

Design Data Management voor FPGA ontwikkeling

Al snel heb je bij electronica ontwikkeling met Design Data Management te maken, zo ook bij FGPA ontwikkeling. Er wordt immers code gegenereerd die beheerd moet worden. Er moet met meerdere mensen tegelijk aan dezelfde code gewerkt worden, er moeten versies worden teruggehaald en er moet getraceerd kunnen worden wat er allemaal is veranderd. In zoverre niets nieuws, en er zijn vele data management pakketten die dit ondersteunen.

Echter FPGA ontwikkeling heeft ook zijn specifieke aandachtspunten op dit gebied. Resultaat files kunnen zo de wereld in gestuurd worden, en gaan daar een eigen leven leiden. Je hebt te maken interfaces naar de PCB, met snel veranderende bibliotheken en met snel veranderende tooling.

Dit alles heeft zijn weerslag op het data management voor FPGA ontwikkeling. Het verdient dan ook meer aandacht dan het *even* neerzetten van een DM pakket.

Trends

De trends binnen FPGA ontwikkeling zijn net als binnen andere gebieden; het moet groter, complexer, meer, steeds sneller, en het moest gisteren klaar zijn. Om aan deze markteisen te kunnen voldoen is een groter hergebruik nodig, omdat je gewoonweg niet alles van scratch af kunt maken. Tevens zullen design teams groter worden en zullen de mensen meer en meer elkaar in de weg zitten in een module. Bovendien worden deze design teams vaker multidisciplinair, en is er steeds meer sprake van specialismen.

Daarbij komt nog eens met een snel veranderende ontwikkelomgeving. De Actel's, Xilinx en Altera's komen voortdurend met nieuwe chips, die hun eigen specifieke tools en device bibliotheken vereisen.

Dit alles schreeuwt om data management.

Data Management

Bij het FPGA ontwikkelproces zal een aantal zaken beheerd moeten worden:

- Werken in teams.
 - o Met meerdere mensen aan dezelfde designs of zelfs dezelfde bestanden werken.
 - o Interfaces naar andere afdelingen.
- Reproduceerbaarheid van de resultaten.
 - o De tester heeft iets gevonden, maar dat valt niet te reproduceren met de nieuwste code, dus zal de versie teruggepakt moeten worden die de tester gebruikt heeft.
 - o Er is iets uitgeleverd en dat moet teruggehaald worden voor een wijziging.
- Borging van de bron-code.
- Traceren van wijzigingen.
- Traceren van de broncode met de resultaat files.

Op dit soort problemen zijn data management oplossingen gericht. Maar niet ieder pakket dat je kunt vinden op het Internet is geschikt.

Vragen

Vragen die opkomen als we hier verder naar kijken zijn:

- Wat moet er precies opgeslagen worden?
- Hoe heeft de designer er zo weinig mogelijk last van. I.e. hoe kan ervoor gezorgd worden dat dit process min of meer automatisch verloopt?
- Met wie en wat heeft het allemaal te maken? Hoe wordt de koppelling tot stand gebracht?
- Welke features van data management zijn nodig? Denk hierbij aan de mogelijkheid van branching en merging, taak gebaseerd werken, build management, etc.
- Wat ondersteunen de design tools?
- Heeft het pakket change management faciliteiten zoals change request/problem report en bug-tracking. Of kan het goed met dit soort systemen integreren?
- Welke integraties zijn er met ERP, PDM, en/of PLM omgevingen? Kan het pakket daar gemakkelijk op aangepast worden?
- Hoe gaat de ondersteuning geregeld worden?
- Zijn er andere data management pakketten in huis?
- Hoe gaat de invoering?

En dan is de lijst van potentiële datamanagement pakketten bijna eindeloos wat een goede keus bemoeilijkt.

Een aantal van eerder genoemde vragen zullen hieronder behandeld worden.

Welke data moet er beheerd worden?

In principe zal alles opgeslagen moeten worden wat niet weer opnieuw gegenereerd kan worden. Een netlist of de programmeerbare file zal dus over het algemeen niet opgeslagen worden, tenzij het veel tijd kost om deze weer te genereren.

Om goed een project terug te kunnen halen dient niet alleen de VHDL of Verilog opgeslagen te worden. Dit is een goed begin, maar zeker niet voldoende. Ook de instellingen van bijvoorbeeld de synthese, de place-and-route tools en de simulaties moeten mee.

Maar ook dat is nog niet voldoende. Immers, de bibliotheek die gebruikt was is inmiddels veranderd en van de place-and-route libraries en tooling is een update geweest.

Datamanagement is niet alleen een kwestie van opslaan, het heeft ook gevolgen voor de ontwikkelomgeving. Als het project als geheel teruggezet moet worden, dan zal ook de oudere libraries en de oudere tooling moeten worden teruggehaald.

Hoe heeft de designer zo min mogelijk last?

Om de designer zo min mogelijk lastig te vallen met datamanagement dient de functionaliteit geïntegreerd zijn in zijn werkomgeving. Een workbench tool is ideaal om dit in op te hangen. Dit is het front-end van de activiteiten van de designer.

Deze tool moet zorgen voor:

- De koppeling van de design tools in een workflow en draagt zorg voor de opslag van de files in het data management tool voordat de volgende stap ingegaan wordt.
- Het borgen en identificeren van de bron-code voor elke resultaat file die gemaakt wordt. Zo wordt ervoor gezorgd dat altijd de broncode kan worden teruggehaald.
- Het beheren van de bibliotheken. De tool stelt de verschillende versies beschikbaar aan de onderliggende design tooling. De workbench zorgt ook voor het vastleggen welke bibliotheken bij dit project gebruikt zijn.
- Het beheren van de tools. De verschillende versies van de tooling moeten kunnen worden gedraaid. Ook moet de workbench zorgen dat de toolversies bij het project worden opgeslagen.

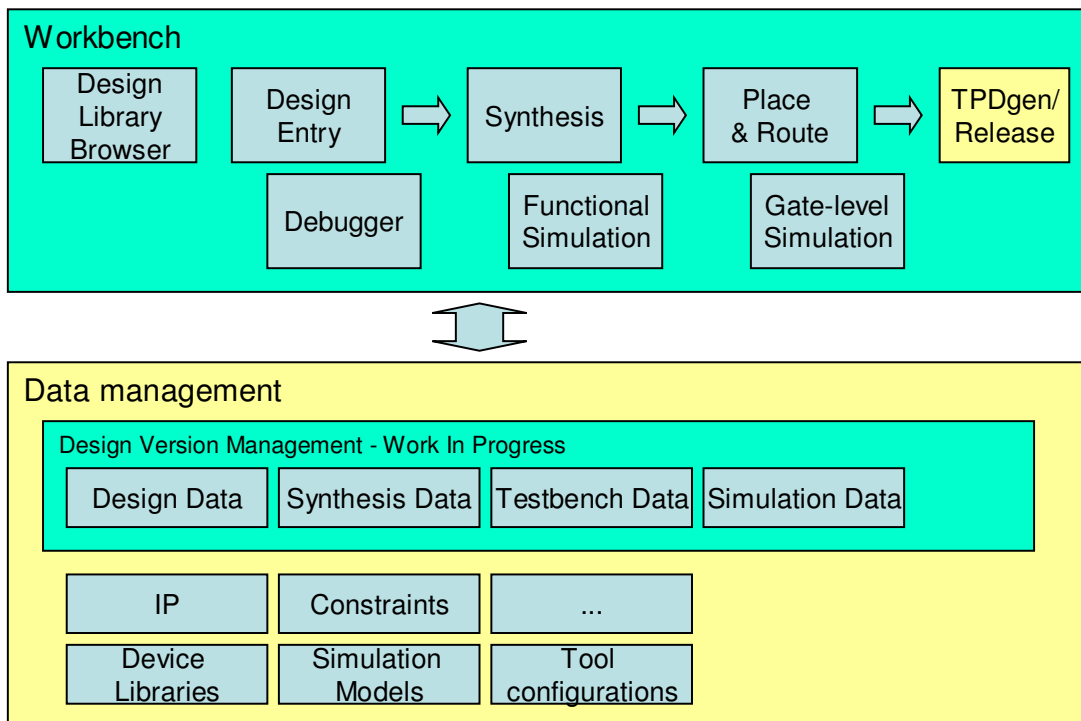


Figure 1: Workbench

Er zijn echter geen standaard oplossingen. Wel kan na een goede pakketkeuze goede resultaten bereikt worden.

Welke invloed heeft de omgeving?

Er is vrijwel altijd een omgeving waarbinnen de FPGA ontwikkeling moet passen. Bijvoorbeeld andere design disciplines zoals board ontwerp en software. Die zullen een koppeling hebben om bijvoorbeeld pinning en data-definities te synchroniseren.

Dan zijn er nog de bedrijfsprocessen waarbinnen het data management voor FPGA zich moet afspelen. Configuratie, change en release management zijn hierbij de belangrijkste. Dit kan geïmplementeerd zijn binnen PDM (Product Data Management) of PLM (Product Lifecycle Management) systemen en processen en in ERP systemen die de productie regelen. Wellicht dat deze nu nog niet aanwezig zijn, maar het is goed om naar voren te kijken, om hier alvast op voorbereid te zijn.

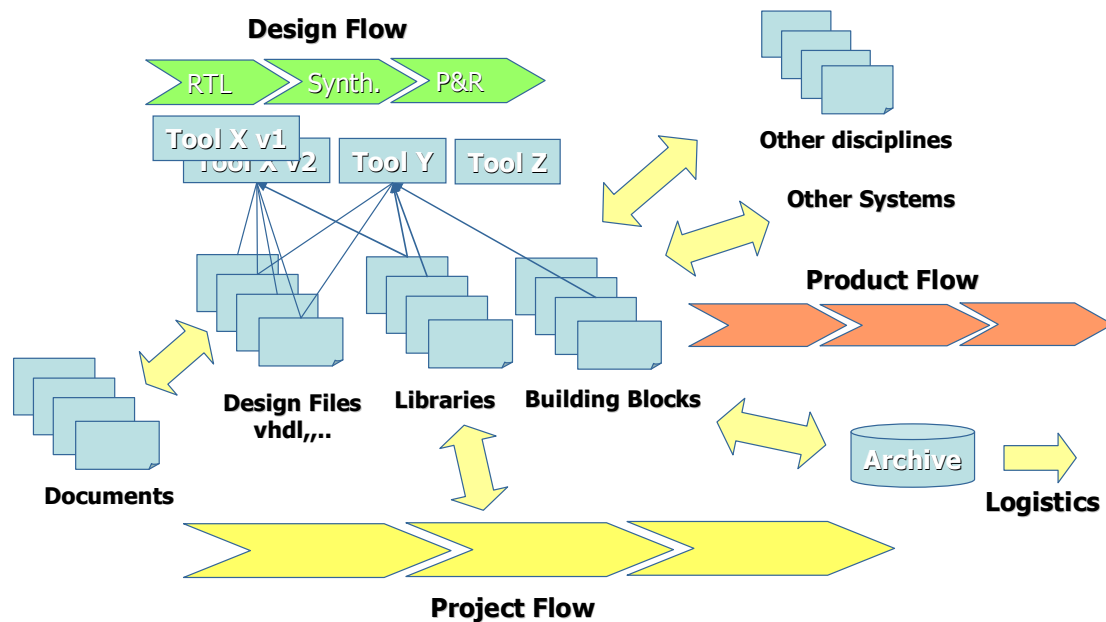


Figure 2: Design Data Management omgeving

Configuration Management

Binnen configuration management worden de relaties tussen de onderdelen beheerd. Dit betekent bijvoorbeeld dat er relaties worden gelegd tussen bepaalde versies van boards waarom het draait, de FPGA versie, de documentatie, de probleem documentatie etc. Dit creëert mogelijkheden tot het delen van informatie, het geeft inzicht in waar het allemaal gebruikt wordt (where-used) wat weer interessant is om de impact van wijzigingen te laten zien.

Change Management

Voor het beheren van wijzigingen in ontwerpen, of ze nu al eens afgeleverd zijn of niet is change management belangrijk. Dit zorgt voor overzicht en tracking wat er allemaal nog te doen staat en wat de 'known issues' zijn.

Release Management

Release management regelt het delen van informatie naar andere groepen. Wanneer je bijvoorbeeld initieel als je met een project begint is het alleen interessant voor het design team. Al snel kom je op het punt dat er meer mensen naar moeten kunnen kijken. Bijvoorbeeld voor een review of om het te testen. Later, als het project verder is, zal ook productie (of de klant) versies krijgen die allemaal weer beheerd en getraceerd moeten worden.

In de diverse stadia worden ook andere eisen gesteld aan wat er allemaal vastgelegd wordt. Initieel nog niet veel, maar als het richting productie gaat zal de volledige documentatie (de TPD - technische product data) aanwezig moeten zijn.

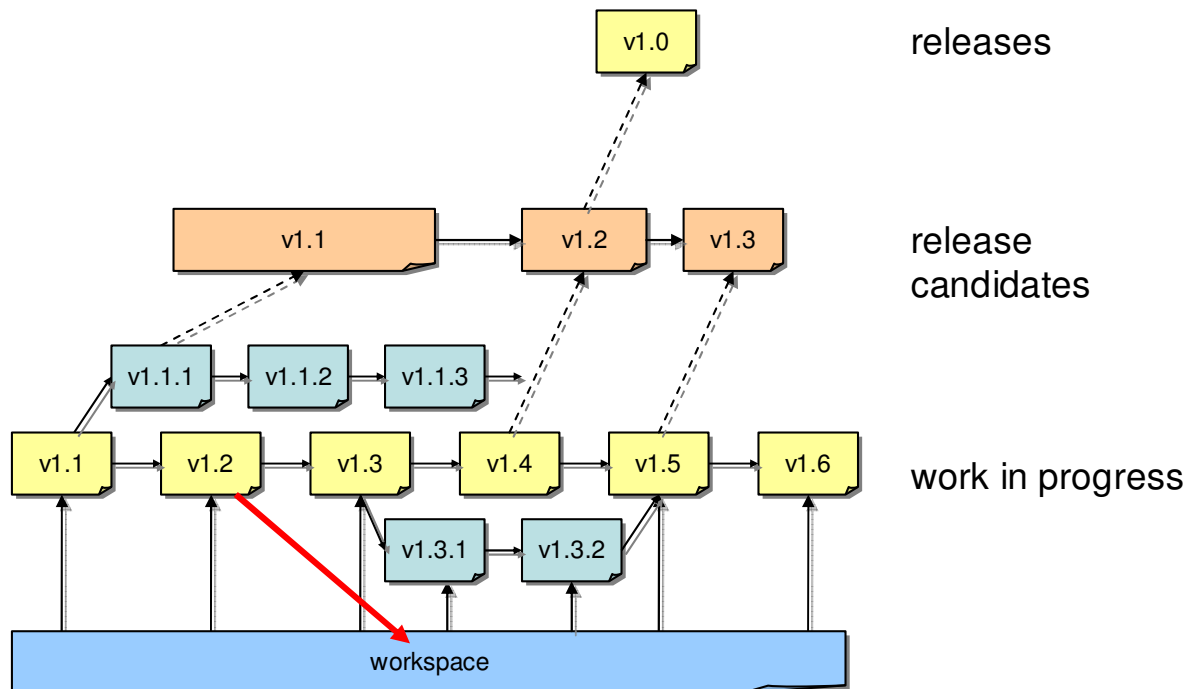


Figure 3: Release Management – niet alle versies zijn voor iedereen van belang

De data management tool zal voor al deze zaken faciliterend moeten zijn.

Waar komt er kijken bij de invoering?

De invoering van een data management methodologie gaat over het algemeen niet van de een op de andere dag. Bovendien is een 'big bang' invoering over het algemeen blokkerend en zal meestal tot onderbrekingen leiden, en dus tot uitloop van projecten.

Het is daarom verstandig om het stap voor stap aan te pakken. Elke stap dient eerst tot een succes gemaakt te worden, voordat de volgende stap genomen wordt. Hierbij dient veel aandacht besteed te worden aan het informeren en de opleiding van mensen. Als de mensen onvoldoende geïnformeerd en opgeleid zijn dan is vaak de acceptatie laag en is een succesvolle invoering bijna onmogelijk.

Conclusie

Er is meer aan de hand met data management voor FPGA projecten, dan simpel een pakket downloaden van internet en dit te installeren. De keuze van de tool waarmee het allemaal beheert gaat worden is afhankelijk van een groot aantal factoren.

Deze keus zal daarom ook weloverwogen genomen moeten worden. Een goed inzicht in de problemen die opgelost moeten worden, de omgeving en processen waarbinnen het gebruikt wordt en een goede visie voor de toekomst is daarbij noodzakelijk.

Richard van der Werf is consultant bij Dizain-Sync (voormalig Transfer Services) en heeft zich gespecialiseerd in (product) data management en product lifecycle management. Dizain-Sync biedt naast consultancy ook trainingen aan op dit gebied.