Ardışıl Devrelerin Yazılım ile Model Tabanlı Sınanması

Onur Kılınççeker

University of Paderborn, Paderborn, Germany
okilinc@mail.upb.de

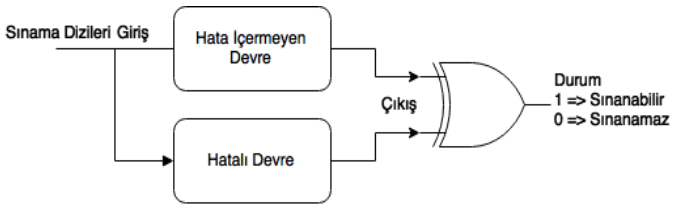
Özet. Bu çalışmanın amacı, ardışıl devrelerin (sequential circuits) ölçeklenebilir düzenli ifadeler (regular expression) ile modellenmesi ve bu ölçeklenebilir model aracılığı ile sınanmasıdır (testing). Sınama işlemi için model tabanlı sınama dizileri (test sequence) kullanılacaktır. Devrenin modellenmesi için hedeflenen hatalar aracılığı ile durum uzayı (state space) sınırlandırılmaktadır. Böylece elde edilen sınırlandırılmış uzayda tekrardan kısmi olarak modellenen devre hem sınama hemde analiz amacıyla kullanılabilmektedir. Devrenin hedeflenen hatalara karşı sınanması esnasında sınama desenlerinin elde edilmesi maliyeti ve kalitesi gözönünde bulundurulması gerekmektedir. Sınama maliyetini azaltırken sınama desenlerinin kalitesini yani hataları kapsama oranının arttırılması soyut modellerin sınırlandırılmış durum uzayında kullanımı ile mümkün olmaktadır. Devrenin sınırlandırılmış durum uzayında düzenli ifadelerle modellenmesi ve bahsi geçen sınama dizileri elde etmek için kullanımı ümit verici sonuçlar vermektedir.

1 Giriş

Model tabanlı yaklaşımların sınama için kullanımında karşılaşılan en ciddi problemlerden biri durum uzayı patlaması (state space explosion) veya diğer bir deyişle durum patlamasıdır (state explosion). Günümüz ardışıl devrelerinin sınanmasında bu problem halen güncelliğini korumaktadır. Bu problem özellikle kapı seviyesinde verilen sıralı devrelerin durum geçiş çizgelerinin(state transition graph) veya sonlu durum makinalarının(finite state machine) elde edilmesi aşamasında ortaya çıkmaktadır ve bu modellerin saklanması küçük devreler haricinde büyük miktarda bellek kullanımını gerektirmektedir [1]. Literatürde bu problemi aşabilmek için çeşitli yöntemler ileri sürülmektedir. Bunlardan ilki modelin kısmi olarak saklanması iken [2] bir diğeri durumlar arası ulaşılabilirlik verisinin modeli kapsayacak şekilde küp kümeleri ile saklanmasıdır [3]. Mevcut hataların sınana-bilir olması durumunda bahsedilen modellerin kullanımı elde edilen sınama dizilerinin en kısa olmasını garanti etmektedir [1]. Ayrıca elde edilen modeller denk hataların tespiti, devrenin doğrulanması (verification) ve fazlalık tanımlama (redundancy identification) ve kaldırma (removal) gibi çeşitli uygulama alanları bulmaktadır.

Özetle, mevcut çalışmada kısmi olarak oluşturulan sonlu durum makinaları ve ardından elde edilen ölçeklenebilir düzenli ifadeleri ilerleyen aşamalarda devrenin sınanması veya sınama devresinin otomatik olarak elde edilmesi için kullanılmaktadır. Şekil 1'de elde edilen sınama dizilerinin devrenin sınanması için uygulaması görülmektedir. Burada ileri sürülen yöntem mevcut sıralı devrenin ön

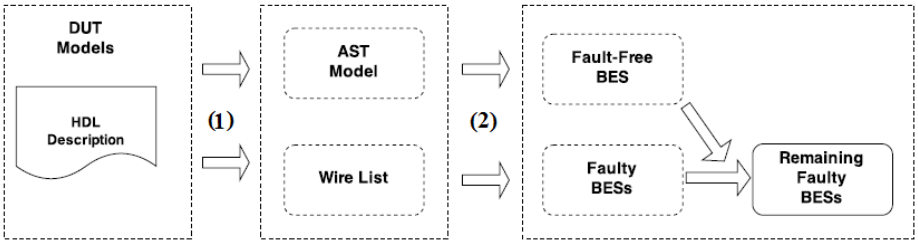
sentezli halinde donanım tanımlama dili (hardware description language) olan Verilog kodu ile verilen halinin soyut sözdizim ağaçlarına (abstract syntax tree) gövdelenmesi ile başlamaktadır. Gövdelenen kod devrenin aynı zamanda boole fonksiyonları ile gösterimine olanak sağlamaktadır ve ayrıca hatalı devrelerin yani mutantların elde edilmesi için kullanılmaktadır. Sınama desenlerinin elde edilmesine değin devam eden tüm süreçler yazılım ile gerçekleştirilmektedir. Bir sonraki bölümde yöntem detaylıca anlatılmakdır.



Şekil 1. Devrenin Sınanması

2 Yöntem

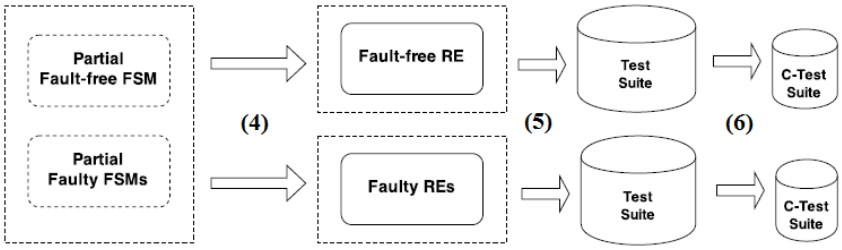
Mevcut çalışmada durum patlaması problemine çözem olarak hata tabanlı buluşsal bir yöntem ileri sürülmektedir. Bunun için hata içermeyen (fault-free) devre ile hata içeren (faulty) devre paralel bir yaklaşımla sonlu durum makinaları elde edilmektedir. Şekil 2'de görüldüğü gibi yöntemin ilk aşaması devre tanımlama dili olarak verilen sınama altındaki tasarımdan (design under test) soyut sözdizim ağacı ve kablo listesinin (wire list) elde edilmesidir. Kablo listesi soyut sözdizim ağaçları üzerindeki hata enjekte edilecek yerleri belirlemektedir. Böylece hata enjekte edilerek hata içeren mutant boole fonksiyonlar elde edilmektedir. Şekil 2'de görüldüğü gibi bu mutantların birbirine veya hata içermeyen boole fonksiyonuna denk olanları elenmektir.Girişleri paralel olarak beslenen bu hata içermeyen ve içeren mutant devrelerin çıkış değerleri gözlenmektedir. Ne zamanki herhangi bir veya birden fazla hatalı devre çıktıları beklenen doğru değerlerle farklılık gösterirse hatalı devre veya devreler listeden çıkarılmakta ardından farklılık gösterdiği durumlar işaretlenmektedir.



Şekil 2. Aşama 1-3

Aynı durumlarda işaretlenen birden fazla hata denk hatalar olarak tanımlanmaktadır. Aynı işlem listede hatalı devre kalmayana kadar işletilmektedir.

Bazı ardışıl devrelerin karmaşıklık doğası gereği belli zaman aralığında çıkışlarında farklılık tespit edilememektedir. Bu tür durumlar için iki temel yaklaşım ileri sürülmektedir. Bunlardan ilkinde listede arta kalan hatalar artık hatalar olarak işaretlenmektedir. İkincisinde ise en son tespit edilen durum başlangıç durumu olarak işaretlenmekte ve aynı işlem bu artık hatalar için tekrarlanmaktadır. İkinci durum yaklaşımının sonunda eğer halen artık hatalar kalırsa aynı işlem tekrarlanmakta ve en son durum başlangıç durumu olarak işaretlenmektedir. Bu yaklaşım durum patlaması problemine bir çözüm olarak düşünülmekte ancak zaman karmaşıklığı açısından halen karmaşık ardışıl devreler için dezavantaj taşımaktadır.



Şekil 3. Aşama 4-6

Şekil 3'te görüldüğü gibi kısmi olarak elde edilen hata içeren ve içermeyen sonlu durum makinaları düzenli ifadelere dönüştürülmektedir. Dönüşüm esnasında da makina durumlar arasındaki bir takım fazlalık geçişler (transition) elenmekte ve böylece model saflaştırılmaktadır (model refinement). Son aşamada elde edilen düzenli ifadelerden sınama takımları (test suite) elde edilmekte ve ardından bu takımlar sıkıştırma (compaction) işleminden geçirilmektedir. Burada iki adet sınama takımının elde edilmesi son aşamada bu takımları karşılaştırmaya olanak sağlamaktadır. Ayrıca elde edilen kısmi durum makinalarından sınama dizileri elde etmek mümkün iken burada daha soyut (abstract) bir modele dönüşüm (model transformation) yapılmasının ana sebebi sınama dizilerini elde etmenin zaman karmaşıklığını azaltmaktır. Ayrıca son aşamada elde edilen sınırlandırılmış düzenli ifadeler devrenin analizinede olanak sağlamaktadır. Bahsedildiği üzere bu çalışmada iki ana amaç verilen devrenin sınırlandırılmış düzenli ifadelerle gösterilmesi ve bu ifadeler yardımıyla sınama dizileri elde edilmesidir.

Kaynaklar

1. Wu, Qingwei, and Michael S. Hsiao."Efficient sequential atpg based on partitioned finite-state-machine traversal," *Test Conference, 2003. Proceedings. ITC 2003. International* , vol.1, no., pp.281,289, Sept. 30-Oct. 2, 2003.
2. Ma, Hi-Keung Tony, Srinivas Devadas, and Alberto Sangiovanni-Vincentelli. "Test generation for sequential circuits." *Computer-Aided Design of Integrated Circuits and Systems, IEEE Transactions on* 7.10 (1988): 1081-1093.

3. Ghosh, Abhijit, and Srinivas Devadas. "Test generation and verification for highly sequential circuits." *Computer-Aided Design of Integrated Circuits and Systems, IEEE Transactions on* 10.5 (1991): 652-667.
4. Cho, Hyunwoo, et al. "ATPG aspects of FSM verification." Computer-Aided Design, 1990. ICCAD-90. Digest of Technical Papers., 1990 IEEE International Conference on. IEEE, 1990.
5. http://www.scaledagileframework.com.
6. https://www.atlassian.com/software/jira.