

FPGA Tabanlı İlk-Örnek Tasarımlar;

Neden Bütün ASIC'ler İlk Olarak FPGA Kullanılarak Tasarlanmalı?

Giriş;

ASIC tasarımların boyutları, karmaşıklığı ve maliyetleri artmaktadır. Aynı zamanda elektronik pazarındaki sıkı rekabet pazara çıkış süresini oldukça önemli bir hale getirmiştir. Dahası pazardaki kollar halen daralmaya devam ediyor; örneğin tipik bir ASIC tasarım 12 24 ay sürmektedir, fakat pazarın sunduğu fırsatları yakalayabilmek için ürünün 2 yada 4 haftada hazırlanıp müşteriye sunulması gerekmektedir.

Pazarın başlangıcında bir ürünlerdeki hata önemli gir gelir kaybına sebep olabilir (veya pazara girmekte geç kalınırsa yatırım veya gelirin tümü de kaybedilebilir). Bu da ASIC tasarımda ilk seferinde geri dönmeden doğru bir tasarım yapmak yönündeki baskıları büyük ölçüde arttırmıştır. Geri dönüşlerde, yonga veya sistem seviyesinde, hız, verim ve maliyetler öne çıkar.

Modern bir ASIC tasarımda, yüksek hızlı bir tasarımı birkaç Hz daha yüksek bir hızda bilgisayar ortamında eşdeğer devre yazılımsal olarak simüle edilebilir. Pratik olarak bu detaylı simülasyon tasarımın sadece küçük bir parçasının doğrulandığı anlamına gelir. Bu yüzden yüksek hızlı simülasyonları başarmak yerine, aşağıda bahsedildiği gibi dağıtılmış üç farklı kategoride, bazı donanımsal yapılar oluşturmak gereklidir.

Hızlandırma: Donanım tabanlı hızlandırıcı çözümleri tipik olarak bir dizi özel amaçlı işlemci yongaları yada FPGA'lar içerir. Hızlandırıcı yapısında dikkat edilmesi gereken en önemli şey, hızlanma moduna alınan ASIC simülasyonun komple sistem olmasıdır. Bu dağıtılmış yapı bir yonga içerisinde ASIC olarak tanımlandığında aynı sistemi doğrulamayacaktır. Bir diğer dikkat edilecek husus ise böyle bir hızlandırıcı çok pahalı olabilir ve gerçekte her birim aynı anda sadece bir veya birkaç tasarımcı tarafından gerçekleştirildiği için bu problem daha da önemli bir boyut kazanmaktadır.

Benzetim: Donanım tabanlı benzetim çözümler de bir dizi özel amaçlı işlemci yongaları veya FPGA'lar içerir. Benzetimin hızlandırıcıya göre avantajı tüm tanımlamanın sistem seviyesinde olmasıdır. Dezavantajı ise simülasyon hızının 1 MHz olmasıdır bu ise bir çok doğrulama için yeterli değildir. Ve bu birim çok pahalı olabilir ayrıca bir anda bir veya birkaç tasarımcı tarafından gerçekleştirilir.

FPGA Tabanlı İlk-Örnekler: Bir çok durumda tasarımı gerçek hızında doğrulamak gerekir. Bir video işlemcisi için örneğin video çıkış verisi akışı doğrulamada da öznel bir kaliteye sahip olmalıdır. ASIC tasarımın donanımsal ilk-örneğini oluşturmak için bir veya daha fazla FPGA kullanmak gerekir. FPGA tabanlı ilk-örnek hem yonga hem de sistem seviyesinde davranışsal olarak ASIC'e özdeştir. Ayrıca gerçek zamanlı simülasyon hızı 10MHz ile 80MHz (Xilinx ile 600MHz) mümkündür, bu tür tasarımlar pahalı değildir ayrıca bir çok tasarımcı tasarım süreci içerisinde yer alabilir bu da tasarım süresinin kısalmasını sağlar.

Synplicity Inc. Aralık 2004 den beri dünya çapında 20000 den fazla tasarımcıya ASIC doğrulama stratejilerinde tasarımlarına denetleyicilik yapmıştır. Sonuçlar gösteriyor ki günümüz ASIC tasarımlarının 1/3'ü FPGA tabanlı ilk-örnekleme yoluyla doğrulanıyor. Bu makale geleneksel FPGA tabanlı ilk-örneklemedeki bazı problemleri tanıtmaktadır. Aynı zamanda İlk-Örneklemede kullanılan tasarım araçları da tanıtılacaktır, bunlar Certify ASIC RTL İlk-Örnekleme, Synplify Proto Tek-Yonga İlk-Örnekleme, Synplify Pro advanced FPGA sentezi ve Identifiy RTL Debugger.

Tek-FPGA İlk-Örnekleme

Daha önce de bahsedildiği gibi günümüz ASIC tasarımların 1/3'ü FPGA tabanlı İlk-Örnekleme ile doğrulanmaktadır. Dahası modern FPGA lerin gelişmiş özellikleri sayesinde bu tasarımların 2/3'ü sadece FPGA tabanlı İlk-Örnekleme ile modellenebilir.

Tek-FPGA geliştirme bordlarını FPGA üreticilerinden veya malzeme teminatçılarından bulmak mümkün. Yani böyle bir borda sahip olmak ile bütün problemler çözülmüyor. Asıl problemler sentezlemede ve programın derlenmesindeki yetersizliklerde ortaya çıkmaktadır. Bu problemler aşağıda tartışılmıştır.

Geleneksel Çözümlerdeki Problemler

FPGA tabanlı birçok İlk-Örnekleme çözümünde rastlanılan en büyük problem ASIC ile FPGA tanım lamasında kullanılan HDL dili arasındaki kod uygunluğudur. Örneğin ASIC kaynak kodu tipik olarak clock-ayarlar yapılarına sahiptir ve bir FPGA gerçekleştirilmesi ile kullanılabilmesi için sayıcı parçalarının clock enableleri için çevrilmesi (uygunlaştırılması) gerekir. Benzer şekilde ASIC kodları çoğu zaman Synopsys DesignWare kütüphanesi elemanlarını içerir. Bu elemanlar hedef FPGA tarafından direkt olarak desteklenmiyorlar, bu durumda RTL eşdeğeri ile değiştirilip kullanılmalrı gerekecektir.

Bir çok İlk-Örnek uygulamasında bu çevrimlerin tasarımcı tarafından yapılması gerekir, sonuç olarak ortada iki ayrı kod dizini ortaya çıkmaktadır. Bunun anlamı ASIC kodunda bir değişiklik yapıldığı takdirde bu değişim FPGA eşdeğerinde de yapılmalıdır. İki ayrı kod dizininde uyumu kaybetmek sürpriz olmayacaktır bunun gerçekleşmesi çok doğaldır. Bu FPGA İlk-Örneğinin istenilen ASIC formundan farklı olması kabus senaryosunun gerçekleşmesine neden olacaktır.

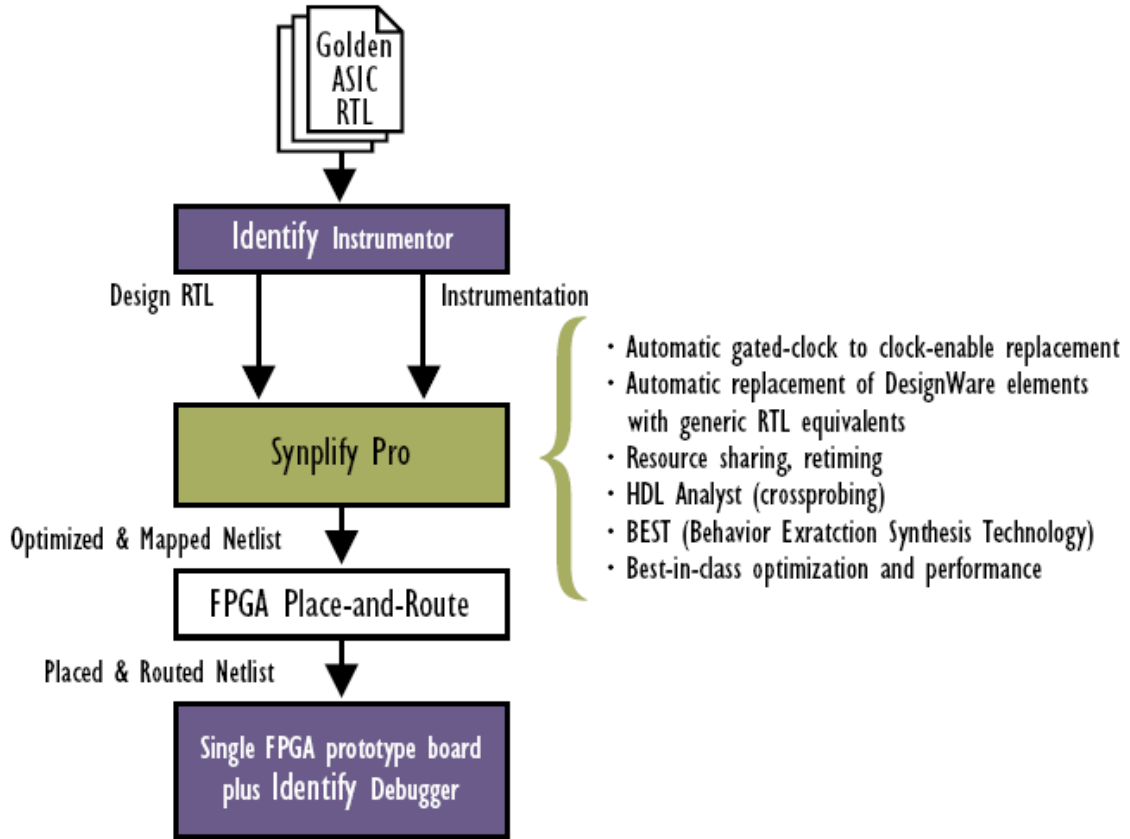
Diğer bir dikkat edilecek husus ise tasarım için uygun görünen FPGA tabanlı bir çok İlk-Örnek çözümünde vardır. Yazılım geliştiricilerin C/C++ kaynak kodu kullanıyor olduklarına dikkat edin yazılım seviyesinde derleyiciler C/C++ tabanlıdır. Benzer şekilde günümüz donanım tasarlama mühendisleri Verilog ve/veya VHDL kaynak kodu kullanıyorlar. Böylelikle maksimum verim için donanımcıların yazdıkları kod dizini içerisinde yazılım seviyesinde derleme tekniği hakkında yetkin olmaları gerekir. FPGA'lerin bir avantajı sanal derleyici mantığı yonga içerisinde kendiliğinden yapılmaktadır. Geleneksel derleyici uygulamalarında bu problem sadece ardışık olarak gelen bilgilerin grafik dalga formlarından takip edilir.

Synplify Proto' nun Avantajları

State-of-the-art çözümlerini, FPGA tabanlı İlk-Örnek tasarımlar üzerinde uygulayabilmek için Synplicity'in Synplify Pro ve Identify uygulamalarının tasarım ve doğrulama (Şekil 1) özellikleri kullanılır. Synplicity bu çözümü, tümleşik bir ürün halinde, Synplify Proto tek-yonga ASIC RTL İlk-Örnek çözümü ismiyle sunmaktadır.

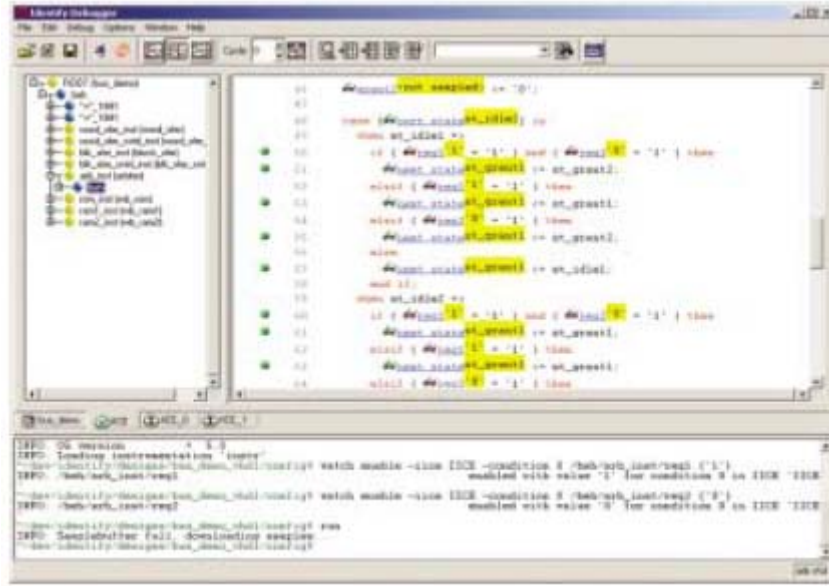
Bu akıştaki ilk eleman RTL derleyicisinin yüzeyi instrumentor' dır. Özelliği ise sezgisel ve kullanımı kolay hiyerarşik kaynak kodu sunucusudur (Şekil 2). Özel göz-camı sembolleri sayesinde herhangi bir sinyalin değerleri örneklenebilir veya sistem içerisinde tetikleme için kullanılabilir. Bu sembolün üzerinde iken fare sol tıkladığında bu sinyalin derleyici tarafından kullanılma sebebi hakkında bilgilendirici bir not çıkar farenin sağ tuşuna basıldığında ise kullanıcının bu sinyali örnekleme, tetikleme veya her ikisi olarak nasıl kullanmak istiyorsa o şekilde kullanmasına izin verir.

Benzer şekilde If-Then-Else ve Case kontrol terimleri ile sembollere kontrol atanabilir. Bu sembollerden birine tıkladığında tanımlama programı bir kırılma noktası (durdurma) oluşturmak bu yapının kullanılabileceği hakkında bilgi verir.



Şekil 1: State-of-the-Art Tek-FPGA İlk-Örnek Akış Diyagramı
Synplify Proto Çözümü

Şekil 1' de resmedildiği gibi "insrumentator" bloğunun çıkışları Synplify Pro sentez makinesini beslemekte.



Şekil 2: Kimliklendirme Arayüzü

Synplify Proto yazılımı içerisindeki sentez kısmı Identify (Kimliklendirme) den gelen enstrümantasyon RTL'yi ve tasarımı alır ve ASIC tasarımı FPGA için uygunlaştırılmış hale getirir. Örneğin ASIC yapılarıdaki herhangi bir klok yapısını FPGA için uygunlaştırır ve FPGA eşdeğerini oluşturur. Benzer şekilde DesignWare bileşeni de otomatik olarak eşdeğer RTL yapısını oluşturur.

Synplify Proto uygulamasının bir diğer önemli bileşeni de HDL Analyst' dır. Bu bileşen yüksek seviyeli yazılımı otomatik olarak yorumlayıp kapı seviyesinde şematik çizimini oluşturur. Bu grafik arayüzü FPGA tasarımlarda oldukça faydalıdır. Synplify Proto ve HDL Analyst uygulamaları HDL kaynak kodu ile blok diyagram ve kapı seviyesinde açıklamalar arasında bağlantı kurulmasını mümkün kılar. Böylece tasarımcının lojik fonksiyonlar, yerleşik sinyaller ve tasarım üzerinde daha etkili ve hızlı olması sağlanır. Bu uygulamalar sayesinde tasarımcı tasarımını ihtiyaç duyduğu bütün teknikleri kullanarak genişletebilir.

Synplify Proto yazılımı ayrıca BEST (Davranış Çıkarma Sentez Teknolojisi) bileşenine sahiptir. Bu bileşen geleneksel sentezleme araçları tarafından ihtiyaç duyulan zamanı optimize eder. Bu algoritmalar RTL'yi analiz eder sentezleme sürecini optimize eder. Bu ileri teknoloji otomatik olarak tanımlamaları yapar, yorumlar ve durum diyagramları, aritmetik işlemler, hafızalar gibi yüksek seviyeli yapıları çıkarır ve işletir.

Bir diğer gelişmiş özellik ise kaynak paylaşımı, hafıza ayarlamaları, zamanlamayı değiştirme, tekrar ve yeniden sentezleme gibi işlemleri Synplify Proto yazılımı içerisindeki MultiPoint bileşeni desteklemektedir. Bu bileşen artan tasarım pratiklerini yerine getirmekte kullanılan uzman bir metodolojidir. Sonuç olarak Synplify Proto çözümü bütün bu özellikleri FPGA gerçeklemlerinde en iyi optimizasyonu ve performansı mümkün kılmaktadır.

FPGA İlk-Örneği içerisine (peşpeşe bu durum makinesini temsil ediyor) tasarım yüklendiği zaman Identify RTL Debugger tasarımı donanım hızında derlemek için kullanılabilir. Daha önce enstürmantasyon safhasında tanımlanan bütün durumlar, durma noktaları oluşturmak için (break point) kaynak kodu içerisinde kullanılabilir. Benzer şekilde enstürmantasyon safhasında örnekler veya tetikleyiciler olarak tanımlanan sinyaller de izleyici (watchpoint bu yapılar paket data beklerler) oluşturmak için kullanılabilirler.

Bu tür bir durdurma veya bekleme noktası oluşturulduğunda eşdeğer sinyaller ve/veya durumlar HDL kod içerisinde otomatik olarak yerleştirilir ve gösterilir. Dahası devresel tamponlar örneklene sinyaller ile ortaklaştırılarak kullanıcının simülasyon boyunca bir adım ileri veya geri ilerlemesi sağlanabilir. (Identify yazılımı FPGA ile JTAG port sayesinde haberleşir bu yüzden bu işlemler için genel amaçlı I/O pinlerine ihtiyaç duyulmaz)

Böylece Synplfy Proto çözümü single-FPGA İlk-Örnek tasarımı akışında ihtiyaç duyulan her şeyi tanımlamış oldu. “golden” ASIC RTL bütün akışı kontrol etmek amacı ile kullanılır. Gerçekte Syplify Pro aracı ASIC merkezli yapıları, gated-clok gibi ve DesignWare ise FPGA eşdeğer yazılım yapılarını yüksek hassasiyetlerde oluşturarak kullanıcıya sunmaktadırlar. Bu arada Identify RTL Debugger bileşeni tasarımcıya tasarımı boyunca orijinal HDL kodunu kullanma şansını tanıyor, bu ise anlaşılabilirliği ve verimi büyük ölçüde artırıyor.

Multi-FPGA (Çoklu-FPGA) İlk-Örnekler

Büyük ASIC tasarımlar için oluşturulan FPGA İlk-Örnek donanımı birkaç FPGA'ya ihtiyaç duyabilir ki FPGA tabanlı İlk-Örneklerin 1/3' ü bu şekildedir. Bu durumda üzerinde bir çok FPGA'nın bulunduğu kartlar üreten bir üreticiden (Digilent gibi) bir geliştirme bordu satın alınabilir. Büyük tasarım şirketleri kendi borlarını tasarlamaktadırlar.

Geleneksel Yöntemlerdeki Sorunlar

Bir ASIC entegre oluşturmak için hazırlanan çoklu FPGA İlk-Örnek tasarımlarında bir çok problem vardır bunların çoğunun sebebi ana tasarımın nasıl parçalara ayrılacağından kaynaklanmaktadır. Fakat ilk olarak ASIC merkezli yapıların orijinal RTL kaynak kodlarının FPGA eşdeğer yapılarına elle çevrilmesi gerekir. Bu da iki ayrı kod dizini anlamına gelir ve senkronizasyonun kaybedilmesine sebep olabilir bu da sonuç olarak ASIC'den istenilen cevap ile FPGA İlk-Örneğinin oluşturacağı sonuçlar arasında farklılıklar olmasına sebep olabilir.

Daha sonra tasarımcılar farklı fonksiyonel grupları bir araya toplarlar burada her grup farklı bir FPGA içerisine toplanır. Bu gruplandırma geleneksel olarak kapı seviyesinde gerçekleştirilirdi (Bu FPGA in verimsiz kullanılması anlamına gelir bir çok kapı bu durumda kullanılamaz). Çok sonraları bazı akış diyagramları RTL seviyesinde gruplandırılmaya başlandı bu durumda grupların sonuçlarının hepsi geleneksel bir FPGA sentez aracı üzerinde işlendi ve böylece değişik FPGA'ların gerçek kaynak kullanımları bilindi. (RTL seviyesinde ayrılınca fonksiyonel bloklar oluşturulmuş oldu)

Bu iki senaryonun bir problemi de değişik grupların alan ve kaynak etkileridir, sonuçları almak için işletilen iterasyonların çok fazla zaman almasıdır. İlk olarak mühendisler tanımlayıcı satır bilgileri girerek her bir satır boyunca “guestimates” ler oluştururlar (“This block will probably consume ‘this’ amount of resources, while this other block may require ‘this’ amount of resources.”). bu guestimate'leri çok sayıda “group” komutu izler, sonra sentez işlemi gerçekleştirilir (RTL tabalı parçalama yapılmış ise), sonra sonuçlar incelenir, daha sonra bir diğer tanımlamanın değerlendirilebilmesi amacıyla çok sayıda “ungroup” ve “regroup” komutu işletilir.

Bu tip ilk-Örnek tasarımlarda FPGA'lerin I/O pinlerinin sayısı sınırlandırıcı bir etkidir eğer etkili bir tasarım yapılamadı ise lojik kaynakların çok küçük bir bölümü kullanılıyor olduğu halde devre üzerindeki I/O pinlerinin %100'ü kolayca tüketilebilir. Bu I/O sınırlamasının önüne geçmek için birleşik I/O gruplarını multiplexreler ile kontrol etmek gerekir ve/veya aynı blokların birden çok FPGA içerisinde oluşturulması gerekir. (Özel performans değerleri yakalayabilmek için çoğunlukla lojik kopyalamaya ihtiyaç duyulur)

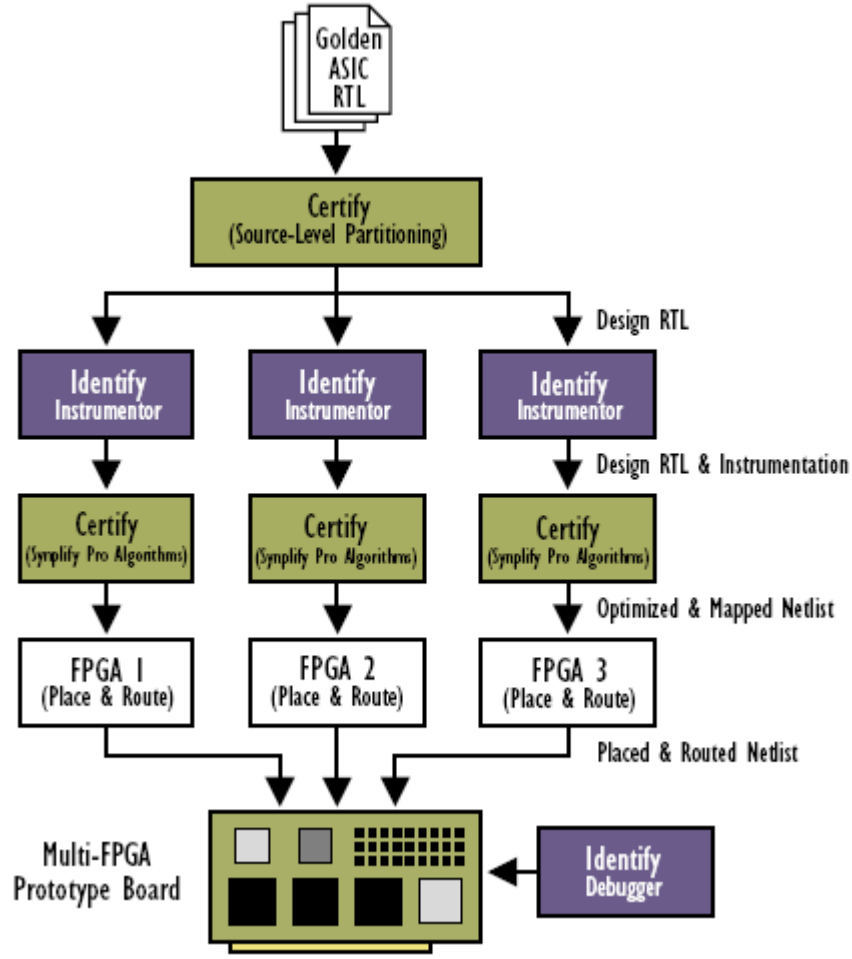
Bu tip bir İlk-Örnek tasarımda kullanılan FPGA'lerin her biri 1000 pinden fazladır, bağlantıları ifade etmek için kullanılacak bir blok çizim yaklaşımı binlerce hücre içerecektir. Böylesine büyük bir tasarımda hataların olması sürpriz olmayacaktır, bu korkunç büyük bir görevdir elle yapılması zaman ve hatalar açısından büyük kayıplara sebep olabilir.

Certify (Identify Yazılımı İle Birlikte) Ürününün Avantajları

State-of-the-art çözümü, Synplicity Certify ve Identify uygulamalarının tasarım ve doğrulama özelliklerini dikkate alarak bir çoklu-FPGA İlk-Örnek çözümünü meydana getirir (Şekil 3).

Bu akış diyagramında yer alan ilk eleman kaynak kodu parçalamakla görevli Certify çözümüdür. Eğer hazır bir çoklu-FPGA bordü kullanılıyor ise kullanıcı, bordü üreticisinden aldığı detaylı listeyi Certify programına tanıtır. Eğer özel amaçlı tasarlanmış bir bordü kullanılıyor ise Certify programı kullanılarak ilgili dosya üretilebilir.

Certify çözümü Synplicity HDL Analyst ile sıkıca entegre edilmiştir. Daha önce de bahsi geçtiği gibi bu araç tasarımı otomatik olarak yüksek seviyeli blok diyagramlar ve kapı seviyesinde şematik olarak görüntüleyebilme özelliğine sahiptir. Certify ve HDL Analyst uygulamaları HDL kaynak kodu, blok diyagram ve şematik arasında iki yönlü ilişkiyi mümkün kılmaktadır. Bu sayede kullanıcının tasarıma daha hakim olması sağlanır, tasarımcı devreyi sinyal veya lojik fonksiyon bazında çok hızlı bir şekilde inceleyebilir.



Şekil 3: *State-of-the-Art Çoklu-FPGA İlk-Örnek Akış Diyagramı*
Identify Aracı ile Certify Yazılımı
(Bu örnek sadece 3 FPGA içindir)

Bununla beraber tasarımındaki değişiklikleri gözlemek, Certify yazılımı İlk-Tasarım bordu FPGA formunun grafik olarak görüntülenmesini mümkün kılar (Şekil 4). Bu sanal bileşenlerin her biri iki adet ortaklaştırılmış “thermometer-type” görüntüye sahiptir: biri devrenin I/O kullanımını diğeri ise alan/kaynak kullanımını yansıtır.

FPGA’ler arasındaki birleştirmeler ve ortaklaştırılmış I/O kullanım bilgilerine göre Certify state-of-the-Art hızlı parçalama teknolojisi (QPT) sayesinde otomatik olarak pin görev tanımlarını yapar ve otomatik olarak sistemi oluşturur. Yada kullanıcı kendisi bu işlemi manual olarak gerçekleştirebilir – Blokları çekip FPGA yapıları içerisine sürüklemek yoluyla bu işi yapabilir – veya bu iki tekniği bir arada kullanabilir.

Certify çözümü parçalara ayırma görevi için bir çok güçlü araç barındırır. Otomatik pin tanımlama, örneğin, Certify aracı sonuçları analiz edebilir ve kullanıcıya Certify Pin Multiplexing (CPM) bileşenini kullanma fırsatı sunar. Bu bileşen sayesinde bir devredeki çoklu sinyal grupları birleştirilmiş seçme işlemlerine tabii tutularak I/O kaynaklarının ortaklaştırılarak azaltılması sağlanır.

Certify çözümlerinde temel yaklaşım şudur: sistemdeki farklı her bir FPGA tasarım hiyerarşisine eklenmiş bir katman olarak düşünülür. Bunun anlamı Certify ürünü, zamanlama birimlerinin optimizasyonunu etkili bir biçimde yapacak yetenekte özel bir yapı sunar öyleki bu alanlar farklı FPGA'lar içerisinde dahi olsa.

Alışıldığı üzere önce kısımlara ayrılmış tasarım İlk-Örnek bordu üzerindeki FPGA'ye yüklendi, Identify RTL Debugger tasarımı donanım hızında derlemek için kullanılabilir. Yeniden Identify uygulaması FPGA'in kendi JTAG portlarını kullanarak, bu iş için herhangi genel amaçlı bir I/O pini kullanmadan FPGA ile bağlantı kurar.

Sonuç

ASIC tasarımların gerçek hızda çalışan İlk-Örneklerinin yapılması gerekliliği günden güne artmaktadır. Bu iş için kullanılan bir çok etkili yöntem FPGA tabanlı İlk-Örnekleme tekniğidir, ve bu günümüz ASIC tasarımcılarının 1/3'ü bu tip bir İlk-Örnekleme yöntemi kullanarak tasarımlarını doğrulamaktadırlar. FPGA tabanlı İlk-Örneklemede tasarımların 2/3'ü tek-FPGA 1/3'ü ise çoklu FPGA kullanılarak gerçekleştirilir.

Synplify Proto yazılımı tek-FPGA İlk-Örnek tasarımının ihtiyaç duyduğu her şeyi sağlamaktadır. Benzer şekilde Certify otomatik parçalama bileşeni ve Identify RTL Debugger birlikteliği de çoklu-FPGA İlk-Örnek tasarımının ihtiyaç duyduğu her şeyi karşılamaktadır.

Her iki durumda da orijinal ASIC HDL kaynak kodu korunur ve “golden” olarak takdim edilir. Certify ve Synplify Proto uygulamalarında ASIC merkezli yapılar, sentez bileşenleri (engine) sayesinde otomatik olarak FPGA eşdeğer yapılarına dönüştürülür. Her iki durumda da Identify RTL Debugger tasarım mühendisinin orijinal HDL kodunu kullanarak tasarımını analiz etmesine ve derlemesine müsaade eder, bu sayede anlaşılabilirlik ve verimlilik artar. Sonuç olarak Certify ve Synplify Proto ürünlerindeki sentez bileşenleri tarafından görev tanımları yapılan state-of-the-Art sentez ve optimizasyon algoritmaları, FPGA İlk-Örnek gerçeklemede optimizasyon ve performans açısından en iyi çözümleri sunmaktadırlar.