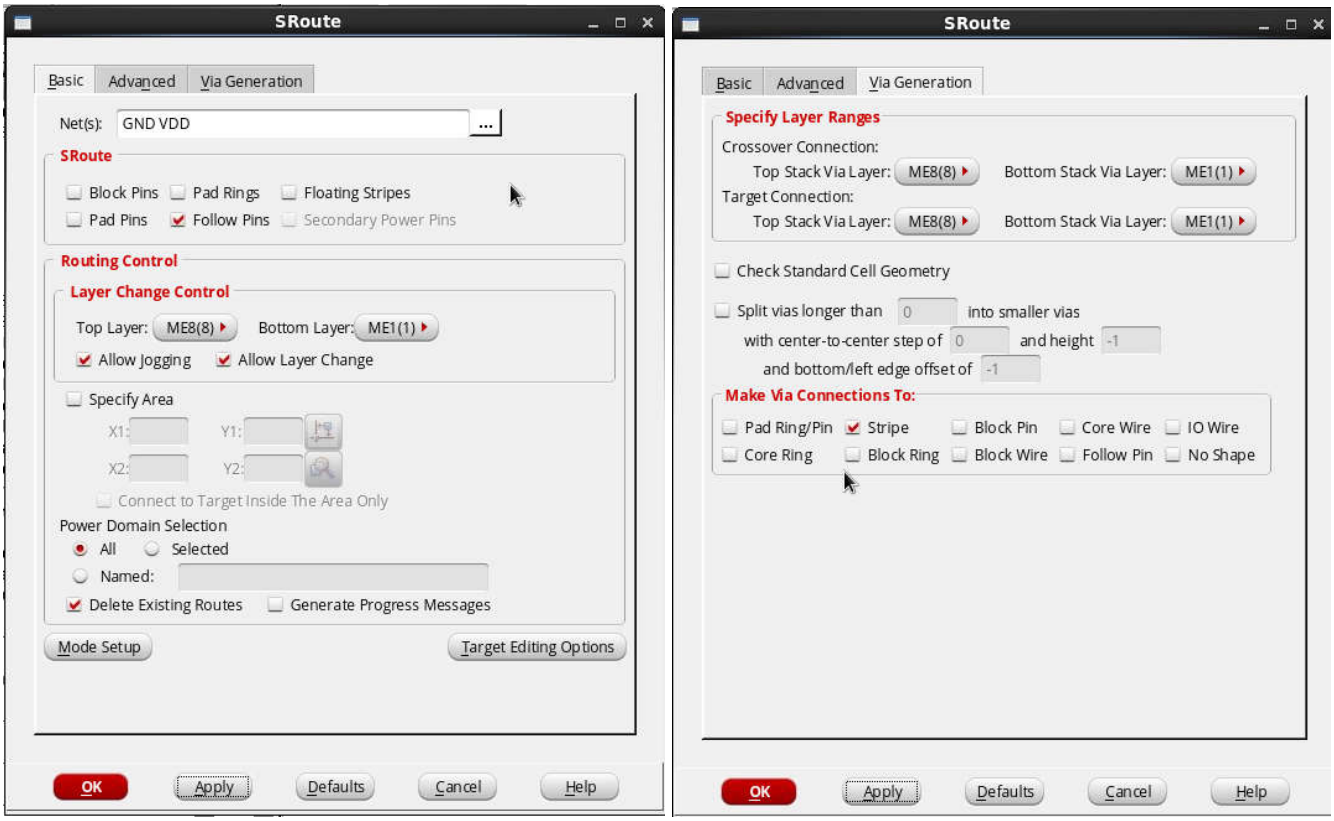
VDD i ***GND***. W polu ***Ring Type*** należy wybrać ***Around core boundary*** a w polu ***Ring Configuration*** należy wybrać metale górny i dolny jako ***ME7 H***, lewy i prawy jako ***ME8 V***, wszystkie o szerokości 4 (width), odstępie 0.5 (spacing) i przesunięciu 0.4 (offset). Prawidłowe ustawienia przedstawione są na rysunku poniżej.

Po kliknięciu ***Apply*** naokoło rdzenia układu powinien pojawić się podwójny pierścień zasilający.

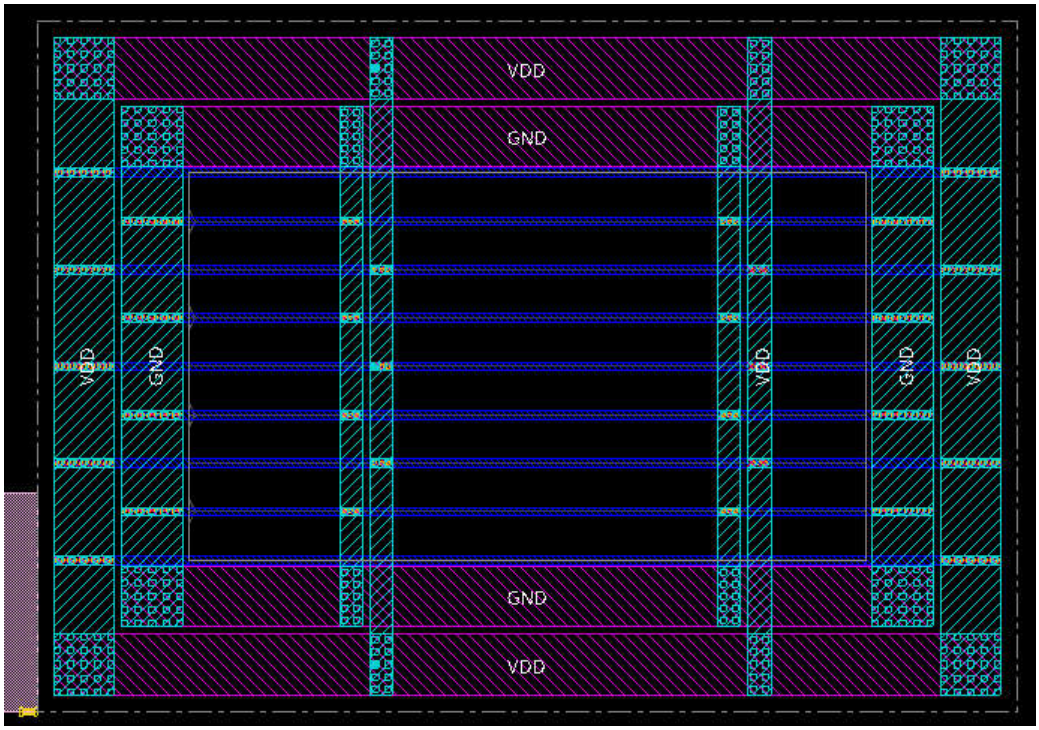
Kolejnym krokiem jest dodanie pionowych linii zasilających, które zwiększają wydajność prądową obwodu zasilania. Takie linie typowo umieszcza się co około 50µm - 150µm, w naszym układzie, który ma tylko około 60µm szerokości nie są one konieczne ale zostaną umieszczone w celach edukacyjnych. Wybieramy menu ***Power/Power Planning/Add Stripe...*** a następnie w oknie zatytułowanym ***Add Stripes/Set Configuration*** należy podać dodać nazwy sieci zasilania i masy. W tym celu klikamy na symbol  i dodajemy nazwy ***VDD*** i ***GND*** następnie jako metal połączeniowy wybieramy ***ME8*** a szerokość pasków zmieniamy na 1.5, spacing na 0.5. W polu ***Set Pattern*** należy wstawić wartość ***Set-to-set distance*** równą 25, w polu ***First/Last Stripe*** należy zaznaczyć pole ***Left*** oraz ustawić wartość ***Relative from core of selected area Start:*** na 15. Prawidłowo wypełniony formularz przedstawiony jest na rysunku poniżej.

Po kliknięciu ***Apply*** powinny pojawić się dodatkowe linie łączące górny i dolny pasek zasilania.

Ostatnim krokiem tworzenia obwodów zasilania jest położenie poziomych linii metalowych na poziomie ***ME1***, które służą do bezpośredniego podłączenia zasilania komórek standardowych. W tym celu wybieramy menu ***Route/Special Route...*** a następnie w oknie ***SRoute*** należy podać nazwy sieci zasilania. W tym celu klikamy na symbol  i dodajemy nazwy ***VDD*** i ***GND***. W polu ***SRoute*** pozostawiamy zaznaczone wyłącznie ***Follow Pins*** a resztę zostawiamy ustawione domyślnie. W zakładce ***Via Generation*** należy zaznaczyć opcję ***Make Via Conection To: Stripe***. Prawidłowo wypełniony formularz przedstawiony jest na rysunku poniżej.



Kliknięcie przycisku ***Apply*** spowoduje wygenerowanie linii zasilających w rdzeniu. Projekt na obecnym etapie powinien wyglądać jak na rysunku poniżej.

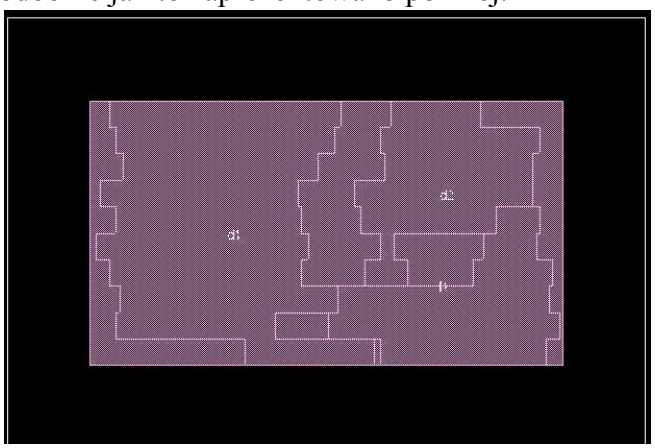


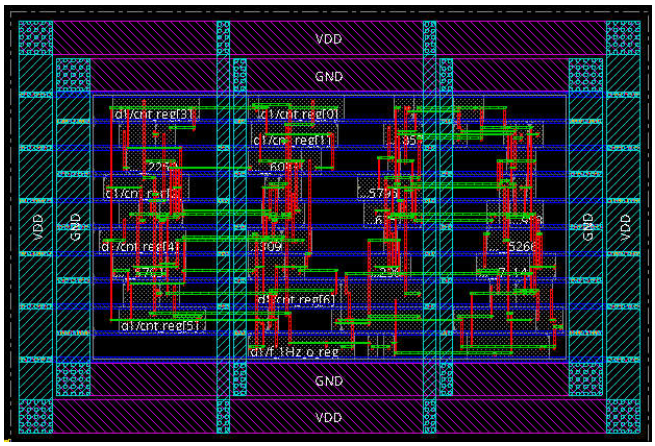
W tym miejscu można wykonać pierwsze weryfikacje poprawności dotychczasowego projektu. Testowanie poprawności połączeń zasilania (specjalnych) wykonuje się poprzez menu ***Verify/Verify Connectivity*** a następnie zaznaczenie ***Special Only***. Klikamy ***OK*** i w oknie

konsoli powinniśmy otrzymać raport. Podobnie można sprawdzić geometrię, menu ***Verify/Verify Geometry*** i ***OK***. Sprawdzenie

4) Rozmieszczanie komórek standardowych (ang. placement) i wstępne zgrubne trasowanie połączeń

W naszym projekcie jest już wstępnie wykonany projekt kształtu rdzenia i zaplanowana jego utylizacja. W celu rozmieszenia komórek standardowych najpierw wybieramy reguły rozmieszczania. 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/Place Standard Cell...*** 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.





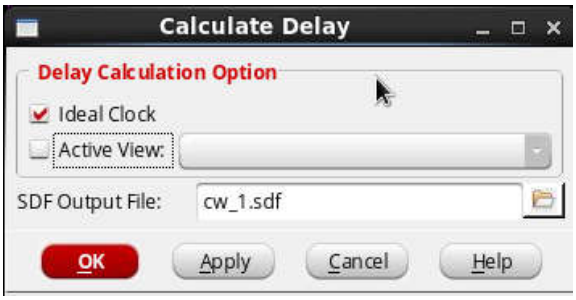
Podobnie jak poprzednio warto sprawdzić poprawność geometrii i połączeń. Wstępne trasowanie niestety generuje dużo błędów geometrii, które można poprawić po wykonaniu kolejnego trasowania.

5) Trasowanie połączeń (ang. routing)

Do trasowania połączeń zasilających używany jest program ***Sroute***. Połączenia standardowe można wykonać używając ***Early Global 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 ***Nano Route*** wybierając z menu ***Route/Nano Route*** a następnie klikamy ***OK*** bez zmiany ustawień domyślnych. To powinno skutkować ponownym wykonaniem połączeń pomiędzy komórkami standardowymi już bez błędów geometrii projektu.

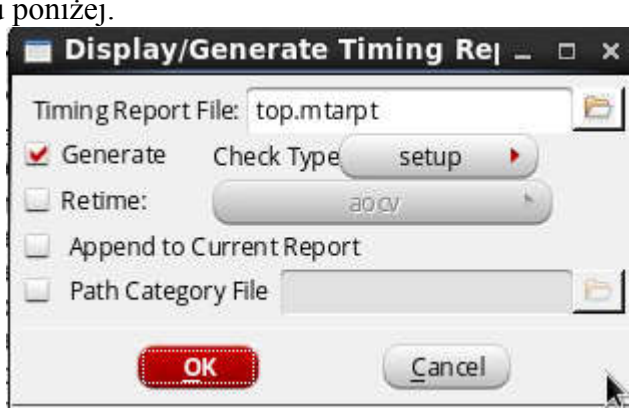
6) Ekstrakcja pojemności i rezystancji ścieżek oraz obliczanie opóźnień

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 zapis SDF poprzez menu ***Timing/Write SDF...*** pozostawiamy opcje domyślne (jak na rysunku poniżej) i klikamy ***OK***.

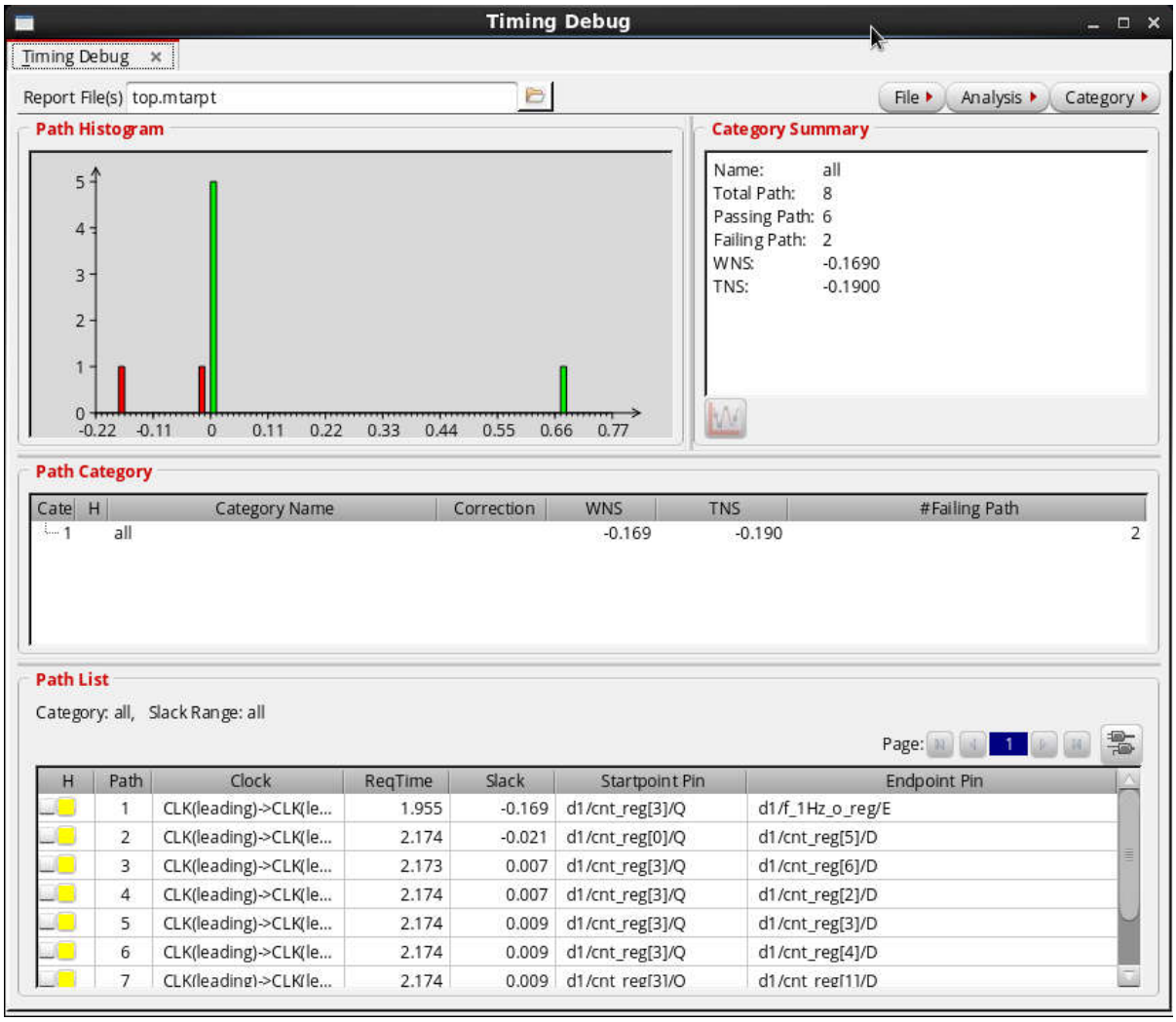


7) Analiza czasów ustalania (ang. setup)

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 w procesie importu projektu na starcie procesu implementacji. Obecnie możemy sprawdzić warunki jakie były wcześniej ustalone poprzez menu ***Timing/MMMC Browser***. Przechodzimy do analizy czasowej poprzez menu ***Timing/Report 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 ***Innovus*** ale lepiej jest użyć menu ***Timing/Debug Timing***. Otwiera się okno jak na rysunku poniżej.



Klikamy ***OK*** a następnie pojawi się okno jak poniżej.



Czerwone słupki, jeśli się pojawią na wykresie, reprezentują ścieżki sygnałowe które nie spełniają wymagań czasowych. 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 a na pionowej liczba ścieżek, które ich nie spełnia. 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. Pojawi się dodatkowe okno zatytułowane ***Timing Path Analyzer***. Wybrana wcześniej ścieżka analizowana jest szczegółowo, kliknięcie w dowolny jej fragment podświetla tą część w oknie głównym programu ***Innovus***. 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 ***ECO/Optimize Design***, 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 polecenia ***Timing/Extract RC, Timing/Write Sdf, Timing/Report Timing*** oraz ***Timing/Debug Timing...*** Pojawi się okno ***Timing Debug*** z wynikami z poprzedniej analizy czasowej. Aby ją zaktualizować należy kliknąć na  a następnie ponownie wygenerować raport.. Optymalizacja nie koniecznie może dać pozytywne rezultaty - jeśli były założone zbyt wysokie wymagania czasowe w czasie syntezy po implementacji może okazać się niemożliwe to do spełnienia. Optymalizacja czasowa niestety powoduje powstanie błędów geometrycznych. Naprawienie ich można wykonać poprzez ponowny routing wykonany jak w pkt. 5.

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 (ang. setup) natomiast poprawę czasów podtrzymania (ang. hold). W celu utworzenia drzewa zegarowego niezbędne jest wydanie poleceń z oknie terminala:

1) Podanie nazw buforów użytych do budowy drzewa zegara:

set_ccopt_property buffer_cells {BUFCKSP16V1_0 BUFCKSP1V1_0 BUFCKSP2V1_0 BUFCKSP32V1_0 BUFCKSP3V1_0 BUFCKSP4V1_0 BUFCKSP8V1_0}

b) podanie nazw inwerterów użytych do budowy drzewa zegara:

set_ccopt_property inverter_cells {INVCKSP16V1_0 INVCKSP1V1_0 INVCKSP2V1_0 INVCKSP32V1_0 INVCKSP4V1_0 INVCKSP8V1_0}

Powyżej użyte bufory i inwertery są specjalnie zaprojektowanymi komórkami o wyrównanych czasach narastania i opadania sygnału na ich wyjściu i stąd są szczególnie przydatne dla prowadzenia sygnału zegara.

2) Umożliwienie wstawiania inwerterów:

set_ccopt_property use_inverters true

3) Podanie maksymalnej wartości opóźnienia zegara:

set_ccopt_property target_max_trans 250ps

4) Podanie maksymalnej wartości skew zegara:

set_ccopt_property target_skew 50ps

5) Zbudowanie drzewa zegara i optymalizacja czasu setup:

ccopt_design

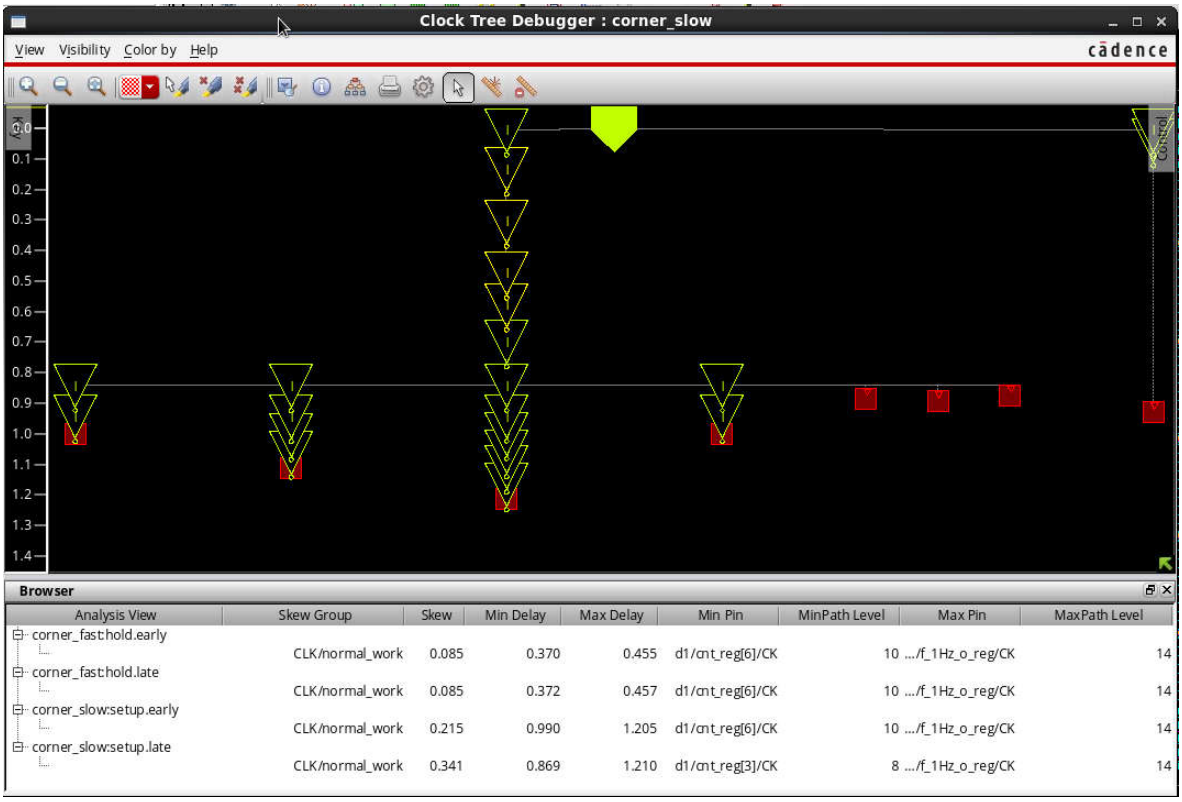
Po wykonaniu powyższego polecenia do projektu powinny zostać dodane elementy drzewa zegarowego i automatycznie powinna być wykonana optymalizacja czasów setup. Aby uzyskać tekstowy raport dotyczący drzewa zegarowego należy wykonać polecenie:

report_ccopt_clock_trees -file clk.rpt

Uwaga: w przypadku bardzo prostych projektów z niewymagającymi wymaganiami czasowymi wygenerowana ścieżka zegara może nie zawierać żadnych elementów aktywnych wówczas pojawi się w konsoli błąd jak niżej, który należy zignorować:

```
ERROR     IMPCCOPT-5054        1  Net %s is not completely connected after...
WARNING   IMPCCOPT-1261        2  The skew target of %s for %s is too smal...
WARNING   IMPTCM-77            9  Option "%s" for command %s is obsolete a...
*** Message Summary: 50 warning(s), 1 error(s)
```

W pliku o nazwie ***clk.rpt*** umieszczone zostanie podsumowanie operacji generacji drzewa zegarowego. Graficzne informacje o drzewie zegarowym można uzyskać poprzez wybranie menu ***Clock/CCOpt Clock Tree Debugger...*** a następnie w kolejnych 2 oknach należy potwierdzić domyślne ustawienia i wygenerowane zostanie okno jak poniżej (dla prostych projektów bez żółtych inwerterów/buforów) :



Analiza czasowa po generacji drzewa zegarowego zmienia się na korzyść gdyż ścieżka zegarowa zwiększa swoje opóźnienie co poprawia wartość czasu setup. Aby to sprawdzić należy wybrać kolejno menu ***Timing/Report Timing***, wybieramy opcję ***Post-CTS*** oraz ***Setup*** i klikamy ***OK***. Podsumowanie analizy mamy na konsoli a szczegóły można przebadać poleceniem ***Timing/Debug Timing***. Aby wczytać bieżące wartości należy jednak w oknie ***Timing Debug*** kliknąć na a następnie ponownie wygenerować raport.

10) Analiza czasów podtrzymania (ang. hold)

Tą analizę wykonujemy podobnie do poprzedniej, zaznaczamy tylko opcję czasów hold zamiast setup. Wybieramy menu ***Timing/Report 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 wygenerować i załadować plik analizy poprzez kliknięcie na wybranie czas ***hold***) .

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 ***ECO/Optimize Design***, 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/Report Timing*** oraz ***Debug Timing***.

UWAGA: Jeśli w trakcie planowania powierzchni (ang. floorplaning) została wprowadzona zbyt duża utylizacja wówczas może zabraknąć miejsca na dodatkowe komórki buforów sygnałowych/zegarowych i optymalizacja się nie powiedzie. Podobnie, nałożenie zbyt wysokich wymagań czasowych również może uniemożliwić prawidłowe wyniki z analiz czasowych jak również błędy geometrii i połączeń. W takich przypadkach projekt należy powtórzyć, oczywiście po wcześniejszej korekcie ww. wymagań.

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

Każdy projekt układu cyfrowego niskoskalowanego musi być przeanalizowany pod kątem 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***.

13) Analiza poboru mocy

Moc pobieraną przez układ można oszacować 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/Setup...*** Pozostawiamy opcje domyślne i klikamy ***OK***. Następnie wybieramy menu ***Power/Power Analysis/Run...*** 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.

W programie ***Innovus*** możliwe jest uruchamianie poleceń w postaci skryptów. W tym celu należy użyć polecenia ***source nazwa_skryptu***. Program w czasie pracy rejestruje wszystkie wydane polecenia do domyślnego pliku ***innovus.cmd***, rejestruje również wszystkie komunikaty konsoli do domyślnego pliku ***innovus.log***.

Załącznik: Wzór protokołu:

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

Ćwiczenie opracował Bogdan Pankiewicz, Gdańsk, wrzesień 2010
aktualizacje: październik 2014, lipiec 2018, październik 2018