Technology Libraries (V1)

Written by: **Fatma Ali**

Content:

==========

Introduction:

اى design فى الاخر المفروض هيوصل لمرحلة transistor level عشان يتصنع وده بيكون اما بنطلع layout ونبعته للمصنع ... فالمصنع بيحدد technology معينة هيشتغل بيها او كذا واحدة عنده وانت تختار واحده منهم مثلا ... المهم ان فيه technology معينة هتشتغل عليها وفى حالة semi custom design زى ما احنا عارفين بيكون معايا library فيها cells مختلفة فى function و الحجم وانواع كتير منها combinational or sequential وباتالى ال tools الى هتحول ال RTL code بتاعى الى layout فى الاخر (على خطوات كتير من ASIC flow) لازم ادخلها فى كل خطوة ال library الى انا شغال عليها دى لان ده الى هصنع بيه فى الاخر فلازم ال design يكون معمول منها ومظبوط وبيحقق ال function المطلوبة & specifications

==========

Standard cell libraries:

- عندنا ال standard cell library ودى بيكون فيها cells معينة سواء combinational او sequential وفى بعض ال cells ممكن تكون فى technology library معينة ومش موجودة فى واحدة تانية مثلا وهكذا ... على حسب الى هتصنع بيها هدخلها لل tools فجى الى فيها ال gates الحقيقة الى هستخدمها بكل معلوماتها سواء Timing or power consumption or others

==========

Design ware library:

دى libraries بيكون فيها الحاجات المعقدة complex gates الى بتسهل عليا ال design بدل ما انا اعملها من الاول زى مثلا Controller فالحاجات دى محطوطة فى ال RTL بتاعى ك black box بالنسبالى ولكنها مش موجودة فى standard cell library ففى وقت ال synthesis بتروح تجيبها من Design ware library الى بدخلها لل synthesis tool

==========

Macro library:

ال library دى بتبقى blocks مش موجودة فى ال standard cell library ولكنها بتسهل ال design كتير بدل ما اروح اعمل انا الكود كله عشان ينفذ الى ال marcos زى مثلا memory ... بس دى بيكون فيها حاجات less complex عن الى فيه design ware

==========

Symbol library:

دى الى بستخدمها فى ال synthesis عشان لما احب اشوف ال netlist ك gates قدامى هحتاج اقول لل tool الى هى كاتباه فى صورة verilog netlist ده يترجم ازاى ل schematic او بمعنى اصح اسماء ال cells الى هى جابتها خلاص من standard cell library هتعرف شكلها ازاى عشان ترسمها ليا .... لازم اقولها ايه ال symbol بتاعها فدى هى ال library الى بتعمل كده

==========

libraries views

- ال library دى بيبعتها المصنع وفيها مجموعة files تتمثل فى three different views وكل view منهم بيستخدم فى وقت معين خلال ال ASIC flow ... ولكن فى الاخر كل ال views دى جواها نفس المعلومات ولكن على حسب ال tool بتفهم انى format فيهم

1- Behavioral view [std_cell.v / std_cell.vhd]

وده بيكون فى صورة vital [verilog/VHDL] وده بيستخدم فى Gate level simulation يعنى اما اجى اعمل simulation لل netlist الى هتتطلع من synthesis tool فبيكون ال domain بتاعها حسب لو مكتوبة ب verliog يبقى (v.) لكن لو مكتوبة ب VHDL يبقى (vhd.) .... بعد ما بخلص synthesis وتبقى ال netlist جاهزة لازم اتاكد من ال functionality بتاعتها انها مظبوطة فعندى اتجاهين:

1- اولهم formal verification وده مفضل بالذات لو عدد ال inputs عندى كتير اوى ... فده له tools خاصة بيه وخطوات هنتكلم عنها فى شرحه فى ASIC flow

2- تانى حاجة هى Gate level simulation وده تقدر تستخدمه لو عدد ال inputs قليل وهنا انا برجع ل modesim وادخله ال netlist الى طلعت مع ال behavioural view of the library لإن دلوقتى عايزين نعمل simulation وناخد فى اعتبارنا ال timing ولكن ال nestlist عبارة عن gates فقط من غير اى معلومات عن ال timing وبالتالى لازم ادخلها مع ال netlist حاجة تقولها ال info بحيث ان فى ال simulation دلوقتى هلاحظ ال delays & transition time بتاع ال signals على عكس اما عملنا simulation on modelsim فى الاول عشان اتاكد من RTL code انه شغال صح بس ساعتها مكناش لسه خدنا فى اعتبارنا اى timing فكانت كل ال signals عندى sharp transition & no delays وهو ده الفرق بين المرتين الى عملت فيهم simulation على modelsim .... ال timing info لكل حاجة فى library موجودة فى behavioural view of the library

==========

2- Liberty Timing view [std_cell.lib] — used in logic synthesis & physical synthesis

هنتكلم فيها كتير .. عند استخدام synopsys tool بندخلها فى شكل تانى وهو(db.) ده بيكون compiled version من lib. ونقدر نحصل عليه من [library compiler (lc_shell), a tool from synopsys] الاتنين فيهم نفس المعلومات بالظبط ولكن هى formats مختلفة

حسب ال tool بتعرف تقراه مين فيهم ولكن .lib ممكن تفتته عادى بس .db هتلاقيها encrypted... هما فيهم ايه اصلا؟ بيكون فيهم معلومات عن كل cell موجودة فى library:

- Cell name / Cell Area / Cell function
- Cell outputs rise/fall transition:

  ده الى بيبقى من 10% الى 90% من output والعكس حسب rise or fall

- Cell rise/fall propagation delay:

  ده الى من وقت تغير ال input لحد ما output يتغير نتيجة للتغير input وبتتقاس من 50% من قيمة ال input الى 50% من قيمة ال out

- Cell internal/leakage power

طب المعلومات دى هتكون موجودة ازاى او انا كمان اقدر افتح ال file واقراها ازاى ؟ احنا هنشوف مثال لل propagation delay ومنه نقدر بردوا نقراه الباقى بنفس الطريقة ... بشكل عام انا عندى حاجتين هياثروا على قيم power و delay و output transition time وهما input transition و output capacitance (load) لان اكيد الفترة الى هياخدها ال input عشان يتغير من القيمة القديمة للجديدة والى على اساسها هيتيغر ال output بردوا هتأثر على output transition , delay , power وكذلك ال load الى عند ال output ده بردوا وبالتالى ال .lib بيكون فيها قيم مختلفة لل input transition و output capacitance (load) وبحسب المطلوب سواء كان output transition , delay , power عند كل قيم موجودة ... خلينا نشوف المثال ده ونفهمه:

```
pin (OUT) {
max_transition : 1.0;
timing() {
related_pin : "INP1";
timing_sense : negative_unate;
cell_rise(delay_template_3x3) {
index_1 ("0.1, 0.3, 0.7"); /* Input transition */
index_2 ("0.16, 0.35, 1.43"); /* Output capacitance
*/
values ( /* 0.16 0.35 1.43 */ \
/* 0.1 */ "0.0513, 0.1537, 0.5280", \
/* 0.3 */ "0.1018, 0.2327, 0.6476", \
/* 0.7 */ "0.1334, 0.2973, 0.7252");
}
```

- أول حاجة احنا بندرس عند output node وبالتالى كاتبلنا pin OUT
- بعد كده عندنا timing () ده كده بيعبر اننا بنقيس قيم ال propagation delay I (فمثلا فى power هتلاقى مكانها internal power()) بعد كده هنبدأ نفتح القوس بقى { وجواه كل المعلومات الى عايزينها
- هنلاقى related_pin دى بحط عندها ال input الى بيتغير وانا بحسب ال delay بتاع ال output عشان يتغير هو كمان ... طب هى هتفرق انى input؟؟ اه طبعا هتفرق لان فى الحقيقة لو مثلا عندى Gate with two or more inputs ال wires الى ال inputs دى واصلة بيها على gates مش زى بعض يعنى هتلاقى مثلا واحد اطول من التانى وبالتالى ال delay بتاعه اكبر وده نتيجة لان بيحصل routing لسه فى الاخر لكل التوصيلات بحيث نقلل ال area على قد ما نقدر فممكن مثلا سلك يبقى واصل straight وواحد تانى قعد يلف شوية على ما وصل وهكذا ... وبالتالى اى حاجة هتقيسها لل output لازم تحدد بتقيسها نتيجة التغير على انى input بالظبط.

- بعد كده فيه timing_sense دى بتحدد هل ال input المذكور ده و output بيتحركوا فى نفس الاتجاه ولا عكس بعض يعنى مثلا بيبقوا 0 او 1 فى نفس الوقت ولا لما واحد يبقى 0 التانى يبقى 1 .. فدى ليها حاجتين اما تبقى **negative_unate** يعنى بيتحركوا عكس بعض او **positive_unate** يعنى بيتحركوا مع بعض ففى المثال هى negative_unate لان ده متاخد ل inverter.
- بعد كده عندى cell_rise ودى بتعبر عن حالة ال output بمعنى هى rise يبقى انا بقيس و ال output بي rise يعنى بيتحرك من 0 الى 1 ولو كانت cell_fall يبقى العكس بيتحرك من 1 الى 0 وتختلف القيم فى rise عن fall وانا بقيس اى حاجة مش بس delay لان فى rise بيكون output بيشحن من خلال pmos لكن فى fall من خلال nmos واحنا المفروض عارفين ان Rp> Rn وبما ان tprop=0.69 RC يبقى اكيد فى rise هيكون اكبر وهكذا فى اى حاجة هنقيسها هتختلف لان مش نفس ال path الى بيشحن هو الى بيفرغ وبالتالى القيم الى بعوض بيها فى المعادلات هتختلف.
- عندنا index_1 ده بيعبر عن قيم input transition مختلفة و index_2 بيعبر عن قيم output capacitance مختلفة زى ما قولنا هنحسب عندهم ال delay
- واخيرا فى ال values هتلاقى اول row هو output capacitance الى كانت فى index_2 واول column هو input transition الى كانت فى index_1 وكل قيميتن مع بعض بيعملوا value لل delay تساوى التقاطع بتاعهم فى الجدول ده زى مثلا عند value = 0.0513 كان عندى input transition = 0.1 و output cap. = 0.16 .... ولكن فيه طريقة مختلفة بتتكتب بيها ال values وهى انها تكتب لوحدها يعنى من غير ما احط cap value and input transition معاهم بس انا ابقى عارف ان كل value هى المفروض نتيجة انى cap & input transition
- لو فتحت ال lib. (file) هتلاقى فيه بقى زى المثال ده وغيره لل power و output transition ودايما هتلاقى كلمة تعبرلك ده فى rise or fall زى هنا كده cell_rise ... واى معلومات تانية هتبقى مكتوبة برضوا لكل cell فى library.
- كمان هتلاقيه حاطط قيم لل setup & hold لكل cell ولكن هنا مش مرتبط بال input لا هيبقى مرتبط بال clk زى ما احنا المفروض عارفين ان setup & hold انا بحسبهم related لل clk عشان كده هتكون related_pin هنا هى clk وبردوا دلوقتى ال values مش معتمدة على input transition & output cap ولكنها بتعتمد فى الوقت ده على data transition & clk transition ودى معروفة من معادلاتهم لما نيجى نحسبهم manually فعشان كده هيكون عندها index_1 هو data transition و عندنا index_2 هو clk transition وبنفس فكرة ال table الى فى values هتكون ال data transition هى اول row و clk transition هى اول column ... وبدل ما كان قبل كده فيه timing_sense هيكون عندى هنا timing_type بحيث اقول فيه عايزين hold or setup وكمان rising or falling فمثلا فى المثالين كان setup_rising and hold_rising

```
timing () {
related_pin : "CK";
timing_type : "setup_rising";
rise_constraint
("setuphold_template_3x3") {
index_1("0.4, 0.57, 0.84"); /* Data
transition */
index_2("0.4, 0.57, 0.84"); /* Clock
transition */
values( /* 0.4 0.57 0.84 */ \
/* 0.4 */ "0.063, 0.093, 0.112", \
/* 0.57 */ "0.526, 0.644, 0.824", \
/* 0.84 */ "0.720, 0.839, 0.930");
}
```

```
timing () {
related_pin : "CK";
timing_type : "hold_rising";
rise_constraint ("setuphold_template_3x3") {
index_1("0.4, 0.57, 0.84"); /* Data transition */
index_2("0.4, 0.57, 0.84"); /* Clock transition */
values( /* 0.4 0.57 0.84 */ \
/* 0.4 */ "-0.220, -0.339, -0.584", \
/* 0.57 */ "-0.247, -0.381, -0.729", \
/* 0.84 */ "-0.398, -0.516, -0.864");
}
```

Important notes about .lib

- فى كذا delay calculation model تقدر تستخدمهم synopsys tool ولكن الأكثر استخداماً هو (NLDM) اختصارا ل non linear delay model وده الى احنا شوفناه وفسرناه فى الامثلة الى فاتت انه بيجيب قيم مختلفة لل input transition & output cap ويطلع ال delay المقابل ويطلق على الشكل ده كده look up table LUT وبالتالى لو فيه قيم اتحطت فى constraints لل input transition & output cap ومش موجودة ضمن المفروضين بتتجاب ب interpolation … هل NLDM بنحسب بيه حاجة تانية غير ال delay ؟ اه بشكل عام نقدر نستخدمه فى حساب:

rise/fall transition - rise/fall propagation - rise/fall power

- القيم المحسوبة فى dynamic power بتكون عند activity factor = 0.5 يعنى بيعتبر ان ال signal بتتغير كل one clk cycle وده مش منطقى لان فى الواقع هتبقى بتتغير بمعدل اقل من كده وبالتالى ده ال worst case ففى اختيار تانى ممكن نعمله وهو file of .vcd الى كنا بنطلعه فى simulation لما بنكتب فى testbench باستخدام verilog عن طريق dumpfile function فهو ال file ده بناخده نحوله ل format تانية تفهمها ال tool وندخله بحيث ان بيكون فيه estimation اقرب للواقع عن معدل تغير كل signal.

==========

3- Physical view [std_cell.lef] & [std_cell.gds]

احنا المفروض عارفين ال ASIC flow بشكل عام خطواته ايه زى logic synthsis وبعد كده هبدأ اعمل Placement & Routing (PNR) & RC extraction & DRC & LVS … لحد ما اوصل ل layout فى format معينة اسمها gds وهى دى الى المصنع هيفهمها ويصنع على اساسها … فال physical view بقى متقسم لحاجتين على الاجزاء دى:

- Abstract of cells [std_cell.lef]:

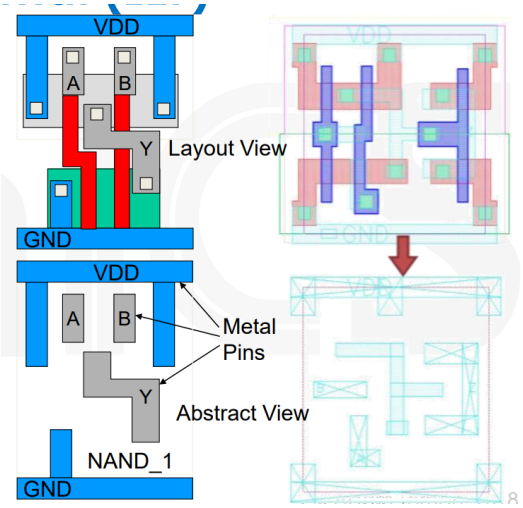
وده بدخله لل tool فى مرحلة Placement & Routing (PNR) & RC extraction وده بيدينى بس الحاجات اماكنها فين على layout يعنى كل cell شكلها ايه وال pins مكانها فين فبيجمع فيه معلومات تقدر تخليه يعمل شكل مبدئى لمكان الحاجات هتكون فين وعلى حسب كده هكمل واطلع layout.

من ال layout بتاع كل cell الى هو موجود فى GDS بتاعها اطلع ال lef بتاعها ثم ال .lef الكامل بقى بيبقى فيه ال lef بتاع كل cell زى اى library فيها معلومات عن كل ال cells الموجودة …. ففى ال lef بيكون عندى شكل مبدئى لل cell او بمعنى اصح اماكن الحاجات جواها فيعنى بيبقى عندى اماكن ال pins & VDD & GND وكده متبقى ال metal الى هيوصل بينهم وده بنحط مكانه blocks تقول لل pnr tool متعمليش routing لل wires الخارجية فى metal 1 هنا .. يعنى ايه؟ دلوقتى ال tool هتاخد شكل ال cell فى .lef كبداية تطلع منها ال layout بعد كده فانا لو سيبت اماكن ال wires الى جوا ال cell نفسها فاضية وانا اصلا عندى wires تانية برا الى هما هيوصلوا ال cells دى ببعض مع VDD & GND … ال chip يعنى من الاخر فيه connections برا ال cells نفسها لسه فكده ممكن ال cell تيجى تعمل routing لل wires الخارجية فى نفس layers والاماكن الى المفروض خاصة ب wires inside the cells فعشان كده بحجز اماكن ال internal connections inside the cell عن طريق انى احط مكانها blocks ….ال tool ملهاش دعوة بايه جوا ال block ده هو اصلا جاهز شكل توصيلته لان دى cell جاهزة … فهو بيسيب المكان وبس.

- Layout of cells [std_cell.gds]:

ده عبارة عن ال layout بتاع كل cell فمنه اقدر احصل على layout بتاع الشكل كله ولكن ال tool بتستخدم ال .lef الى شوفناه… عشان كده شكل ال cell تحت فى ال layout view هو ده GDS فكامل من كله وال layers كلها محطوطة

وبالتالى ده هيكون فيه كل المعلومات الى هنحتجها عشان ال final checks زى DRC, LVS لان من جوا عندى بالفعل حاجات متوصلة جاية من المصنع اصلا فهتقول مثلا ال min spacing or min width و اى معلومات هحتاجها بقى عامة عشان check على ال layout الى هطلعه لل design كله فى الاخر ونت check عليه باستخدام DRC Design rule check بحيث اتاكد انه فعلا بيحقق ال design rules زى المسافات بين wires والحاجات الى بيبعتها المصنع عشان نضمن نسبة خطأ اقل فى التصنيع بدل ما يحصل مثلا short circuit او حاجتين ميوصلوش لبعض اصلا وهكذا .... وكمان LVS Layout vs Schematic وهنا بتاكد انه Layout فى الاخر مطابق فعلا لل schematic المطلوبة.



فيعنى نقدر نقول انه lef بتخلى PNR تحط الحاجات فى اماكنها وتعمل routing مظبوط ثم اعمل checks على ال layout بتاعى الاخير باستخدام gds.

==========

Operating conditions:

ال operating conditions معناها PVT process voltage temp:

- بالنسبة لل temp فالاحسن ان تكون درجة الحرارة قليلة لان لما بتزيد بتزود threshold voltage الى بيزود ال delay
- باالنسبة لل voltage فالاحسن انه يكون اكبر حاجة موجودة لان كل ما supply زاد فانت هتزود التيار اللى بيشحن ال cap وبالتاالى هيقل ال delay
- بالنسبة لل process فوقت الصنيع فى حاجات كتير ممكن تاثر عليا زى doping, impurity densities, diffusion depth and oxide thickness .... كل دى حاجات فى physics ال transistor وانا الى يهمنى هى ال chip هتشتغل ولا لا فى الظروف دى .... بعبر عن ال process برقم بحيث لو كان بواحد يبقى احسن حاجة واقل من واحد يبدأ يبقى اسوء فاسوء

وبكده يبقى احسن ظروف ممكن تتحط فيها هى :lowest temp - highest voltage - process = 1 ودى بسميها fast او min على اساس اقل delay واسوء ظروف هى : highest temp - lowest voltage - process = 0 ودى بسميها slowاو max على اساس اكبر delay

اكيد ال chip لما تتصنع وترجع مش هتكون دايما فى operating conditions ثابتة وبالتالى كل library هلاقى منها كذا واحدة ليها نفسها ولكن عند operating conditions مختلفة:

1- fast fast (ff) library (min library) : means lowest delay in the technology (fast) & best process parameter (equals 1) (fast) …. gives the worst case for the hold analysis so it used during it

2- slow slow (ss) library (max library): means highest delay in the technology (slow) & best process parameter (equals 0) (slow) ... gives the worst case for the setup analysis so it used during it

3- typical typical (tt) library (normal library) : means normal delay in the technology & best process parameter (normal)

يبقى اختصارات ss ff tt اول حرف فيها بيعبر عن ال delay والتانى لل process

لما بدخل ال library بدخلها فى three conditions دول يعنى كانى بدخل 3 libraries ولكن فى الواقع هما نفس الحاجة باختلاف ال operating conditions

==========

Contact info:

Linkedin profile: linkedin.com/in/fatma-ali-57b1a6200

E-mail: fatma.ali.2028@gmail.com

All ASIC files are provided on VLSI - ASU Community:

https://drive.google.com/drive/folders/1aLwCLZj0YG9KJhln1D60_nWM7p-L3q9a?hl=ar

==========