Laboratuvar Çalışması 1

Temel Lojik Devreler, Boole Cebri ve Donanım Tanımlama Dillerine Giriş

A. Amaçlar

1. Boole cebri denklemleri ile doğruluk tablosu ve lojik devre diyagramları ile arasındaki ilişkiyi gözlemlemek ve dönüşümleri gerçekleştirmek,
2. Basit lojik kapıları kullanarak devre tasarlamak ve donanım tanımlama dili (DTD) simülasyonuyla çalışmasını doğrulamak.
3. Boole cebri kullanarak lojik sadeleştirme gerçekleştirmek ve sadeleştirilmiş lojik devrenin çalışmasını DTD fonksiyonel simülasyonla doğrulamak,
4. Zamanlama diyagramı kullanmayı ve yorumlamayı öğrenmek,

B. Bilgilendirme

1. Lab'a başlamadan önce ders kadrosu tarafından verilen ModelSim ve Quartus Prime kullanımı adlı eğitim dökümanını takip edin, ve orada yapılan aşamaları uygulayabildiğinizden emin olun.
2. Dosya adlarini her zaman küçük harflerle **labX_gY_pZ**.uzantı olarak kaydedin. **X** yerine lab numarasını, **Y** yerine grup numaranızı, **Z** yerine problem numarasını ekleyin. Örneğin lab3, SystemVerilog, Problem 1 ve grup numaranız 18 ise, **lab3_g18_p1.sv** olarak kaydedin. Devre adını da **lab3_g18_p1** olarak belirleyin. Test tezgah (Testbench) isimlerinin başına **tb_** önekini koyun, ve aksi belirtilmedikçe girişler arası bekleme süreleri için 10ns kullanın.
4. Rapor hazırlarken bu dokümanın sonundaki EK bölümünde olduğu gibi bir metin kutusu içerisine mono fontlarla (Courier New, Roboto Mono, Consolas gibi) kodlarınızı ekleyin. Kodlarınızı ayrı bir dosya olarak göndermeyin. Resim olarak eklemeyin, ve formatlamanıza özen gösterin.
5. Ek bölümünde verildiği gibi bir şablonu her kodunuzun başına ekleyin.

C. Teslim Edilecekler

1. Laboratuvar raporu **tek bir PDF dosyası** halinde olacak ve **farklı bir formatta teslim ettiğiniz raporlar geçersiz sayılacaktır.** İçerik olarak Lab0 daki örnek rapora bağlı kalınız. Tüm devre şemalarını ve simülasyon sonuçlarını gösteren dalga şekillerini ve zamanlama diyagramlarını eklemeyi unutmayınız.

D. Problemler

Problem 1 - Boole cebri kullanarak lojik devre tasarımı

Denklem 1: **Y = G’TU’E + GTUE + GT’U’E + GTU’E + G’TUE + G’T’UE + GTUE’**

A. Denklem 1'de verilen Boole denkleminin doğruluk tablosunu çıkararak, devrenin bütün olası giriş kombinasyonlarına hangi çıkışları verdiğini hesaplayınız, temel lojik kapılar (And Or Not) cinsinden devre şemasını çiziniz ve bunları raporlayınız.

B. Denklemin devre şemasına bakarak yapısal (structural, gate-level) tarzda donanım tanımlama dili (DTD) modelini ve ilgili fonksiyonel test tezgahı kodunu girişlere bütün olası kombinasyonları her 10ns lik adımlarda uygulayacak ve çıkış sinyalini gözlemleyecek şekilde SystemVerilog DTD ile yazarak simüle ediniz. Elde ettiğiniz zamanlama diyagramını kodlarınızla birlikte raporlayınız.

C. Simülasyon sonuçları ile, teorik hesaplarınızı karşılaştırınız ve raporlayınız.

D. Tasarladığınız Devrenin DTD modelini, Intel Quartus Prime yazılımı yardımıyla sentezleyiniz. Sentezleme sonucunda ne kadar yer kapladığı hakkında bilgi veriniz, sentezlenen RTL ve eşleştirme sonrası devre şemalarını ekleyerek raporlayınız.

Problem 2 - Boole Cebri Teoremlerini Kullanarak Lojik Devre Sadeleştirme

A. Denklem 1'de verilen Boole denklemini, Boole cebri teoremlerini kullanarak en sade şekline getiriniz ve raporlayınız. Kullandığınız teoremleri belirtiniz.

B. Sadeleştirilmiş denklemin doğruluk tablosunu çıkarınız ve Problem 1 dekiyle karşılaştırınız.

C. Sadeleştirilmiş denklemin devre şemasını çizerek yapısal (structural) tarzda yazılmış SystemVerilog DTD modelini ve ilgili fonksiyonel test tezgahı kodunu girişlere bütün olası kombinasyonları her 10 ns de bir uygulayacak ve çıkış sinyalini gözlemleyecek şekilde yazarak simüle ediniz. Elde ettiğiniz zamanlama diyagramını kodlarınızla birlikte raporlayınız.

D. Simülasyon sonuçları ile, teorik hesaplamarınızı karşılaştırınız.

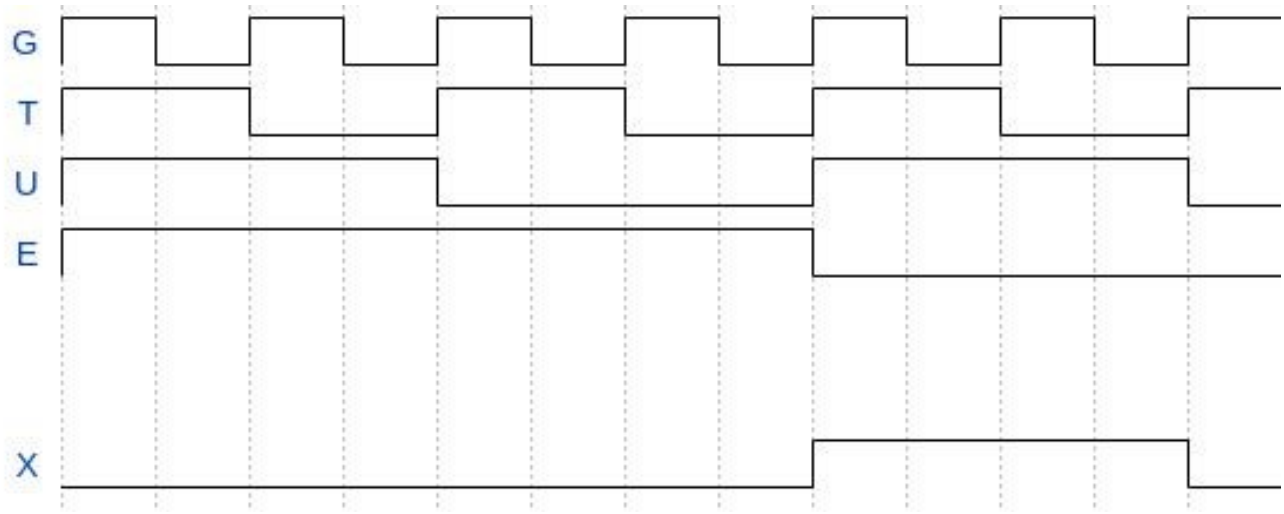
E. Devrenin, sentezleme sonucunda ne kadar yer kapladığı hakkında bilgi veriniz, sentezlenen RTL ve eşleştirme sonrası devre şemalarını ekleyiniz. Problem 1'deki devre sentezi ile karşılaştırınız ve sonuçları yorumlayınız.

Problem 3 – Zamanlama diyagramı ve dalga şekli yardımıyla lojik devre tasarımı

A. Şekil 1'de verilen zamanlama diyagramındaki dalga şekline sahip iki farklı devre için doğruluk tablosu oluşturunuz. **GTUE** etiketli devrelerin 4 girişi, **X** ve **Y** etiketleri ise devrelerin çıkışlarını temsil etmektedir. Doğruluk tablosunu kullanarak Boole cebri denklemlerini elde ediniz. (PoS veya SoP formunu kullanabilirsiniz). Boole cebri teoremlerini kullanarak en sade şekline getirerek raporlayınız. Kullandığınız teoremleri belirtiniz.

B. Sadeleştirilmiş hali ile devreleri çiziniz (Her çıkış için bir devre) ve fonksiyonel simülasyonlarını yapınız ve Şekil 1'de verilen dalga şekli ile karşılaştırınız.

C. Devrenin, sentezleme sonucunda ne kadar yer kapladığı hakkında bilgi veriniz, sentezlenen RTL ve eşleştirme sonrası devre şemalarını ekleyiniz.



Şekil 1

E. Ekler

1. Örnek bir SystemVerilog DTD tasarımı ve test tezgahı

```
/* lab1_g0_p10.sv
*
* Hazırlayanlar:
* Ihsan Cicek
*
* Notlar:
* ELM235 2023 Bahar Lab1 - Problem 10
* Y = NOT A and B denkleminin gerçeklemesi
*
*/
module lab1_g0_p10 (
input logic a, b,
output logic y
);
      assign y = ~a & b;
endmodule


/* tb_lab1_g0_p10.sv
*
* Hazırlayanlar:
* Ihsan Cicek
*
* Notlar:
* ELM235 2023 Bahar Lab1 - Problem 10 Testbench
* Y = NOT A and B denkleminin simulasyonu
* Bütün olası girişlere göre çıkış gözlemlenir.
*
*/
// Zaman birimi ve simulasyon cozunurlugu
`timescale 1ns/1ps
module tb_lab1_g0_p10 ();
// Test tezgahlarinda port bulunmaz

logic a_tb, b_tb; // test tezgahi giris sinyal tanimlari
logic y_tb; // test tezgahi cikis sinyal tanimlari

// Test edilecek modulun yaratimi ve port baglantilarinin yapilmasi
// mut = module under test,
lab1_g0_p10 mut0(.a(a_tb), .b(b_tb), .y(y_tb));

// Bu kisimda sinyaller test edilen devreye sirali olarak uygulanir.
// Sonuclar test edilen devre cikislarinda gozlenebilir.
initial begin
      a_tb = 0; b_tb = 0;
      #10; // a = 0, b = 0 yap, 10 ns bekle

      b_tb = 1;
      #10; // a = 0, b = 1 yap, 10 ns bekle
      a_tb = 1;
      #10; // a = 1 yap, b = 1, 10 ns bekle

      b_tb = 0;
      #10; // a = 1, b = 0 yap, 10 ns bekle

      #20; // simulasyondan sonra 20 ns bekle
      $finish; // sımulasyonu sonlandır
end
endmodule
```