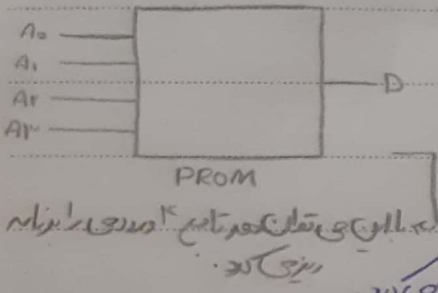


PROM ← آیات الی (برقی) حافظه ای توان logic ها را بیا به ساری کرد.
(برای ساخت یک آیات الی (برقی) حافظه ای توان logic ها را بیا به ساری کرد.)

ستون خردی در جدول ارزش رادر PROM (ضمیمه کنیم)، از این حافظه ای توان به عنوان Combinational logic استفاده کرد.
(به این معنی که بر اساس ورودی ها، خروجی ها را به صورت یکتا و یکتا به دست می آید.)



PROM حافظه ای غیر پوی است. (یک بار برنامه ریزی می شود.)

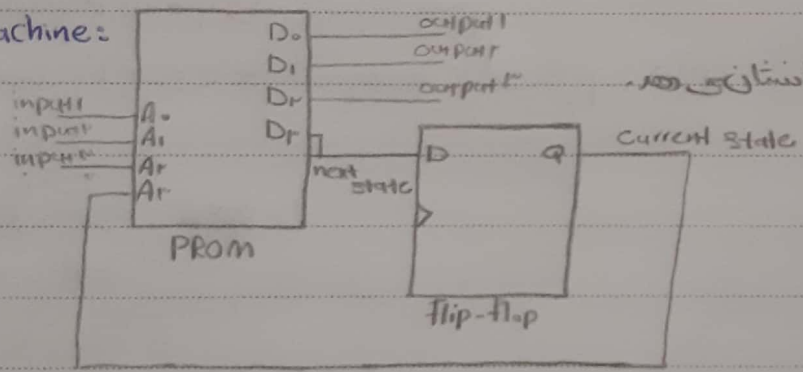
ستون جدول ارزش رادر خانه های متناظر حافظه را ضمیمه کنیم.

PROM می تواند مدارهای ترکیبی را بیا به ساری کند.

آیتم های PROM، flip-flop، و آیات الی (برقی) حافظه ای توان (ترکیبی/تریتی) بیا به ساری کرد.

عیب: سرعت آن است. چون مدار به حافظه زمان نیاز است و با افزایش تعداد ورودی، بیشتر می شود.

PROM-based state machine:



Flip-flop ها: حافظه ای توانی که می تواند حالت های مختلفی داشته باشد.

Standard logic chips: (TTL) و CMOS

با قرار دادن چند تا از chip ها کنار هم می توان مدارها ساخت.

حفظ برنامه ریزی کرد. programmable

عیب: حجم زیادی از logic باید استفاده شود تا یک سیستم کوچک بیا به ساری شود.

programmable logic Technology:

① SPLD (PLA, PLL)

Logic ها را به صورت sum of product برنامه ریزی کنیم.

② CPLD

PLA: ورودی ها را دریافت می کند و در مدارهای پیچیده و پیچیده مدارها را می تواند پیاده سازی کند (ورودی ها را می تواند پیاده سازی کند).

Programmable و یک بار برنامه ریزی می شود.

and plane: توان می تواند برای پیاده سازی مدارهای and و or استفاده کند.

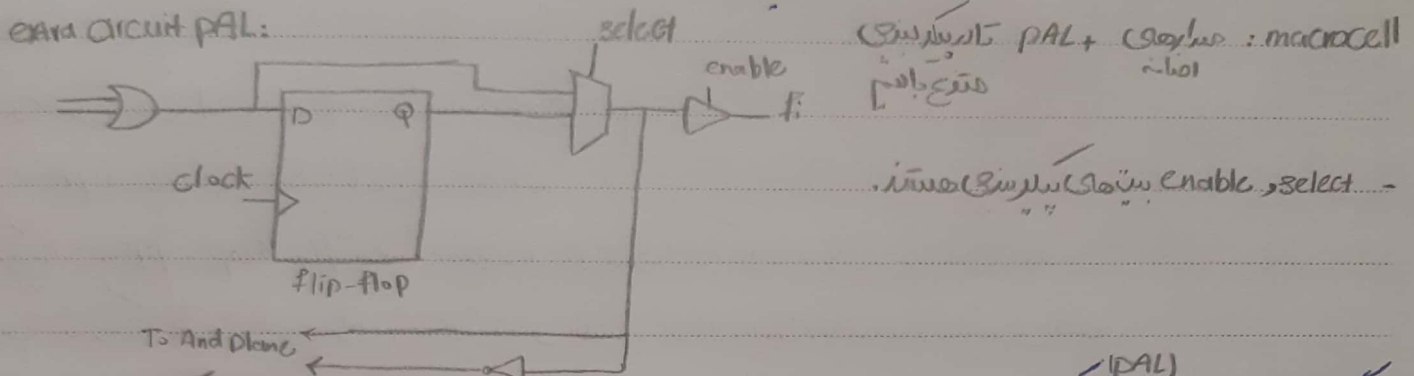
sum of product or plane می تواند پیاده سازی شود.

and plane → PLA ها می توانند برنامه ریزی شوند. (Connections) می تواند به ساری شود و می تواند به ساری شود.
(معمولاً)

programmable - یعنی راحت بنا می آید و آن چه در ذهنتان است. (به عنوان مثال برای پیاده سازی)

PAL : or plane قابل برنامه ریزی نیستند. and gate و and های قابل برنامه ریزی هستند. و این ترانه PAL ها

اندکنار PAL و PLA ها یک سری logic اضافه (Flip-flop) قرار می دهیم sequential logic را پیاده سازی کنیم
- حرف select را در این جا programmable در قاری کنیم به مدار ترکیبی باشد یا تریبی.



اندکنار زبانی از مدار بالا را می نامیم قرار می دهیم به وسیله ی نیم (PAL)
های قابل برنامه ریزی و یک سری block هم در درون قرار می دهیم CPLD
- دردی مدار از block می آید block ها از طریق inter connection wire می آید
- برای مدارهایی است که می توانیم در PAL پیاده سازی کنیم
- D flip flop را به عنوان and plane به feed back

CPLD شامل مدارهای SPLD ها.

0 → high output
1 → Input
ی ورودی

XOR یک not قابل برنامه ریزی است.

* مدار برای مدارهای PLD های مختلف : چه قابلیت پیاده سازی توانایی پیاده سازی را دارد. (مثلا مدارهای دیجیتال)
- امکان قرار دادن gate های پیاده سازی را می دهیم و به صورت ترکیبی می آید.

* قابلیت پیاده سازی PLD شامل مدار NAND در درون برای پیاده سازی است.

CPLD ها قابل پیکربندی هستند.

طرح ۱۳ :

FPGA:

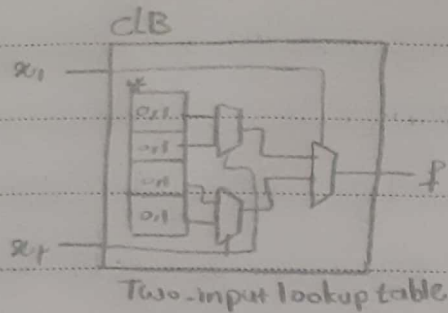
FPGAs are extensions of the idea of PROMs for logic circuit realization.

از PROM با یک مدارهای اضافه به عنوان یک block خفتر در فضا می شود. مدارهای از این logic ها در کنار interconnection wire (برای اتصال بلاک ها)

از block ها امکان اتصال تسطیل از بیرون به بیرون را می دهیم. و این ها FPGA هستند.

کتابی مفیدی قابل پیدایی (می تواند تمام دستگاه های دیجیتال را مدتی است خواند و بشود)

CLB:

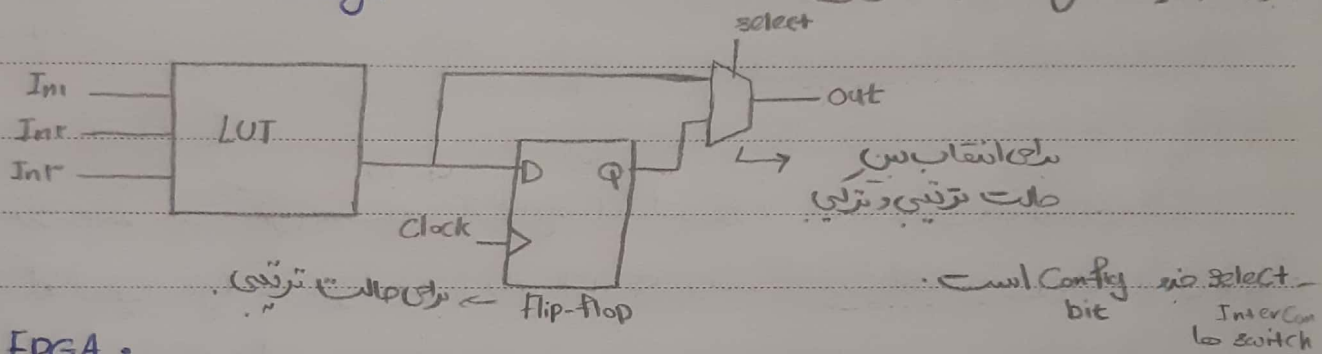


| AND | x1 | x2 | f |
|-----|----|----|---|
| | 0 | 0 | 0 |
| | 0 | 1 | 0 |
| | 1 | 0 | 0 |
| | 1 | 1 | 1 |

در LUT یک 4x4 مابین ها داریم که می توانیم با آن ها هر چیزی را بسازیم (مثلاً در درستی و 2 ورودی و 1 خروجی دارد) (معمولاً 2 ورودی و 2 خروجی در صورت لزوم)

هر کدام از قطعات ما را می توانیم با یک switch در Config وجود دارد که می تواند با یک switch از یک بسته کند

FPGA logic block extra Circuitry:



CPLD vs. FPGA:

CPLD → non-volatile → (پایدار) و بسته به ولتاژ می تواند کار کند

FPGA → Volatile → (غیر پایدار) و بسته به ولتاژ می تواند کار کند

Config mem → FPGA بیت های مقابل برنامه ریزی است و در حافظه می تواند کار کند و بسته به ولتاژ می تواند کار کند

PLP programming:

① off-board (CPLD): برنامه ریزی در یک برنامه ریزی که در یک برنامه ریزی می تواند کار کند

② on-board (FPGA): برنامه ریزی در یک برنامه ریزی که در یک برنامه ریزی می تواند کار کند

programming: Config bit → mem → Config bit

JTAG: رابطی برای ارتباط با دستگاه

serial interface technology

برای ارتباط با دستگاه می توانیم از JTAG و UART استفاده کنیم

جایگاه JTAG: برای برنامه ریزی در یک برنامه ریزی

TDO: خروجی از دستگاه

PAPCO

مکانده JTAG: برای انتقال داده‌های مختلف از طریق سخت‌افزار، TDO، device 1، TDI، device 2 و وصل می‌شوند.
 device‌های جدیدی ... و TDO از به عنوان TDOهای است.
 JTAG device ها وصل می‌شوند و Tck نیز می‌شود.

test data از طریق registerهای در Core logic منتقل می‌شود. (بیت‌های TDI در رجیسترهای در Core logic می‌باشد)
 و در صورتی تست اعلی می‌شود و در صورتی از طریق TDO منتقل می‌شود و در صورتی درست است یا خیر.
 Core logic چیزی است که می‌خواهیم تست کنیم.

ASIC (Application specific Integrated Circuit)

از برای طراحی خاص و مخصوص که قابل تغییر نیست.
 در اینجا ترانزیستور با به طور کامل می‌شود و در بخش‌های که به مقدار نیاز با این به طراحی به برای به وسیله ترانزیستور می‌شود.
 Connection بین بخش‌ها را مستقیماً از طریق می‌ماند و در بخش‌های به استاندارد را برای می‌کند. (gate array, standard cell)
 کم هزینه‌تری می‌شود. می‌تواند است که بعضی از gate ها استفاده می‌شوند. (area بیشتر می‌شود).
 در Fpga انتقال بین logic و در logic همین با در از switch بر می‌شود تا خیر یا درست و می‌تواند انتقال از طریق
 هم است و سرعت بیشتری دارد. (non-volatile) (تغییر برنامه می‌شود).
 در واقع connection ها programmable هستند.
 طبقه ۱۴:

Configurable logic: توانایی در تغییر و تطبیق با نیازها.
 Interconnect network: Config block ها را وصل می‌کند.
 peripheral ها در Fpga های مختلف متفاوت هستند (است. نوع Fpga)
 معماری Fpga: (مثل با در LUT) معادله‌های به بلاک‌های که می‌تواند به در (بزرگ) زیاد است.
 ① Fine grained: (بزرگ)
 ② Medium grained:
 ③ Coarse grained: (بزرگ) بلاک‌های بزرگ و بیشتر می‌تواند به در (بزرگ)

Node-based reconfigurable Architectures: کامپیوترهایی که به صورت عمومی قابل بازپیکربندی هستند.
 (در صورت نیاز) به بعضی از این device ها با تغییر به در قابل بازپیکربندی است.

بلوک‌های در سطح Fpga بخش شده اند؟

① symmetrical array - به صورت متقارن تقسیم شده اند

② Row-based - فقط بلوک‌های در یک ردیف به یکدیگر متصل می‌شوند

③ Sea-of-gates - چینی و زیاد است

④ hierarchical - یک سری بلوک نام کوچک و بلوک‌ها زیر بلوک‌های دارند

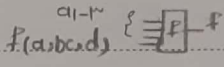
- DSP: وسیله‌ای برای محاسبات ریاضی و منطقی در یونیت پردازش دیجیتال (Digital Signal Processor) به صورت یک بلوک خاص استفاده می‌شود

که‌اند (یعنی به وسیله Config. block یا منوی تنظیمات) (processor) نوشته و آن‌ها (on-chip) (one-chip)

RAM-block: (برای پردازش هم‌زمانی) یا (برای ذخیره‌سازی) یا (برای حافظه محلی) خاص استفاده می‌شود

استفاده می‌شود (برای محاسبات خارج از chip) (off-chip) استفاده می‌شود (یعنی perform) (برای محاسبات هم‌زمانی) خارج از chip استفاده می‌شود

- logic block Fpga چند کاربرد دارد:

① به عنوان LUT (Look Up Table) -  (function block)

② RAM - به عنوان حافظه

③ shift registers - به عنوان رجیستر (serial-input)

- در Fpga های جدید برای logic های پیچیده تر از slice های عادی استفاده می‌شود

و برای طراحی‌های که توان پردازش آن‌ها را از slice های بیشتر استفاده می‌شود

- CLB ها از طریق Interconnect network ها متصل می‌شوند

2 logic cell → slice , 4 slice → CLB

مزیت اصلی این است که (HDL) Fpga یک ابزار قدرتمند برای طراحی و پیاده‌سازی Fpga ها را

انجام می‌دهد. سیستم‌های طراحی پیچیده با ابزار بیشتر می‌تواند. هرچه سیستم‌ها پیچیده‌تر شوند، ابزار بیشتر به کار می‌آید

- ساختار برای Carry chain ها و محاسبات هم‌زمانی می‌شود. (برای ساخت Adder) : Carry chain (استفاده می‌شود) (بهترین است)

(تأخیر انتقال اطلاعات) Compensate ها در زمان اجرای برنامه به کاهش تأخیر می‌دهد و عملکرد را بهبود می‌بخشد

- در xilinx یک feedback حلقه استفاده می‌شود

Interconnect networks: logic ها و logic cell ها

- به منظور استفاده از ابزار بیشتر و توانایی بیشتر طراحی

ultra fast line بین یک رجیستر و یک رجیستر دیگر (تأخیر انتقال بسیار کم)

تاخیر k_{hop} در direct چیست است. $k_{op} =$ تغییر

جلسه ۵:

Inter Connection برای اتصال بین ۲ بلاک با قابلیت SRAM و بازویته شدن switch ها اتفاق می افتد. (programmable)

Control line ها، stage های بین pipeline را کنترل می کنند.

block memory: (لایه مشترک آن ها اغلب برای حافظه به حافظه دارند)

digital clk manager: مدیریت کلاک ها

dedicated Adder and Multiplier: (Dsp) سرعت افزاینده ضرب و جمع (7/6 مگابایت در ثانیه و ضرب با استفاده از 64 بیت)

چراغی در block memory ها به صورت دسته ای قرار می گیرند چرا که نشان multiplier بهم زمین مقدار حافظه محاسبه با منطق ضربی block

clock Trees: (علامه بر قلب) جمع می کنند و نتیجه را MAC

طراحی ها مشترک هستند flip-flop های مایه ای کلاک مایه ای کنند و این کلاک باید در تمام طراحی خیر و صحه بین جلوگیری شدن به flip-flop مایه ای. (اما سبیل چرا انتقال بهم تا چند ابعاد می کنند که این تاخیر برای کلاک می شود.)

clock manager: کلاک از طریق اینچنین در اینجند و در هر کلاک رفتی بسیار در این کلاک های مایه ای

jitter removal: کلاک های مایه ای را بدون jitter توی کلاک

frequency synthesis: فرکانس های مختلف از یک فرکانس

phase shifting: فاز شیفت با تاخیر معینی

clock de-skewing: جهت اریب شدن کلاک

jitter: (کلاک ایده آل) چه مقدار کلاک در کلاک های مختلف مدلیان است

لامرعات (کلاک مایه ای) تا 4000 تا 5000 فرکانس در هر ثانیه در هر ثانیه

T است در هر ثانیه در هر ثانیه (مقدار) اختلاف بین کلاک های مایه ای

- در فرکانس های بالا clock می تواند مشکل ایجاد کند.
 - DCM: برای clock که نه فرکانس CLK را ضرب یا تقسیم به مقدار مشخصی کنیم. (معمولاً $f_{clk} = f_{ref} \times \frac{M}{N}$) چون بعضی بخش ها با فرکانس متفاوتی کار می کنند.
 - Overclock: Stage 2 که فرکانس f_{clk} را بیشتر می کند و می تواند به f_{clk} بالایی می بخشد (time-share) اتفاق بیوفتد. برای اینکه هم به f_{clk} به f_{clk} کار انجام دهد بخش مشترک باید سرعتهای فرکانس که A و B تاخیر ندهند به Overclock کنیم تاخیر زمان در logic از آن استفاده نشود.

- phase shifting: کدک را با این میزان شیب زمانی حرکت کند مثلاً شیب 90°: clock به زمانی 1/4 در هر دو clock، عملیاتی می شود.

کاربرد: عملیات ADC بدون راه اندازی 1/4 تاخیر به f_{clk} می دهد برای اینکه بتوانیم clock را به f_{clk} تغییر
 ایجاد شود. phase shifting انجام می دهیم که در هر دو clock تاخیر

- skew: به این تاخیر ناشی از مسیرهای سیگنال، سیگنال را مقداری فاصله برسد. (و با این شیب) به f_{clk} انتقال شده مشخصه های f_{clk} تاخیر نسبت به DCM می تواند d-skew انجام دهد. کدک را در هر دو f_{clk} می دهد تا تاخیر سیگنال شود و نه clock اصلی دریافت نشود.

1) برای کاهش کون ضربه های مختلف
 f_{clk}

2) حمایت کردن f_{clk} با فاج

- به f_{clk} d-skew به کاری می توان انجام داد phase-shifting

clk feed back - d-skew را انجام می دهد.

- global clock buffer: برای تقویت clock استاندارد می شود.

- clock skew: د- skew را می توان از بین برد و با این توان فاجی d-skew انجام داد. (clock feed back را از فاج f_{clk} می گیریم)

- clock good: clock فاجی هر قدر بیشتر که clock f_{clk} skew دارد

جلسه 9:

general purpose I/O:

- برای اینکه سیگنال از محیط خارج به داخل f_{clk} و برعکس، ارسال بشود از پایه های ورودی خروجی استفاده می شود.
 - که در f_{clk} به عنوان معینه پایه های جهت عنوان I/O شناخته می شوند. (در برخی های مشترک دارند قابلیت تعمیم می دهند)

- برای سرعت بالاتر و performance بهتر سیگنال و ولتاژهای می آید.

- توان خروجی f_{clk} به max باشد f_{clk} impedance و دستگاه تقابلی داشته باشند که به صورت دیجیتال می تواند کنترل شود.

- روی پایه های ورودی خروجی می توان سرعت افزایش داد (مثال: buffer) و مستقیم به LUT وصل نشده اند.

- I/O buffer: پایه ای را که به هم وصل است هم خروجی می شود f_{clk} bus می توان قرار داد (مثلاً برای ارسال در محیط data)

- روی بعضی از f_{clk} ها برای مارج های با سرعت بالا امکان انتقال f_{clk} bus های ارسال و دریافت (transceiver) گنجایش را داریم.

- از transceiver برای انتقال f_{clk} bus های transceiver استاندارد می شود.
 block

(Intellectual property)

IP Cores:

- داشتن استفاده شده در طراحی به طراحی از پیش می رود به آن مالکیت معنوی گویند.
 - طراحی با صرف انجام داده های انجام شده به مالکیت معنوی وقت اعتبار او است.

IP Core می تواند سرعت افزایش دهد و می تواند به صورت طراحی و در نرم افزار (باشد).
 (Library) و قطعه سیگنال یا (Source code) در طراحی استفاده می شود (مثال: floating point)

- ابزار هندسه که سطح را تبدیل می کند به چیزی که روی f_{clk} قرار می گیرد.

Firm IP Core:

library می شود و می شود به صورت فیزیکی در (Contig bit) ساخته می شود. (source code)

بعضی فاجی از f_{clk} برای Firm IP در هر دو قطعه می شود.

Firm IP Core: micro blaze Core, soft processor, RISK, hardware memory, 32 bit data bus, flexible, ASIC هسته ای 2, micro processor, f_{clk} استفاده کرد. f_{clk} می تواند از افزایش می پایه flexibility می دارد.

* (logic cell \rightarrow slices \rightarrow CLB)

distributed RAM = LUT

block RAM = حافظه بلوکی

پارامترهای طراحی

لگیک سل + LUT + حافظه بلوکی
LUT: حافظه بلوکی
max

speed grade (4 \rightarrow power) (کلاس سرعت)

Architectural design: طراحی معماری block

block یا پارامترهای دیگر (مربوط به قرارگیری)

مربوط به قرارگیری

behavioral simulator: شبیه‌ساز رفتار در سطح entry

design Entry:

(HDL, FSM) برای سطح بالا

RTL:

پارامترهای ورودی و خروجی در سطح RTL

synthesis: تبدیل RTL به entry

technology mapping: (تبدیل LUT به دروازه‌های FPGA)

LUT block RAM max

placement and routing: (LUT Placement) (قرارگیری و مسیریابی)

routing (inter connection) LUT \rightarrow مسیریابی بین LUTها

FPGA/CPLD Configuration using bitstreams: (رشته بیت) Config bit (انرا در بیت ساز)

Final in-system testing: (تست در سیستم) FPGA با رشته بیت

کلاس 7

micro processor FPGA design flow

در FPGA بلاک‌های دیجیتال داریم: micro processor, Function, module (طراحی معماری)

editing: تغییراتی که در برنامه می‌کنیم

debug: عیب‌یابی

micro processor برنامه باید در ROM لود شود. تا برنامه اجرا شود در FPGA ضربه‌های دیگر Config bit می‌دهیم و این‌ها را تست می‌کنیم.

1. Design Entry (FPGA) (طراحی ورودی)

schematic editors: (ویرایشگرهای شماتیک) LUTها، mac unitها، انتخاب می‌کنیم

hardware description languages (HDL): verilog, VHDL

finite state machine (FSM) editors: (ویرایشگرهای ماشین حالت محدود)

system level tool, HLS: (high level synthesis) (ابزارهای سطح سیستم) (مثل matlab, xilinx) (ویرایشگرهای سطح سیستم) bitstream تولید می‌کند

2. Functional simulation \rightarrow (شبیه‌سازی عملکرد)

behavioral simulation: (شبیه‌سازی رفتاری)

structural simulation: (شبیه‌سازی ساختاری)

در این مرحله می‌توانیم رفتار را تست کنیم

در این مرحله می‌توانیم رفتار را تست کنیم

3. logic synthesis (HDL to hardware) fpga

HDL \rightarrow Boolean Equations \rightarrow Technology mapping
 روابط بوی \rightarrow به ازای رابط بوی نه
 صنعت افزاری با ابزار استاندارد
 دستور

Config bit
 فایل بتیل به bit

behavioral Code: ممکن است دستور دینور یا نه
 Structural Code: دستور دینور

- جزئی logic synthesis netlist است.
- netlist: لیست از افعال بین سازول حالست. مثال: $(C = a \text{ AND } b)$ یا $(C = a \text{ OR } b)$
- استاندارد برای netlist: EDIF - به حرفی را EDIF و گفته.
- XNF - جزئی ابزار رنشر xilinx است نه فایل قدیش است.

Xilinx fpga design flow:

Synthesis: HDL \rightarrow gate level netlist
 (boolean func.)

* گیت های داده ساده و دستور از UNISIM که یک xilinx library است.
 در مثال basic primitive های xilinx است

translate: merge netlist های دیگر
 یکنه در صورتی ها در رفتی تیرم یکی xilinx design به صورتی دیگر

map: netlist های گیت level را (design) fit یکنه روی resource های که
 map یکنه به resource های موجود.
 مثال netlist های بتی file
 صنعت های مارج
 available fpga

place and route: fpga ها یکی LUT و BK ها یکی LUT قرار گیره و فیزیکی LUT ها فیزیکی به هم
 عقلی یکنه
 که تأخیر نداشته باشند
 * برای باید دوری بیشتر بشه که سرعت بیشتر بشه درست
 * LUT ها را کنار هم قرار ده (LUT های که بیشتر به هم مرتب شده باشند)

generate programming file:
 Config bit را تولید یکنه که روی device قرار یکنه تیرم fpga config یکنه دستور.

Electronic Design Automation:

- چگونه طراحی برای کامپیوتر توصیف کنیم و به کامپیوتر بفهمانیم تا بتواند اراده می طراحی را انجام دهد؟
- ① schematic design tool/3: شباهت طراحی را در اینتر فیس کنیم که کامپیوتر هم اقتدار من قرار یکنه و در از مارشل های استفاده می کنه و هم کامپیوتری هم (فیزی در دستر دار در راش صنعت افزاری زیادی می فراهم و مارج در دستر است).
- ② hardware description languages: به یک زبان می توصیف صنعت افزاری دستور. (VHDL, Verilog, ...)
- ③ Class و library: به یک مجموعه که شبیه سازی صنعت افزاری را انجام دهد و برای آن CAD بسیاریم (ابزار رنشر) نه آن بتیل به EDIF کنه.

طرحه ④:

- طراحی های دیجیته text برتر picture است، handle و analyze کن برای ابزارهای دستور زبان های parser ساده تر است.
- Schematic design/HDL: حالت دسلله مرادقی می تواند وجود داشته باشد (از بت مارشل هر یک مارشل به کار بسته آورده دستور).

- زبان توصیف صنعت افزاری به روشی های دیر داشته باشد؟

- ① مسطح های مختلف Abstract را چینیایی کنه چکن صنعت افزاری ممکن است در سطح مختلف توصیف دستور. (صحت قرار در سطح transition توصیف دستور انتقال transition ها یا دسل لا ترا استله از gate های منطقی یا دسل لا از P.TL و یا یکی دسل لا از System level افکار دستور.)

- ۲) معماری‌های مختلف قابل اعمال در توصیف باشد
- ۳) برای مدلی نسبت افزاینده‌ای که قابل توصیف نیست، یک توصیف کلی‌تر و منصفانه‌تر در ارائه شود. (اگرچه نه‌اشته باشد)
- ۴) Simulation بتوان انجام داد و ابزارهای ^{مستند} وجود داشته باشد که بتواند زبان را تحلیل و پردازش کند و نسبت افزار توصیف شده با زبان توصیف را استخراج کند و شبیه‌سازی کند.

۵) Test Vector: (نمونه‌های داده) را به ابزار تست و در درجه‌ها به نسبت افزار اعمال کند و مدلی استخراج کند. (کار با قابل پشتیبانی می‌شود).

۶) بتوان data structure های مختلف را از روی توصیف نسبت افزار استخراج کنیم. (مثل درخت یا گراف) بین آن‌ها ارتباط‌های نه‌داریم برای شبیه‌سازی‌ها که در ابزار تست قرار است استفاده شود. مهمی در این data structure ها نیست. (برای تحلیل‌های زمانی مثلا از سرعت استفاده می‌شود).

۷) tool chain مناسبی برای تبدیل توصیف نسبت افزار به 1 نسبت افزار وجود داشته باشد.

* ابزار تست زبان سطح بالا مثل RTL را به سطح‌های پایین‌تر مثل bit stream تبدیل می‌کند. برای fpga

- HDL ها که قابل اجرا تولید می‌کنند. (ابزار تست توصیف را به نسبت افزار ترجمه می‌کند) مولف زبان‌های C، VHDL و ...

Verbose: توصیف زیاده‌از‌زیاده
استاره‌دار