UNIVERZITET U NIŠU, ELEKTRONSKI FAKULTET

UPUTSTVO ZA

Projektovanje Digitalnih ASIC Kola metodom standardnih ćelija u Mentor Graphics Pyxis Okruženju

Dejan Mirković, asistent

2017.

Sadržaj

Uvod

Ovo je kratko uputstvo za tok projektovanja digitalnih ASIC kola u Mentor Graphics (MGC) Pyxis okruženju korišćenjem standardnih ćelija iz studenske verzijie PDK-a, Generic Desing Kit (GDK) v10.2_12.1_r02.

Potrebni MGC alati za realizaciju ovog toka projektovanja su:

- ModelSim (HDL simulator),

- LeonardoSpectrum (RTL kompajler za sintezu do nivoa gejtova),

- Pyxis Project Manager (GUI za prikaz i menadžment projektnih fajlova),

- Pyxis Design Architect (Šematik editor),

- Pyxis IC Station (Layout Editor) i

- Pyxis Assembly Place & Route (set alata za automatizaciju projektovanja).

Pošto CAD/EDA alati često generišu veliki broj sporednih fajlova dobra je praksa pre početka rada kreirati odgovarajuću direktorijumsku strukturu. U ovom uputstvu biće korišćena sledeća direktorijumska struktura:

- `$HOME/hdl/src/counter` (direktorijum za smeštanje izvornih HDL fajlova)

- `$HOME/hdl/sim` (radni direktorijum ModelSim simulatora)

- `$HOME/hdl/sim/gdk` (ModelSim biblioteka)

- `$HOME/hdl/syn` (radni direktorijum LenardoSpectrum kompajlera)

- `$HOME/hdl/syn/net` (direktorijum za smeštanje netlisti na nivou gejtova (nakon sinteze))

- `$HOME/hdl/syn/rpt` (direktorijum za smeštanje izveštaja nakon sinteze)

- `$HOME/mentor-graphics/work` (radni direktorijum Pyxis alata).

Tok projektovanja će biti demonstriran kroz jednostavan primer projekovanja četvorobitnog brojača.

Napomena: Prethodno navedena direktroijumska struktura je već kreirana u `$HOME` direktorijumu svakog korisnika. Direktorjimu `$HOME/mentor-graphics/work` je već kreiran u prvom delu kursa. Do direktorijuma se može doći i simboličkim linkom `pyxis` koji se nalazi u `$HOME` direktorijumu koristnika.

Pre-Synthesis Simulacija (ModelSim)

ModelSim je multi-language simulator. Demonstracije radi RTL model brojača je opisan u VHDL, a testbenč u Verilog HDL jeziku. Naravno, sve može biti opisano isim jezikom po izboru korisnika.

VHDL kôd brojača:

```
library IEEE;
use IEEE.std_logic_1164.all;
use IEEE.numeric_std.all;

entity COUNTER is
    generic ( N : natural:= 4 );
    port (
        clk : in std_logic;
        rst : in std_logic;
        en  : in std_logic;
        cnt : out std_logic_vector(N-1 downto 0)
    );
end entity COUNTER;

architecture COUNTER_RTL of COUNTER is
    signal cnt_buff : std_logic_vector(N-1 downto 0);
begin

    CNT_P : process(clk, rst, en) is
    begin
        if (rst = '0') then
            cnt_buff <= (others => '0');
        elsif (rising_edge(clk) ) then
            if (en = '1') then
                cnt_buff <= std_logic_vector(unsigned(cnt_buff) + 1);
            end if;
        end if;
    end process CNT_P;

    cnt <= cnt_buff;

end architecture COUNTER_RTL;
```

Verilog kôd testbenča:

```
`timescale 1ns/1ps

module COUNTER_TB;

    parameter N=4;
    integer ENDSIM;

    wire [N-1:0] cnt;
    reg clk, rst, en;

    `ifdef SYN
        COUNTER
           DUT(.clk(clk), .rst(rst), .en(en), .cnt(cnt));
    `else
```

```
        COUNTER #(.N(N))
           DUT(.clk(clk), .rst(rst), .en(en), .cnt(cnt));
    `endif

    initial begin

        $display("\tTime\trst\ten\tcnt");
        rst = 0;
        en = 0;
        clk = 0;

        #25;
        rst = 1;
        #50;
        en = 1;
        #210;
        en = 0;
        #130;
        en = 1;
        #53;
        rst = 0;
        #10;
        $stop(2);
    end

    always begin
        #10 clk = ~clk;
    end

    always @(*) begin
        $display("\t%0d\t%0d\t%0d\t%0d", $time, rst, en, cnt);
    end

endmodule
```

Treba primetiti da se izbor DUT instance bira pomoću kompajlerske direktive `ifdef (promenljiva SYN). Ukoliko se prilikom kompajliranja testbenč fajla definše promenljiva SYN biće instacirano sitetizovano kolo u suprotnom instacira se RTL model. HDL fajovi su smešteni u $HOME/hdl/src.

Napomena: U $HOME/hdl/src korisnik treba da kreira direktorijum za smeštanje HDL fajlova za svoj projekat

```
cd $HOME/hdl/src
mkdir project_name
```

U project_name prekopirati sve HDL fajlove (uključujući i testbenč fajl).

Pozicionirati se u $HOME/hdl/sim direktorijum i otvoriti counter.do fajl za ModelSim simulator.

Napomena: Prilikom rada na sopstvenm projektu prekopirati counter.do fajl i izmeniti deo označen kurzivom bold.

DO fajl (counter.do):

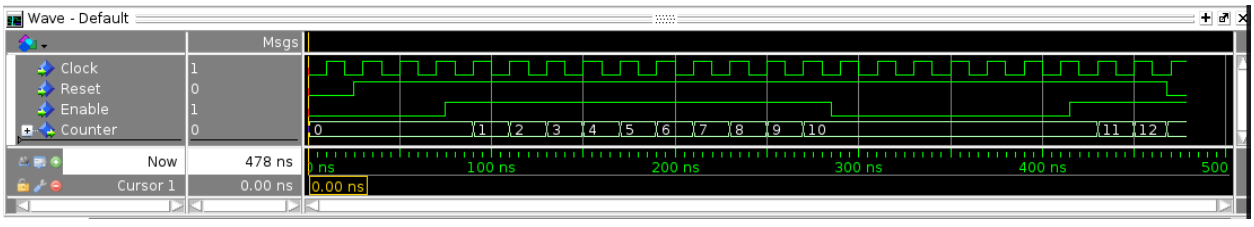
```
# User defined variables
```

```
set DESIGN counter
set TB COUNTER_TB
set VHDL_FILES [ glob ../src/$DESIGN/*.vhd ]
set VERILOG_FILES [ glob ../src/$DESIGN/*.v ]

# Define & Map work library
vlib rtl
vmap work rtl

# Compile VHDL & Verilog files
if { [ lsearch [info vars] VHDL_FILES ] != -1 }\
{ vcom $VHDL_FILES }
if { [ lsearch [info vars] VERILOG_FILES ] != -1 }\
{ vlog $VERILOG_FILES }

# Set the Top-Level module/entity for the simulation
vsim work.$TB

# Define which ModelSim windows to view
view structure
view signals
view wave


# Add signals to the waveform
add wave -label Clock DUT:clk
add wave -label Reset DUT:rst
add wave -label Enable DUT:en
add wave -label Counter -radix unsigned DUT:cnt

# Run simulation
run -all

# Zoom to fit waves to the window
wave zoom full
```

DO fajl za automatizaciju postupka simulacije i prezenatacije rezultata se pokreće sledećom komandom:

```
vsim -do counter.do &
```

Po završetku simulacije dobiće se rezultati kao na slici Sl. 1.



Sl. 1 Rezultati simulacije RTL modela četvorobitnog brojača

Naravno, nije uvek potrebno pokretati ModelSim grafičko okruženje tj. simulator se može pokrenuti i u konzolnom/komandnom načinu rada. Za pokretanje simulatora u konzolnom načinu rada treba koristiti sledeću komandu:

```
vsim -c -do counter.do
```

Sinteza (LeonardoSpectrum)

Sledeći korak u toku projektovanja je sinteza RTL model projektovanog digitalnog kola do nivoa gejtova u ciljnom tehnološkom procesu. U ovom slučaju radi se o CMOS 130nm tehnološkom procesu.

Napomena: Podrazumeva se da je HDL (Verilog ili VHDL) kôd RTL modela projektovanog kola uspešno verifikovan (bez sintaksnih grešaka, a funkcionalnost kola potvrđena logičkom simulacijom – prethodno poglavlje).

Sinteza podrazumeva niz optimizacionh rutina kojim se od RTL modela dolazi do strukturne predstave kola na nivou gejtova (standarndih ćelija). Postupak sinteze je automatizovan odgovarajućim Tcl skriptom. Primer skripta za automatizaciju postupka sinteze, `leospec.tcl`, se nalzi pod `$HOME/hdl/syn`. Detaljna dokumentacija vezana za LeonardoSpectrum komande se može pogledati u `/home/docs/leospec_ref.pdf`.

Napomena: Prilikom rada na sopstvenom projektu prekopirati skript `leospec.tcl` i izmeniti deo koji je označen podebljanim slovima.

TCL skript za sintezu (`leospec.tcl`):

```
clean_all

# User defined variables
set USER dmirkovic
set DESIGN counter
set TOP COUNTER
set TOP_ARCH COUNTER_RTL
set TB_FILE counter_tb.v
set FCLK 50.0e6

# Internal veriables
if { $TOP_ARCH != [] } { set VIEW .work.${TOP}.${TOP_ARCH} } \
else { set VIEW .work.${TOP} }
set HDL_FILES [ glob ../src/$DESIGN/* ]
set ID [ lsearch $HDL_FILES ../src/$DESIGN/$TB_FILE ]
if { $ID >= 0 } \
{ set HDL_FILES [ lreplace $HDL_FILES $ID $ID ] }
set TCLK [ expr ceil(1e9/$FCLK) ]
set TCLK_PROC [ expr ceil(0.6*$TCLK) ]

# Tool specific variables
set verilog_write_skip_open TRUE
set verilog_dummy_net_name dummy
set edifout_power_ground_style_is_net TRUE
set force_user_load_values
set max_fanout_load 14

# Set working directory
set_working_dir /home/$USER/hdl/syn
```

```
# Load tech. library
load_library
"/home/cad/pdks/mgc/STDC/GDKgates/GDKgates_utilities/hdl_libs/gdk.syn"


# Read HDL files
read $HDL_FILES -technology "gdk"

# Timing Contraints
set_attribute -port clk -name clock_cycle -value $TCLK
set register2register $TCLK
set input2register  $TCLK_PROC
set register2output $TCLK_PROC
set input2output    $TCLK_PROC

# Synthetize to gates level
optimize $VIEW -target gdk -effort quick -macro -hierarchy flatten

# Re-synthetize to gates level with Timing Constraints
optimize_timing $VIEW -through [ all_outputs ]

# Report area and timing
report_area rpt/${DESIGN}.area.rpt -cell_usage -all_leafs
report_delay rpt/${DESIGN}.delay.rpt -critical_path -clock_frequency
#report_delay rpt/semafor.delay.rpt -longest_path -critical_path

# Write gate level netists
apply_rename_rules -ruleset VERILOG
auto_write net/${DESIGN}.gate.sdf
auto_write net/${DESIGN}.gate.v
```

Bitne promenljive u skriptu za sintezu su:

`USER`	Ime korisnika Linux sistema (ono što vrati Linux komanda `echo $USER`).
`DESIGN`	Naziv projekta.
`TOP`	Naziv modula/entiteta najvišeg u hijerarhiji (top-level).
`TOP_ARCH`	Naziv arhitekture top-level entiteta (samo za VHDL model, u slučaju Verilog modula staviti prazna string `""`),
`TB_FILE`	Naziv testbenč fajla.
`FCLK`	Frekvencija signala takta (bitno za optimizaciju kašnjenja).

TCL fajl za automatizaciju postupka sinteze se pokreće sledećom komandom:

```
spectrum -file leospec.tcl
```

Postupk sinteze se može pratiti putem poruka (`Info`, `Warning`, `Error`) koje se ispisuju u standardni izlaz (terminal). Kompletan sadržaj koji se ispisuje u terminal se može pogledati i u `leospec.log` fajlu nakon sinteze.

Sintetisana Verilog netlista brojača (`counter.gate.v`):

```
module COUNTER ( clk, rst, en, cnt ) ;

    input clk ;
    input rst ;
    input en ;
    output [3:0]cnt ;

    wire NOT_rst, nx26, nx32, nx83, nx93, nx103, nx113, nx125, nx132, nx134,
         nx136, nx146;
    wire [3:0] dummy ;

    dffr reg_cnt_0 (.Q (cnt[0]), .D (nx83), .CLK (clk), .R (NOT_rst)) ;
    inv01 ix126 (.Y (nx125), .A (en)) ;
    inv01 ix128 (.Y (NOT_rst), .A (rst)) ;
    dffr reg_cnt_1 (.Q (cnt[1]), .D (nx93), .CLK (clk), .R (NOT_rst)) ;
    oai21 ix94 (.Y (nx93), .A0 (nx132), .A1 (nx125), .B0 (nx136)) ;
    oai21 ix133 (.Y (nx132), .A0 (cnt[0]), .A1 (cnt[1]), .B0 (nx134)) ;
    nand02 ix135 (.Y (nx134), .A0 (cnt[1]), .A1 (cnt[0])) ;
    nand02 ix137 (.Y (nx136), .A0 (cnt[1]), .A1 (nx125)) ;
    dffr reg_cnt_2 (.Q (cnt[2]), .D (nx103), .CLK (clk), .R (NOT_rst)) ;
    mux21 ix104 (.Y (nx103), .A0 (cnt[2]), .A1 (nx26), .S0 (en)) ;
    xnor2 ix27 (.Y (nx26), .A0 (cnt[2]), .A1 (nx134)) ;
    dffr reg_cnt_3 (.Q (cnt[3]), .D (nx113), .CLK (clk), .R (NOT_rst)) ;
    mux21 ix114 (.Y (nx113), .A0 (cnt[3]), .A1 (nx32), .S0 (en)) ;
    xnor2 ix33 (.Y (nx32), .A0 (cnt[3]), .A1 (nx146)) ;
    nand03 ix147 (.Y (nx146), .A0 (cnt[2]), .A1 (cnt[1]), .A2 (cnt[0])) ;
    xor2 ix84 (.Y (nx83), .A0 (cnt[0]), .A1 (en)) ;

endmodule
```

Po završetku sinteze u fajlovima sa ekstenzijom `area.rpt` i `delay.rpt` se nalaze detaljni izveštaji vezani za površinu i kašnjenje. U `delay.rpt` treba proveriti da li kolo ispunjava zadata ograničenja po pitanju kašnjenja. Kao mera ispunjenosti zahteva definiše se tzv. SLACK parametar kao SLACK = RT – AT, gde je RT (Required Time) zadato/zahtevano vreme od strane projektanta, a AT (Arival Time) vreme pristizanja signala koje alat procenjuje na onsnovu podataka o kašnjenjima iz bibliotečkog fajla (`gdk.syn`). Pojednostavljeno govoreći, pozitivan SLACK (ili pozitivna vrednost SLACK parametra) znači da su ulazi svih registara (flip-flop) postavljeni i dostupni pre pojave aktivne ivice taktnog signala. Praktično, zadavanjem ograničenja po pitanju kašnjenja projektant postavlja ciljne vrednosti za proces optimizacije koji se odvija u samom alatu. Na osnovu ovih ograničenja alat iterativno pokušava da obezbedi uslov AT < RT instanciranjem odgovarajaućih ćelija (veća ćelija veće kašnjenje, ali bolje drajverske karakteristike (veći fanout)) iz dostupne biblioteke standardnih ćelija. Pošto na ovom nivou projektovanja nemamo informaciju o vezama (dužina, tip, nivo metala, stablo takta, RLC raspodeljeni parametri, itd.) alat jedino može da optimizije na osnovu informacija o kašnjenjima individualnih ćelija. Zato je obično dobra praksa da se na ovom nivo projektovanja ostvari dovoljno velika margina (tj. dovoljno velika vrednost SLACK parametra) kako bi nakon implementacije (faza kreiranja lejauta) kolo i dalje ispunjavalo ograničenja po pitanju kašnjenja. Kolika će ta margina biti odlučuje projektant na osnovou iskustva i dokumentacije za ciljani tehnološki porces. Obično je ta vrednost neki procenat periode taktnog signala.

Ograničenja se obično postavljaju od ulaza ka svim izlazima (`all_outputs` je komanda alata koja vraća listu svih izlaza kola). Putanja od nekog od ulaza ka nekom od izlaza koja ima najveće kašnjenje se naziva kritična putanja. Može se desiti da kritčna putanja bude razgranata po više

internih putanja. Ove interne putanje se takođe jednostanvo nazivaju kritične putanje. Prema tome da bi projekat ispunio ograničenja po pitanju kašnjenja, sve vrednosti SLACK parametra treba da imaju pozitivnu vrednost za sve kritične putanje. Ukoliko to nije slučaj, relaksirati ograničenja za optimizaciju kašnjenja smanjivanjem frekvencije (`FCLK`), ili povećavanjem intervala kašnjenja (`TCLK_PROC`). Informacija o kašnjenjima po pojedinim kritičnim putanjama je dostupna u `delay.rpt` nakon procesa sinteze. Kada se dobije sintetisano kolo koje ispunjava ograničenja po pitanju kašnjenja može se preći na naredni korak tj. implementaciju.

Post-synthesis simulacija (ModelSim)

Nakon postupka sinteze dobra je praksa još jednom potvriti funkcionalnost kola pre postupka implementacije (back-end). Pošto sintetisana Verilog netlista (`.gate.v`) predstavlja strukturni opis koji sadrži samo instance standardnih ćelija potrebno je HDL modele standardnih ćelija kompajlirati u odgovarajuću biblioteku. Biblioteka `$HOME/sim/gdk`. sadrži prethodno kompajlirane HDL modele standarndih ćelija. Ovu biblioteku treba linkovati prilikom simlulacije sintetizovane netliste (swithc `-L` u `vsim` komandi simulatora).

Za simulaciju sintetizovane netliste može se koristiti `counter-syn.do` čiji je sadržaj dat u nastavku.

Napomena: Prilikom rada na sopstvenom projektu prekopirati `counter-syn.do` fajl i izmeniti deo označen kurzivom bold.

DO fajl za simulaciju sintetizovane netliste (`counter-syn.do`):

```
# User defined variables
set DESIGN counter
set TB COUNTER_TB
set TB_FILE counter_tb.v

# Define & Map work library
vlib syn
vmap work syn

# Compile HDL files
vlog +define+SYN ../src/$DESIGN/$TB_FILE
vlog ../syn/net/$DESIGN.gate.v

# Set the Top-Level module/entity for the simulation
vsim -L gdk work.${TB}


# Define which ModelSim windows to view
view structure
view signals
view wave

# Add signals to the waveform
add wave -label Clock DUT:clk
add wave -label Reset DUT:rst
add wave -label Enable DUT:en
add wave -label Counter -radix unsigned DUT:cnt

# Run simulation
```

```
run -all

# Zoom to fit waves to the window
wave zoom full
```

Treba primetiti da je sintetizovano kolo instacirano u testbenč pomoću kompajlerske opcije (switch) `+define+SYN` kojom je definsana promenljiva `SYN` (pogledati Verilog kôd testbenča). Rezultati simulacije treba da budu identični onima dobijenim na osnovu simulacije RTL modela. U suprotnom, treba se vratiti na prethodni korak i proveriti da li je za top-level izabran pravi modul/entitet.

Učitavanje sintetizovane Verilog netliste (Pyxis Design Architect)

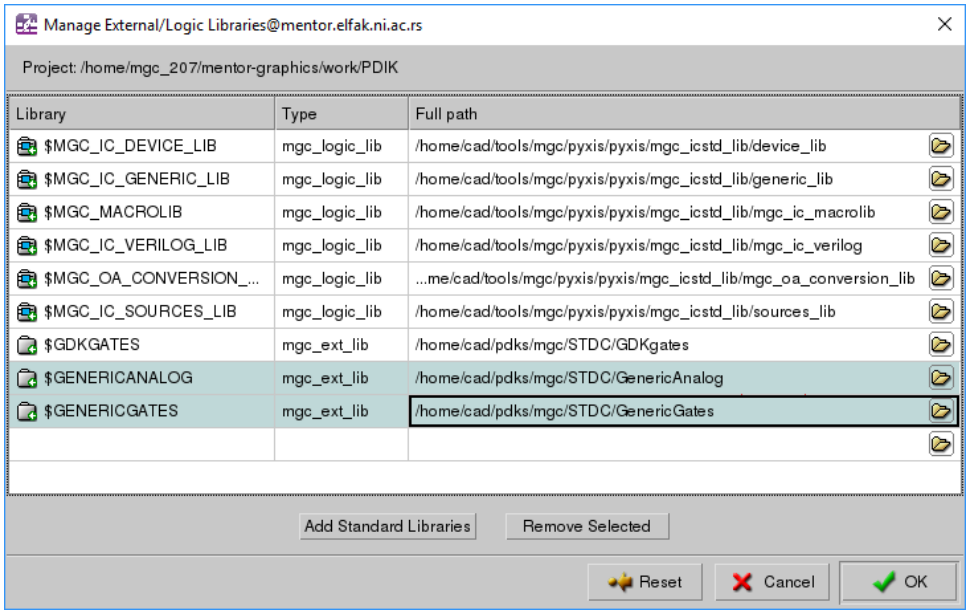
Pozicionirati se u `$HOME/mentor-graphics/work` i pokrenuti Pyxis Project Manager pomoću komande

```
dmgr_ic &
```

U stablu sa leve strane selektovati projekat (projekat je kreiran u prethodnom delu kursa) i importovati biblioteku standardnih ćelija pomoću `Main Menu → Edit → External/Logic Libraries`. Pritisnuti na browse dugme na kraju poslednjeg reda u tabeli i navesti putanju do odgovarajuće biblioteke. Uključiti sledeće biblioteke:

```
/home/cad/pdks/mgc/STDC/GDKgates
/home/cad/pdks/mgc/STDC/GenericGates
```

kao što je prikazano na Sl. 2.



Sl. 2 Izgled forme za importovanje eksternih biblioteka

Nakon ove akcije u stablu sa leve strane će se pojaviti nova stavka, `GDKgates` pod kojom se nalaze šematik i lejaut prikazi standardnih ćelija dostupnih u GDK-u. Na ovaj način su biblioteke standardnih ćelija vidljive u Pyxis okruženju.

Ponovo selektovati projekat u stablu i krierati novu biblioteku pomoću `Main Menu → File → New → Library` i staviti

```
Library Name: IC
```

Zatim selektovati biblioteku IC u stablu, pritisnuti desni taster miša, iz padajućeg menija izabrati `New → Schematic`. i staviti:

```
Schematic name: schematic
Cell name: COUNTER
```

Napomena: Za ime ćelije, `Cell Name`, obavezno staviti ime top-level modula u sintetisanoj Verilog netlisti (obično isto kao i ime top-level modula/entiteta u izvornoj HDL netlisti)

Nakon ove akcije otvoriće se prozor Pyxis Schematic editora sa praznim šematikom. Zatvoriti prazan list šematika (`COUNTER sheet1`). Učitati Verilog netlistu odabirom sledeće opcije: `Main Menu → File → Import Verilog`. Formu popuniti na sledeći način:

```
Netlist Files: $HOME/hdl/syn/net/counter.gate.v
Mapping Files: $GDKGATES/GDKgates_utillities/hdl_libs/gdk_map.vmp
Output Directory: $PDIK_2016_2017/IC
Replace all Schematics and Symbols : selected
Keep Intermediate Files : selected
```

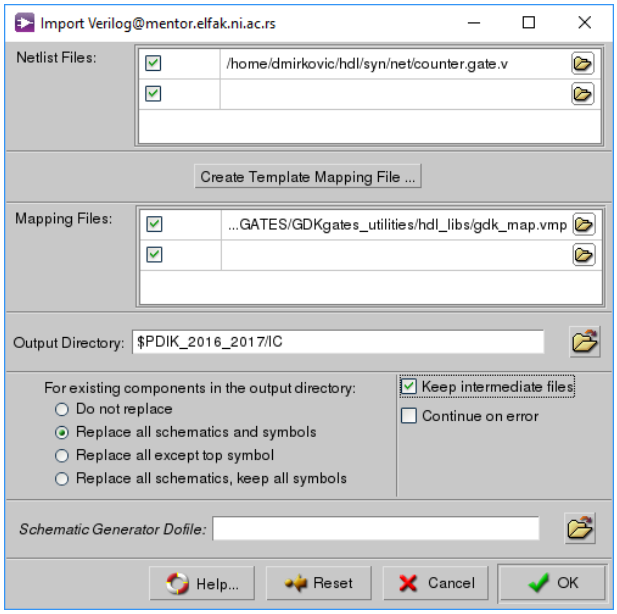
Izgled popunjene forme za učitavanje sintetizovane Verilog netliste je prikazan na Sl. 3.

Ukoliko je sve u redu, u `Message Area` Pyxis Schematic prozora treba da stoji poruka:

```
Note: Importing Verilog …
Note: Import Verilog completed. See session transcript for details.
```

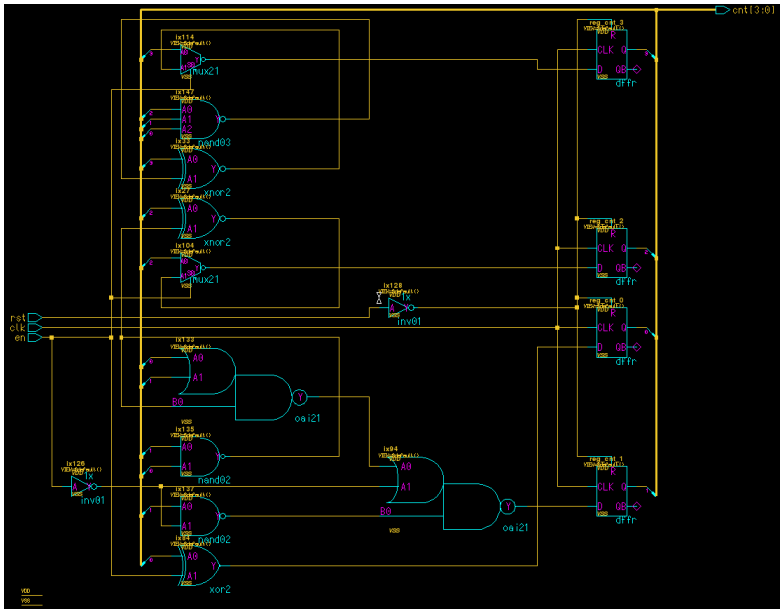
Lokacija transkripta (log fajla) se može videti u `Message Area` Pyxis Project Manager prozora gde npr. može da piše

```
Note: Transcript for /home/cad/tools/mgc/pyxis/pyxis/bin/da_ic
will be written to
/home/cad/tools/mgc/pyxis/pyxis/tmp/dmirkovic_da_ic_1493513044_1.txt
```



Sl. 3 Izgled forme za učitavanje sintetizovane Verilog netliste

Ukoliko dođe do problema prilikom učitavanja Verilog netliste pogledati u transkript fajl i videti detaljniji opis greške. Ispraviti grešku i ponoviti postupak za učitavanje Verilog netliste. Nakon uspešnog učitavanja Verilog netliste, otvoriti šematik izborom sledeće opcije u Pyxis Shematic editoru `Main Menu → Open`, a zatim izabrati `IC → COUNTER → Schematic`. Šematik brojača nastao kao rezultat učitavanja sintetizovane Verilog netliste je prikazan na Sl. 4.



Sl. 4 Šematik prikaz sintetizovane Verilog netliste u Pyxis Schematic okruženju

Nakon vizuelne provere šematika, odabrati opciju `check/save` (plava ikonica u obliku diskete sa zelenim znakom za štikliranje ili `Main Menu → File → Check Schematic`). U `Report` prozoru ne bi trebalo da ima grešaka (upozorenja se, uglavnom, mogu ignorisati). Ukoliko ima, ispraviti greške i ponovo verifikovati šematik opcijom `check/save`. Zatvoriti Pyxis Schematic

prozor. Treba primetiti da se pored šematika automatski generiše i simbol. Više informacija o Pyxis Schematic editoru se može naći u

```
pyxis_schematic_qref.pdf
pyxis_schem_ref.pdf
pyxis_schem_user.pdf
```

pod `/home/docs` direktorijumom.

Generisanje lejauta (Pyxis IC Station)

Na osnovu šematika sada se može pristupiti projektovanju lejauta (back-end). Kao i u prethodnom delu kursa (projektovanje standardne ćelije) biće primenjena SDL (Schematic Driven Layout) metodologija s tim što će u kasnijoj fazi biti intezivno korišćene opcije za automatski razmeštaj (floor plan) i povezivanje (routing).

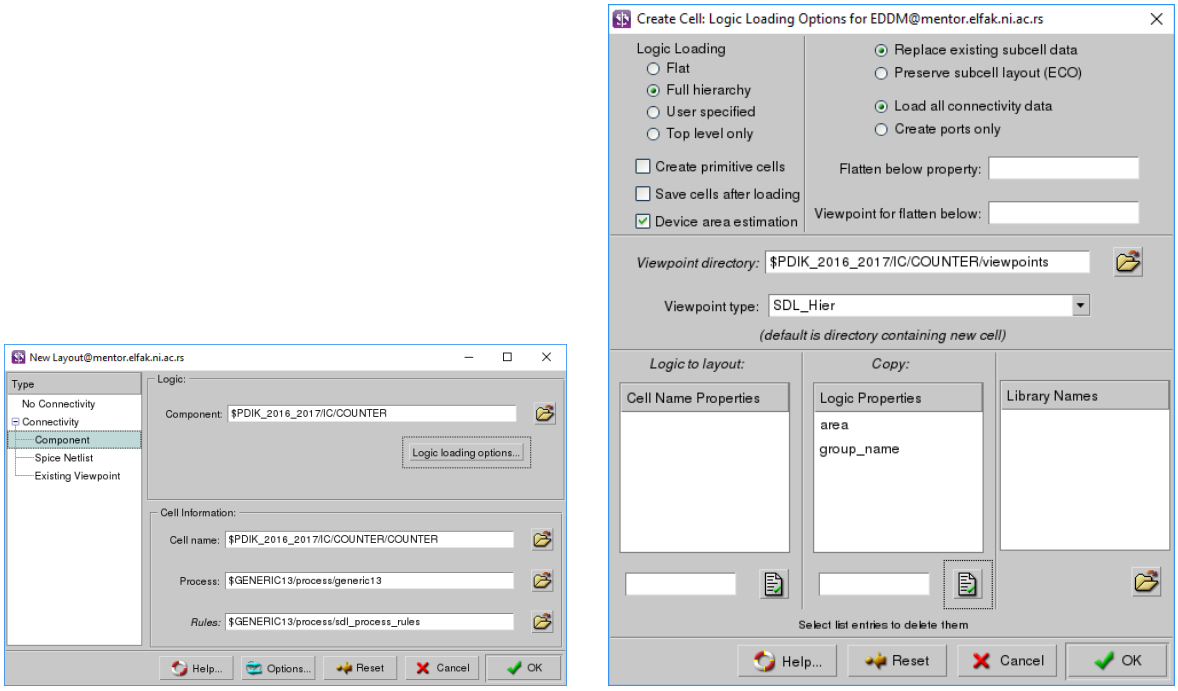
U stablu Pyxis Project Managera selektovati ćeliju `COUNTER`, pritisnuti desni taster miša, i odabrati opciju `New → Layout`, ostaviti sve difoltno:

```
Layout name: COUNTER
Cell name: COUNTER
```

i odabrati `OK`. U `New Layout` prozoru pritisnuti `Logic loading options` i odabrati:

```
Device area estimation : selected
Viewpoint type : SDL_Hier
```

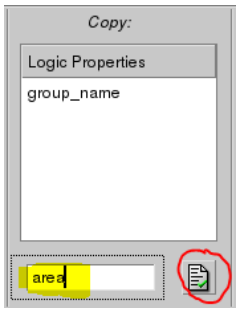
kao na Sl. 5b.



(a) (b)

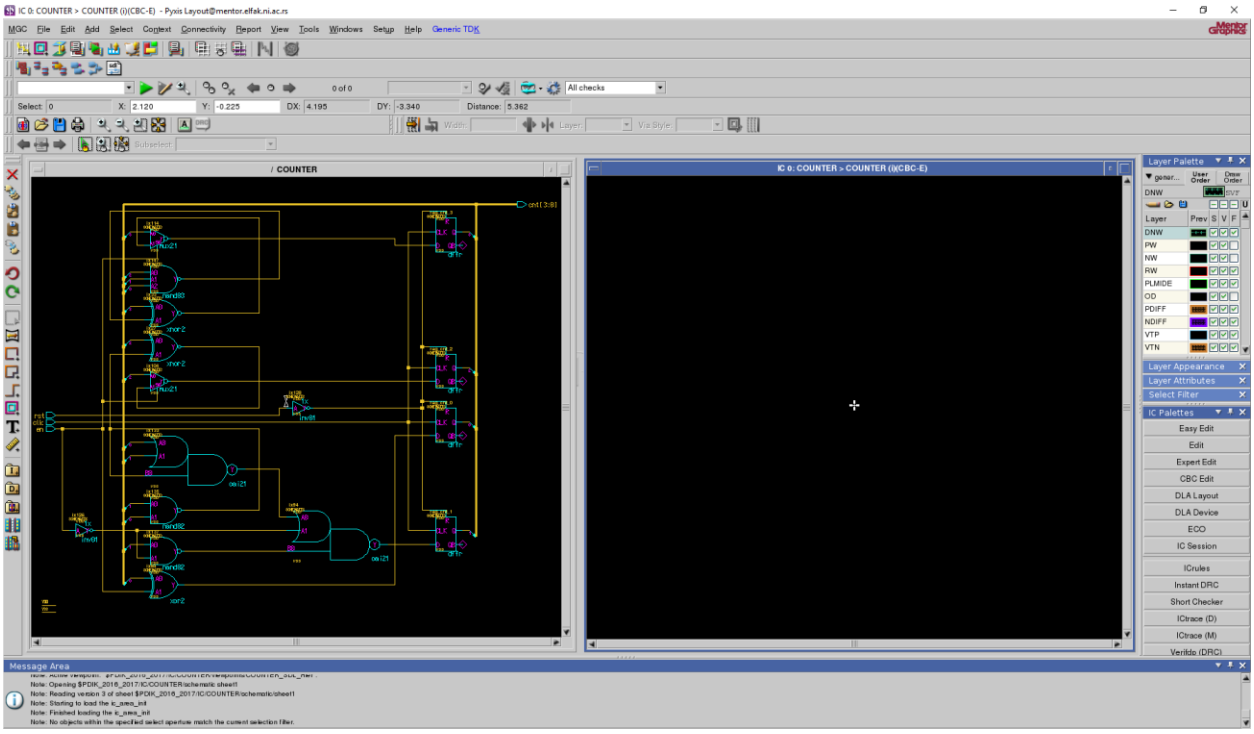
Sl. 5 Izgled formi za kreiranje novog lejauta

U formi `Logic loading options` pod sekcijom `Copy`: uneti novu stavku unošenjem naziva `area` i pritisnuti dugme označno na Sl. 6. Konačno pritisnuti dugme `OK` u obe forme sa Sl. 5.



Sl. 6 Unošenje nove stavke u Loading Logic Options formi

Nakon ove akcije otvoriće se dva prozora u Pyxis Layout editoru. Jedan sadrži neizmenljiv prikaz šematika (logic view), a drugi prostor za crtanje lejauta. Difoltno prozori su u Tabbed modu. Pošto će u početnoj fazi biti potrebna oba prozora dobro je prikazati ih istovremeno, jedan pored drugog. To se postiže sledećom opcijom `Main Menu → MGC → Setup → General → Workspace → View → Default → Left Right Tilling`. Izgled Pyxis Layout Prozora je prikazna na Sl. 7.



Sl. 7 Izgled Pyxis Layout prozora sa prikazom logike (levo) i prozora za crtanje lejauta (desno)

Napomena: Ukoliko se šematik iz nekog razloga difoltno ne otvori, može se otvoriti pomoću `Main Menu → Windows → Logic`. Alternativno može se koristiti i `IC Paletts → Plan & Place → Logic Source → Open`.

Pyxis Layout okruženje za rad je već poznato iz prvog dela kursa. Podsećanja radi ovde će biti date neke osnovne smernice za rad u Pyxis Layout editoru. Sve opcije (meniji, palete, prečice na tastaturi) su zavisne od konteksta. Naime, kada je u fokusu prozor za crtanje lejauta (desno) dostupne su jedne opcije, a kada je u fokusu šematik prozor (levo) onda su dostupne druge opcije, kada je u fokusu neka od paleta onda su, pak, dostupne drugačije opcije itd. Sa desne strane glavnog prozora može se uočiti nekoliko paleta koje sadrže dosta opcija. Svaka opcija otvara novu paletu ili aktivira neku komandu. U ovom slučaju najbitnija paleta za rad je `IC Paletts` . Pod njom se nalaze još dve bitne palete `Plan & Place` i `Route`. Dugmad za aktiviranje ovih paleta se nalaze na dnu palete `IC Palletts` pod sekcijom **`Pyxis Assemble`**. Prvo treba podesiti tzv. comaptibility editing opciju izborom `Main Menu → Generic TDK → Setup Pyxis Compatibility Editing`. Bitne prečice na tastaturi za rad u lejaut editoru su date u nastavku.

Prečica	**Akcija**
`f`	Fit to window
`m`	Move selected object
`v`	Vertical/Horizontal alignment of the selected object with the surrounding objects when in Move mode.
`c`	In general Copy object. Prevent vertical movement of the selected object when in Move mode.
`s`	Streach object
`z`	Zoom Area
`Shift + z`	Zoom out by two
`p`	Path
`r`	Rectangle
`q`	Quarey (Selected Object Properties)
`k`	Ruler
`Shift + k`	Remove all Rulers

Za kompletniji spisak prečica i detaljnije uputstvo za rad u Pyxis Layout editoru pogledati sledeće dokumente:

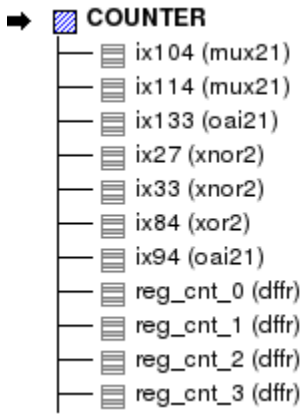
```
pyxis_layout_qref.pdf
pyxis_layout_ref.pdf
pyxis_layout_user.pdf
```

Više informacija o korišćenju GDK-a u Pyxis okruženju se može naći u:

```
pyxis_quickstart.pdf
Pyxis_SPT_HEP.pdf
```

Svi dokumenti se nalaze pod `/home/docs` direktorijumom. Ukoliko prilikom rada dođe do nepredviđenih problema čitalac se savetuje da konsultuje gore pomenutu dokumentaciju.

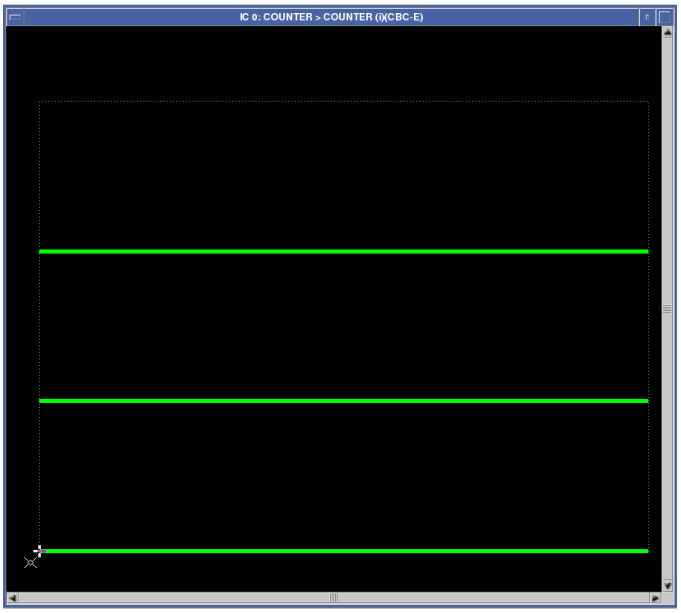
Selektovati prozor za crtanje lejauta i otvoriti prozor za prikaz hijerarhije pomoću `Main Menu → Windows → Hierarchy Window`. Verifikovati da se pod top-level modulom vide samo instance standardnhi ćelija kao na Sl. 8



Sl. 8 Izgled hijerarhije projekta sa standardnim ćelijama

Ukoliko je sve u redu zatvoriti `Hierarchy Window` prozor.

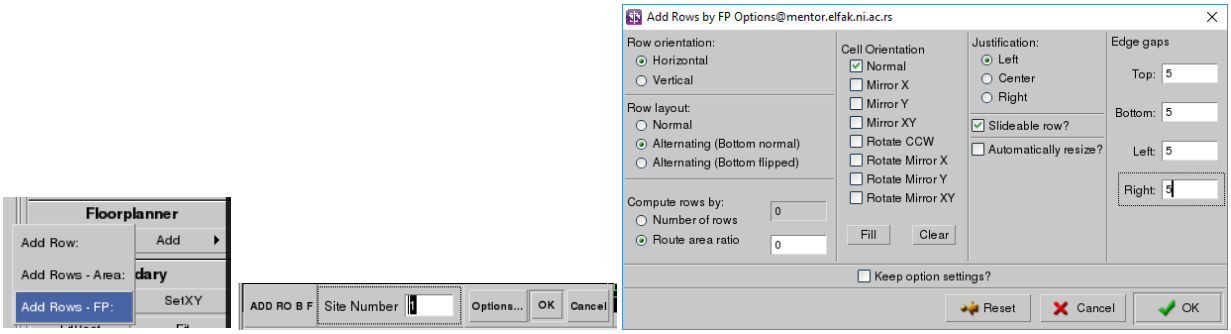
Vratiti se ponovo na prozor za crtanje lejauta. Otvoriti paletu `Plan & Place` pomoću `IC Paletts → Plan & Place`. U `Plan & Place` paleti pritisnuti dugme `Autofp` pod sekcijom **`Floorplanner.`** Nakon ove akcije lejaut će biti izdeljen na odgovarući broj vrsta (rows), Sl. 9.



Sl. 9 Izgled lejauta nakon automatskog razmeštaja vrsta za smeštanje standardnih ćelija (floorplan)

Ovo je najjednostavniji način za automatsko planiranje razmeštaja. Naravno, postupak se može detaljnije podesiti kroz opcije koje su dostupne pod `Plan & Place → Floorplanner → Add → Add Row -FP:`. Da bi se došlo do opcija za kreiranje redova za smeštanje standarndih ćelija potrebno je zadržati levi taster miša na dugmetu `Add`, a zatim iz padajućeg menija odabrati stavku `Add Row -FP:`. (Sl. 10a). Nakon ove akcije pojaviće se pod-prozor u donjem levom uglu prozora za crtanje lejauta pod nazivom `ADD RO B F` (Sl. 10b). U pod-prozoru `ADD RO B`

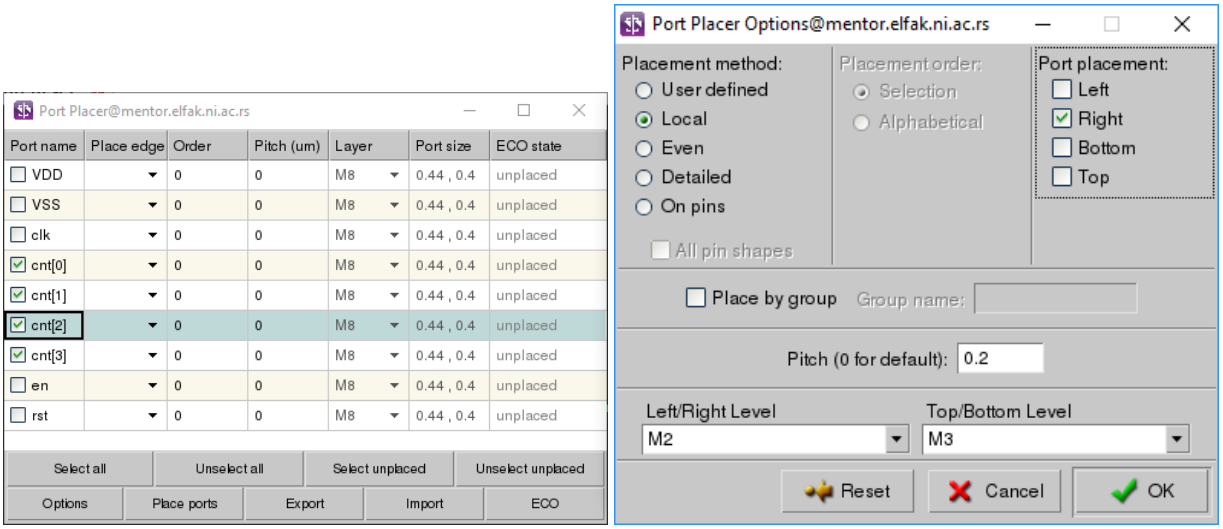
`F` odabrati `Options`, nakon čega se otvara forma `Add Rows by FP Options` prikazana na Sl. 10c. Za projektovani brjač izabrane su opcije prikazane na Sl. 10c. Naravno, može se eksperimentisati i sa ostalim opcijama/stavkama dostunim u meniju sa Sl. 10a.



(a) (b) (c)

Sl. 10 Podešavanja opcija za autoatsko pnairanje razmeštaja (floor plan): (a) ativiranje Add Rows alata, (b) pod-prozor ADD RO BF i (c) Opcije za dodavanje redova u koje se smeštaju standardne ćelije

Pre razmeštaja standardnih ćelija treba pozicionirati portove top-level ćelije. Pokrenuti `Port Placer` izborom sledeće opcije `Main Menu → Setup → Windows → Port Placer`. Na Sl. 11 je prikazan izgled `Port Placer` prozora (Sl. 11a) zajedno sa formom za podešavanje opcija `Port Placer Options` (Sl. 11b). Do forme sa Sl. 11b se dolazi pritiskom na dubme `Options` u dnu `Port Placer` prozora.



(a) (b)

Sl. 11 Izgled Port Placer prozora (a) i forme za definisanje opcija Port Placer alata (b)

Pomoću `Port Placer Options` forme može se postaviti jedinstveno pravilo za više selektovanih portova. Npr. na Sl. 11 prikazan je sledeći scenario. Selektovani bitovi magistrale `cnt` postavljaju se sa desne strane top-level ćelije (`Port placement → Right`) pri čemu je međusobni razmak između centara portova 0.2μm (`Pitch → 0.2`), a portovi će biti u metalu M2 (`Left/Right Level → M2`). Širina i dužina portova je difoltno podešena u samom GDK-u i iznosi 0.44 × 0.4μm (pretposlednja kolona u formi `Port Placer`).

Ovaj metod je pogodan kada je u pitanju manji broj portova. Međutim, za veći broj portova pogodnije je pripremiti odgovarajući skript fajl kojim se definiše razmeštaj portova. Praktično, ovaj fajl sadrži AMPLE[1] komande (funkcije) kojima se `Port Placer` alatu izadaju direktive za pozicioniranje portova. Primer AMPLE skripta za pozicioniranje portova brojača dat je u nastavku.

AMPLE skript za pozicioniranje portova borjača (`counter.pinplacement.ample`):

```
$set_row_data( "clk", "Left", 1, 0, "M2", [0.44 , 0.4] );
$set_row_data( "rst", "Left", 2, 0, "M2", [0.44 , 0.4] );
$set_row_data( "en" , "Left", 3, 0, "M2", [0.44 , 0.4] );
$set_row_data( "cnt[0]", "Right", 4, 0, "M2", [0.44 , 0.4] );
$set_row_data( "cnt[1]", "Right", 5, 0, "M2", [0.44 , 0.4] );
$set_row_data( "cnt[2]", "Right", 6, 0, "M2", [0.44 , 0.4] );
$set_row_data( "cnt[3]", "Right", 7, 0, "M2", [0.44 , 0.4] );
$set_row_data( "VDD", "", 8, 0, "M2", [1] );
$set_row_data( "VSS", "", 9, 0, "M2", [1] );
```

Skrip se nalazi pod `$HOME/pyxis` direktorijumu.

Napomena: Kada je `Pitch` jednak nuli koristi se difoltno setovanje zadato u GDK-u.

Argumenti AMPLE funkcije `$set_row_data()` odgovaraju hederu tabele u prozoru `Port Placer` sa Sl. 11a. AMPLE skript se učitava izborom opcije `Import` u `Port Placer` prozoru. Nakon učitavanja `counter.pinplacement.ample` skripta, i instaciranja portova `Port Placer` prozor izgleda kao na Sl. 12.

Port Placer

Port name	Place edge	Order	Pitch (um)	Layer	Port size	ECO state
clk	Left	1	0	M2	0.44 , 0.4	placed
rst	Left	2	0	M2	0.44 , 0.4	placed
en	Left	3	0	M2	0.44 , 0.4	placed
cnt[0]	Right	4	0	M2	0.44 , 0.4	placed
cnt[1]	Right	5	0	M2	0.44 , 0.4	placed
cnt[2]	Right	6	0	M2	0.44 , 0.4	placed
cnt[3]	Right	7	0	M2	0.44 , 0.4	placed
VDD		8	0	M2	1	unplaced
VSS		9	0	M2	1	unplaced

Sl. 12 Izgled Port Placer prozora nakon učitavanja AMPLE skripta za pozicioniranje portova

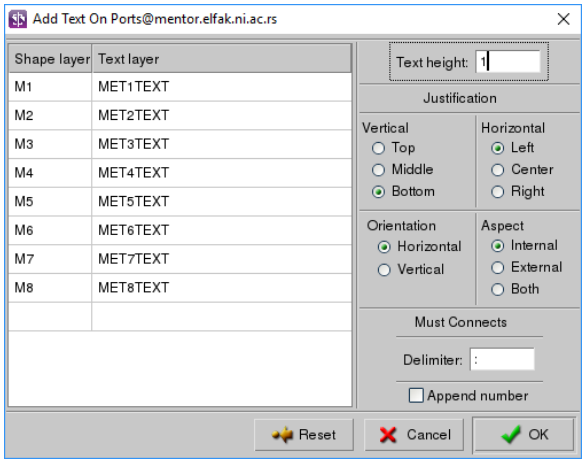
Kada se odabere zadovoljavajući raspored portova (bilo AMPLE skriptom ili putem forme `Port Placer Options`) pritisnuti dugme `Place ports` u dnu `Port Placer` prozora. Treaba uočiti da su portovi za napajanje izostavljeni pošto je praksa da se oni potavljaju direktno na prstenove za napajanje (power rings). Portovi za napajanje će biti postavljeni kasnije, nakon rutiranja prstenova za napajanje. ECO (Engineering Change of Order) daje informaciju o statusu porta. U ECO polju se mogu pojaviti sledeće poruke:

[1] AMPLE (Another Mentor Programming Language) je jezik kojim je realizovan Mentor Graphics Falcon platforma na kojoj su izgrađene sve Mentor Graphics aplikacije.

Poruka	Značenje
`unplaced`	Port se nalazi u lejautu (layout view) i nije postavljen na zadato mesto, a pri tome ima odgovarajuću predstavu (simbol) u šematiku (logic view).
`placed`	Port se nalazi u lejautu i postavljen je na zadato mesto, a pri tome ima odgovarajuću predstavu u šematiku.
`missing`	Port ima odgovarajuću predstavu u šematiku ali ne i u lejautu.
`Not in logic`	Port ima odgovarajuću predstavu u lejautu ali ne i u šematiku.

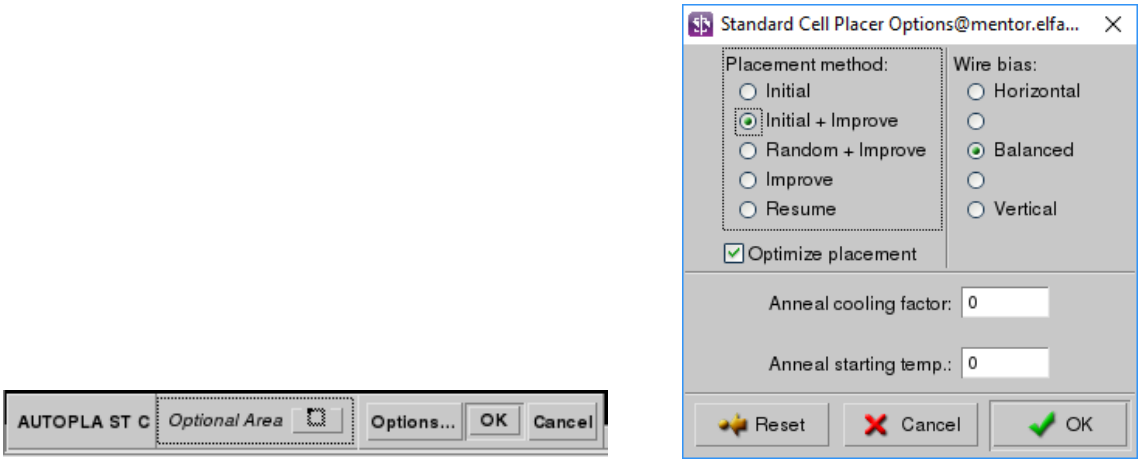
Napomena: Ukoliko se u ECO polju pojavi poruka `missing` ili `Not in logic` onda je najverovatnije došlo do greške u nekom od prethodnih koraka (HDL opis, sinteza, importovanje Verilog netliste). Prvo traba proveriti HDL opis kola pre sinteze i uočiti da li ima nekih neregularnosti tj. proveriti da li je HDL opis prilagođen sintezi (npr. da li su korišćeni sintetazibilni tipovi podataka). Zatim treba proveriti da li je u procesu sinteze došlo do neke čudne optimizcije signala koji je vezna za port za koji alat prjavljuje da je u stanju `missing` ili `Not in logic`. To se može utvrditi vizuelnom inspekcijom sintetizovane Verilog netliste. Konačno može se desiti da je u pitanju i bag samih alata i u tom slučaju kontaktirati predmetnog asistenta.

Kada se proces instanciranja portova završi, dodati labele (tekst) na portove pomoću opcije `Main Menu → Add → Text on Ports`. U `Add Text On Ports` formi u polju `Text height` staviti najmanje 1 (visina teksta 1μm), Sl. 13. Nakon pritiska na dugme `OK` svi instancirani portovi će imati labele.



Sl. 13 Izgled forme za automatsko instaciranje labela na portove

Sledeći korak je automatsko instanciranje standardnih ćelija. Standardne ćelije se instaciraju pomoću opcije `IC Paletts → Plan & Place → Auto Placement → StdCell`. Nakon ove akcije otvara se pod-prozor `AUTOPLA ST C` u donjem levom uglu prozora za crtanje lejauta (Sl. 14a).

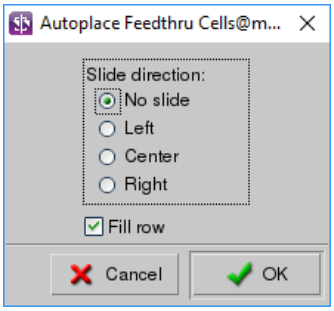


(a) (b)

Sl. 14 Izgled pod-prozora u prozoru za crtanje lejauta (a) i forme za podešavanje opcija Standard Cell Placer alata (b)

Algoritam (`Placement method`) razmeštanja ćelija se može odabrati u formi `Standard Cell Placer Options`, Sl. 14b. Do ove forme se dolazi pritiskom na dugme `Options` u `AUTOPLA ST C` pod-prozoru. Za više informacija o algoritmima za razmeštaj ćelija pogledati `pyxis_layout_user.pdf` dokument. U konkretnom projektu brojača izabrane su opcije prkazane na Sl. 14b. Naravno, može se eksperimentisati i sa drugim opcijama. Nakon pritiska na dugme `OK`, standardne ćelije će biti instancirane u prethodno definisane vrste.

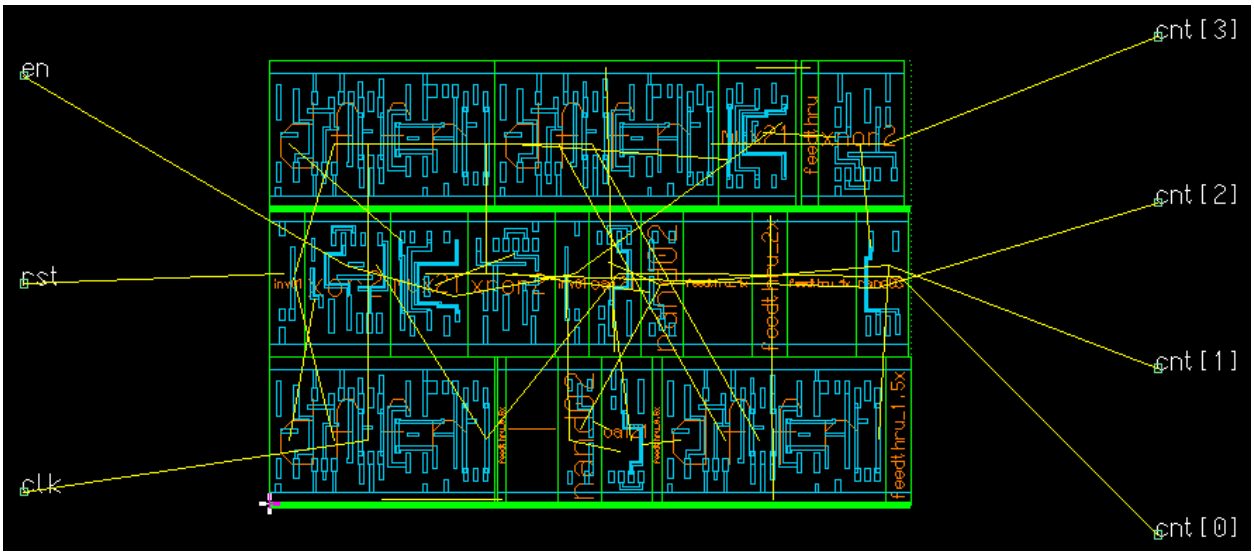
Kada se instanciranje ćelija završi treba popuniti prazan prostor u vrstama (nije dobra praksa da ostane prazan prostor na čipu između ćelija). U te svrhe se koriste tzv. filler ćelije. Filler ćelije mogu sadržati razne pasivne komponente (Šotki diode, filtarske kondenzatore za napajanje, kontakte ka Well-ovima (Tap ćelije) itd.). GKD podržava samo tzv. feedthrou tip filler ćelija. Ove ćelije obezbeđuju samo kontinuitet linija za napajanje i Well-ova. Feedthrou ćelije se automatski instanciraju pomoću opcije `IC Paletts → Plan & Place → Auto Placement → Feed` nakon čega se otvara forma prikazana na Sl. 15.



Sl. 15 Izgled forme za podešavanja opcija za automatsko instaciranje feedthour ćelija

Za brojač je izabrana opcija opcija `No slide` tako da položaj prethodno instanciranih, standardnih ćelija, ostaje nepromenjen nakon insatcniranja feedthporu ćelija. Može se izabrati i neka od opcija koja će pomeriti standardne ćelije u levo (`Left`), ka centru lejauta (`Center`) ili u desno (`Right`).

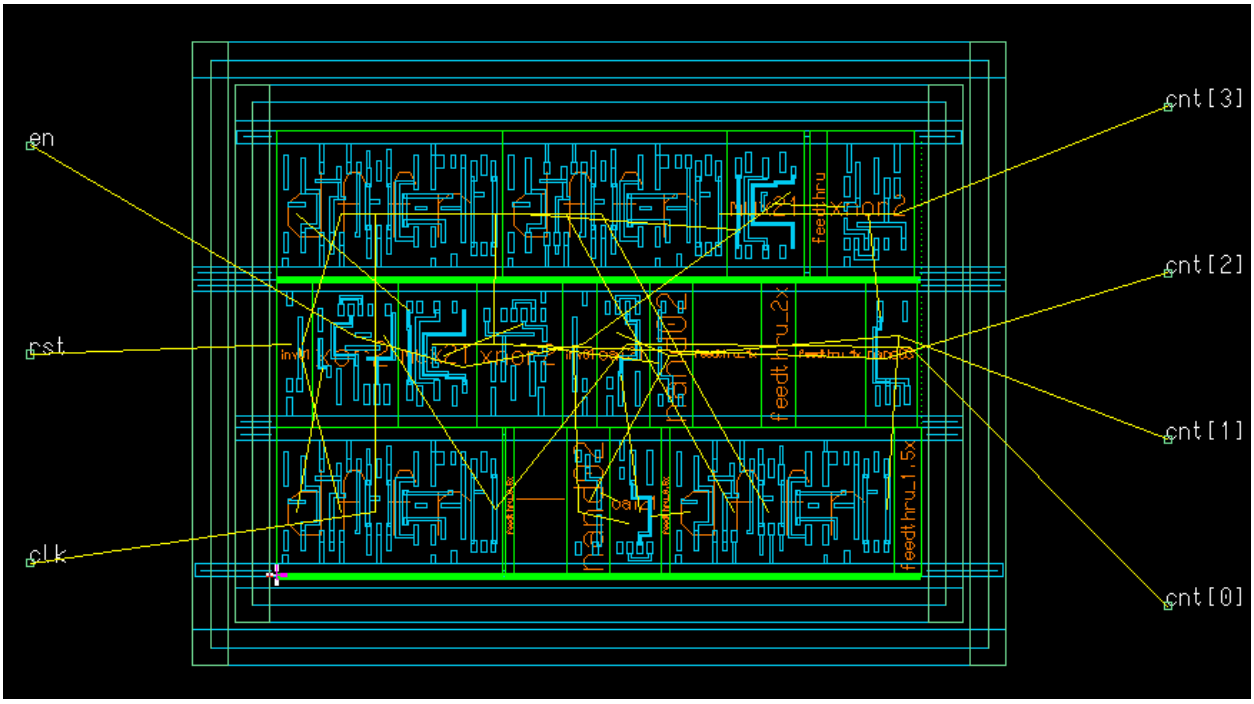
Izgled lejauta nakon automatskog planiranja razmeštaja i instanciranja portova, standardnih i filler ćelija je prikazan na Sl. 16.



Sl. 16 Izgled lejauta nakon planiranja razmeštaja i instaciranja portova, standardnih i feedthrou ćelija.

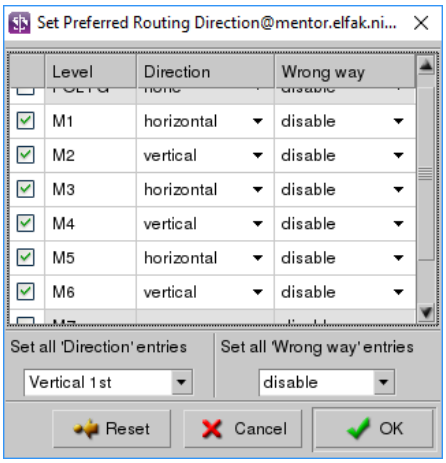
Sledeći korak je rutiranje. Za automatsko rutiranje se koristi alat `ARoute`. Do palete ovog alata se može doći diretkno iz `Plan & Place` palete pritiskom na dugme `Route` (dugme se nalazi na vrhu `Plan & Place` palete). Alternativno, pritiskm na dugme `Top` aktivira se `IC Paletts` paleta koja je najviša po hijerarhiji, a onda se odatle može ići na `IC Paletts → Route`.

Prvo treba rutirati napajanje. Za rutiranje prstenova za napajanje izabrati `ARoute → Power Routing → Run`. Debljina prstenova za napajanje je definisana u GDK-u. Pošto su linije za napajanje u standardnim ćelijam rutirane u metalu M1, onda su sve vertikalne linije u prstenovima za napajanje rutirane u metalu M2, a sve horizontalne u metalu M1. Izgled lejauta nakon rutiranja napajanja je prikazna na Sl. 17.



Sl. 17 Izgled lejauta nakon rutiranja napajanja

Odabrati opciju `ARoute → ARoute Setup → Direction` nakon čega se otvara forma sa Sl. 18.



Sl. 18 Forma za odabir smera rutiranja i nivoa metala za rutiranje.

Čekirati samo nivoe metala M1 do M6 i pritisnuti dugme `OK`. Top-level metali M7 i M8 su obično rezervisani za povezivanje stopica (pads) ili za realizaciju blok kondenazatora. POLYG se koristi za gejt tranzistora pa je rutiranje u ovom nivou obično zabranjeno.

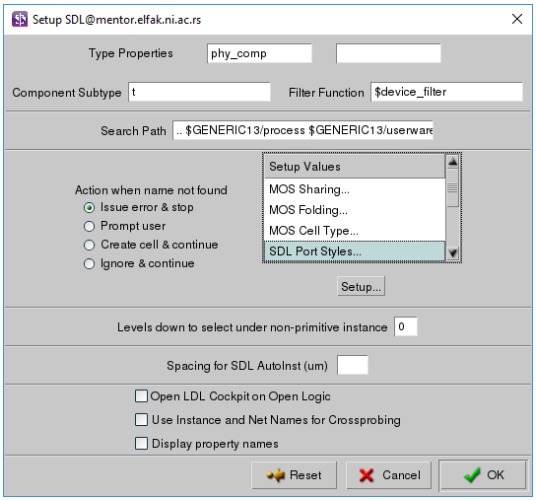
Sada se može rutirati ostatak veza. Odabrati opciju `ARoute → ARoute Setup → Overflows` i zadržati levi taster miša, a zatim iz padajućeg menija odabrati `Nets` kao na Sl. 19.

Sl. 19 Pdešavanje netova za rutiranje u ARoute alatu

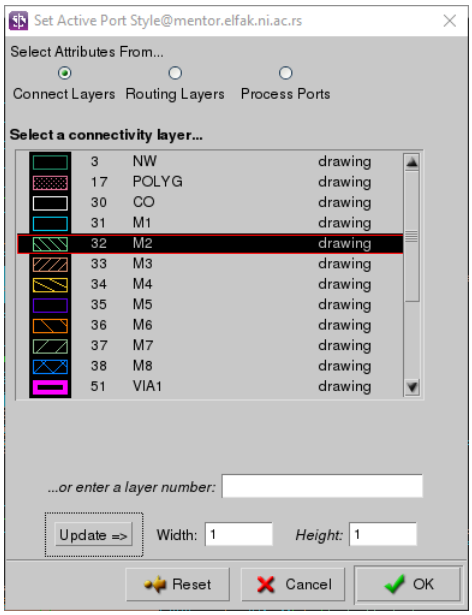
Nakn podešavanja netova za rutiranje aktivirati autoruter zadavanjem komande `ARoute → ARoute Commands → Run`. Sačekati da se rutiranje završi. Proces rutiranja može da traje od reda desetina minuta do reda sati zavisno od kompleksnosti projekta.

Kada se rutiranje završi, proveriti da li u lejautu ima nedovršenih veza tzv. overflows. Izborom opcije `ARoute → Routing Results → SOvrf` mogu se selektovati svi overflows (dugme se nalazi pri dnu `ARoute` palete). Overflows su nedovršene veze (veze koje autoruter nije mogao da realizuje) i ukoliko ih ima, nakon ove akcije biće označene podebljanim, žutim, linijama u lejautu (thick fly lines). Za jednostavnije projekte ne očekuje se da bude nedovršenih vezna. Ukoliko ih ima, odabrati `Overflows` iz padajućeg menija sa Sl. 19, aktivirati opciju `ARoute → ARoute Region → Define` i selektovati region gde se nalazi overflow koji se želi rutirati. Nakon toga pokrenuti ponovo autoruter sa `ARoute Commands → Run`. Postupak ponavljati sve dok svi overflows ne budu rutirani. Alternativni načini autorutiranja (ili interaktivnog rutiranja alatom `IRoute`) overflow-a se mogu pogledati u `Pyxis_SPT_HEP.pdf` str. 472 - 484.

Konačno, ostaje da se instanciraju portovi za napajanje, `VDD` i `VSS`. Pre instaciranja treba izabrati opciju `Main Menu → Setup → SDL`. U formi `Setup SDL` odabrati `SDL Port Styles`, a zatim pritisnuti dugme `Setup` (Sl. 20a).



(a)



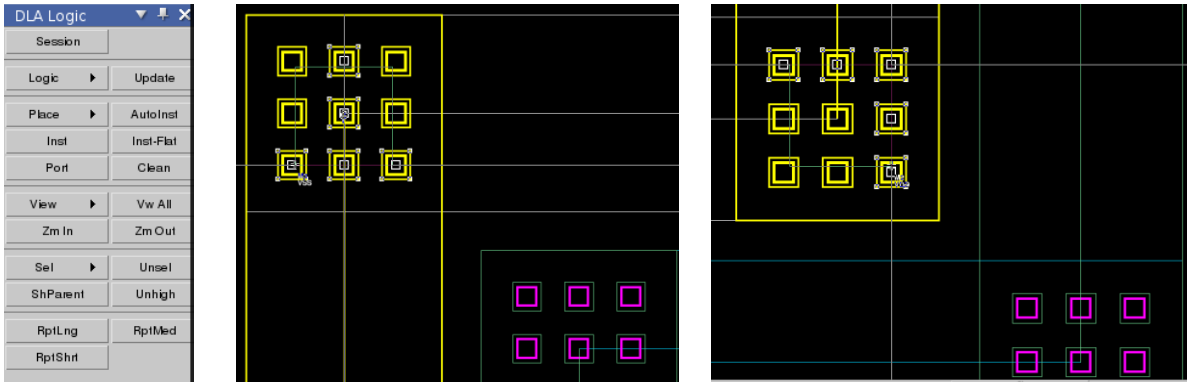
(b)

Sl. 20 Izgled formi za SDL podešavanja

U formi `Setup Active Port Style` selektovati metal M2, postaviti veličinu porta 1× 1μm (`Width: 1, Height: 1`) kao na Sl. 20b i pritisnuti dugme `OK` u obe forme.

Napomena: može se izabrati i metal M1 i/ili veća/manja veličina porta.

Selekotvati šematik prozor, zatim selektovati `wire` sa labelom `VSS`. U `DLA Logic` paleti (Device Level Automation, Sl. 21a) odabrati `Port`. Nakon ove akcije kursor miša se automatski pozicionira u lejaut, a za kursor miša biće prikačen port (kvadrat dimenzije 1× 1μm u metalu M2 kao što je ranije izabrano). Objekat na koji je dozvoljen staviti port je oivičen žutim linijama u lejautu (Sl. 21b i c)

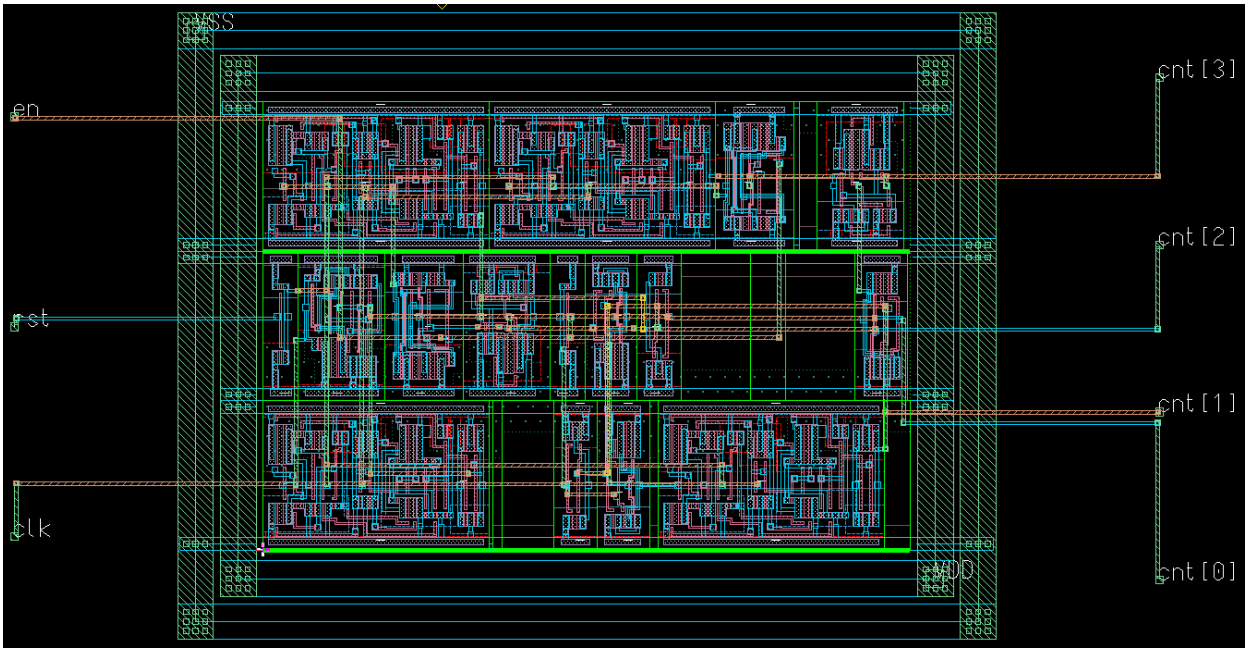


(a) (b) (c)

Sl. 21 Izgled DLA palete (a) i detalji u lejautu prilikom instaciranja VSS (b) i VDD (c) portova

Generalno, port se može postaviti u bilo kom delu objekta označenog žutom bojom. Međutim, često je praksa da se portovi za napajanje stave u suprotne uglove ćelije. Na ovaj način portovi za napajanje su lakše uočljivi prilikom instanciranja projektovane ćelije u lejaut više hijerarhije.

Konačni izgled lejauta kola brojača je prikazan na Sl. 22.



Sl. 22 Izgled lejauta četvorobitnog brojača

Da bi se videli svi nivoi hijerarhije i šrafure na svim lejerima koji imaju definsan fill pattern, na tastaturi treba pritisnuti taster `2`, a u paleti `Layer Palett` pritisnuti dva puta taster `f`.

Dalje se projektovani lejaut može verfikovati pomoću Calibre LVS/DRC alata (`Main Menu → Tools → Calibre`). Međutim, pošto se radi o studentskoj verziji GDK-a, Calibre DRC rezultuje velikim brojem grešaka od kojih neke nije moguće ispraviti pošto su deo samih ćelija.

Na kraju treba napomenuti da je Pyxis skup alata namenjen više projektovanju analognih ASIC kola, a ne digitalnih. Nažalost, alati koji u potpunosti podržavaju time/power constrained driven placement & routing poput Oasys-RTL (RTL kompajer) i Nitro-SoC (back-end alat za implementaciju) nisu pokriveni studentskom, HEP (Higher Education Program), licencom za 2017 godinu. Prema tome, postupak projektovanja digitalnog ASIC kola se ovde završava.