**FB-CPU SystemVerilog Testbench**

**metin, saat içeren bir resim

Açıklama otomatik olarak oluşturuldu**

**Serhat Erdoğan, Hüseyin Berk Işıldak**

**Fenerbahçe Üniversitesi**

**Bilgisayar Mühendisliği**

**İstanbul, Türkiye**

**e-mail: {serhat.erdogan,huseyin.isildak** [**}@fbu.edu.tr**](mailto:%7d@fbu.edu.tr)**,**

*Özetçe*— Bu proje kapsamında dijital tasarım dersinde tamamlanan FB-CPU işlemcisinin SystemVerilog dili ile otonom kontrolünü yapan bir doğrulama ortamı geliştirilecektir.

Anahtar Kelimeler — FPGA, CPU, SystemVerilog, Verilog.

*Abstract*— Within the scope of this project, the SystemVerilog of the FB-CPU system, which was completed in digital education, will be developed in a way that will make it an autonomous controller.

Keywords — FPGA, CPU, SystemVerilog, Verilog.

# Giriş

tb\_fbcpu.sv dosyasında iki adet sınıf tanımı yapılmıştır. Bunlar;

• dosyaSinifi

• testSinifi

Bu sınıflardan objeler türetilerek initial begin bloğu içerisinde sınıfın çeşitli fonksiyon ve değişkenleri kullanılarak FBCPU test edilmektedir.

Proje kapsamında dosyaSinifi ve testSinifi isimli sınıfların boş bırakılan sınıf içerikleri doldurulacaktır. Aşağıda dosyaSinifi ve testSinifi isimli sınıfların içerdiği değişken ve fonksiyonlar verilmektedir. testSinifi isimli sınıf dosyaSinifi sınıfından kalıtım yapılmıştır.

Proje kapsamında geliştirilen testbench hatalı bir FBCPU kodu ile denenerek testbench görevinin doğru yerine getirip getirmediği kontrol edilecektir.

# Sistem Mimarisi

Aşağıda dosyaSinifi isimli sınıfın fonksiyonları açıklanmıştır.

• new():Constructor’dır. Tüm değişkenleri 0’a atamaktadır.

• dosyayiInitializeEt(string girisDosyaAdi):Kendisine verilen string argümandaki dosya adı ile dosyayı açmaya çalışır. Dosya açıldığında geriye dönen file descriptor’u fileDescriptor değişkenine atar. Dosya başarılı olarak açılırsa fileInitialized değişkeni 1 olur, diğer durumda ise 0 olur. Fonksiyon geriye başarılı iken 1, değil iken 0 döndürür.

• dosyadanOku(): fileInitialize değişkeni 1 ise, dosyadan 1 satır okuyup bunları memAddr ve data değişkenlerine yazar. Dosyanın sonuna erişildiğinde ise fileInitialize değişkenini 0 yapar. Dosyadan her başarılı okunan satır için lineCount değişkenini bir arttırır. Başarılı okunmalarda fonksiyon geriye 1, diğer durumda ise 0 döndürür.

Aşağıda testSinifi isimli sınıfın fonksiyonları açıklanmıştır.

• new():Constructor’dır. testNo değişkenini 0’a atar ve girisDosyasi, cikisDosyasi değişkenlerini new ile initialize eder.

• testNoAyarla(int girisTestNo):testNo değişkenine girisTestNo argümanını yazar

• testInitializeEt(): Sınıfın içinde bulunan testNo değişkenin değerine göre girisDosyasi.dosyayiInitializeEt fonksiyonu ile input1, input2, input3 txt dosyalarından birini açar (örn girisDosyasi.dosyayiInitializeEt("input1.txt")) ve çıkış dosyalarından output1, output2, output3 dosyalarından birini açar. Dosyaların açılmasında sorun olursa $finish; komutu ile simülasyonu durdurur

• kontrolEt(reg [7:0] memLocation, reg [21:0] expectedValue ):Kendisine argüman olarak verilen memLocation bilgisini kullanarak, BRAM’deki adres’e bakar. O adresteki içeriğin değeri ile expectedValue değerini karşılaştırır. Aynı ise simülasyon başarılı olarak çıktı verir, değil ise simülasyon hatalı olarak çıktı verir.

Dosya Sınıfı



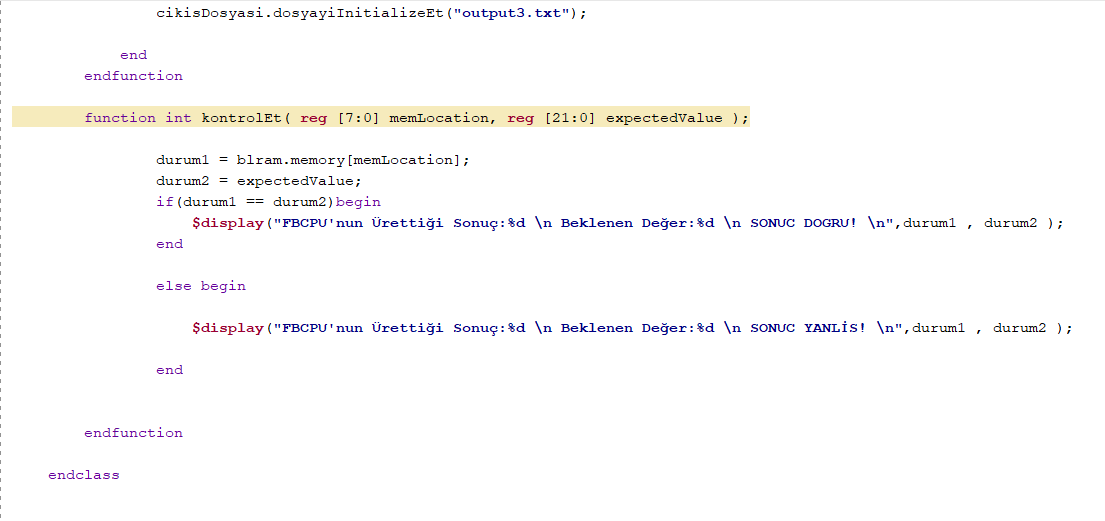
metin içeren bir resim

Açıklama otomatik olarak oluşturuldu

Test Sınıfı

metin içeren bir resim

Açıklama otomatik olarak oluşturuldu



# Kullanılan Yazılım

Bu proje kapsamında Xilinx Vivado Design Suite kullanılmıştır.

Vivado Design Suite, Xilinx tarafından donanım tanımlama dili tasarımlarının sentezi ve analizi için üretilmiş bir yazılım paketidir ve Xilinx ISE'nin yerini alır ve bir çip geliştirme ve üst düzey sentez üzerinde sistem için ek özellikler sunar.

# Sonuçlar

Proje kapsamında geliştirilen testbench hatalı bir FBCPU kodu ile denenerek testbench görevinin doğru yerine getirip getirmediği kontrol edilmiştir.

Bu projemizde elde ettiğimiz sonuçlara göre düzgün çalışan CPU’dan alınan sonuçların da doğru olduğunu gözlemleyebiliyoruz.

Ancak hatalı çalışan bir CPU koyarsak testbench hatalı olduğunu bize söyleyecektir.

Bu proje ile birlikte dijital tasarım dersinde tamamlanan FB-CPU işlemcisinin SystemVerilog dili ile otonom kontrolünü yapan bir doğrulama ortamı geliştirdik ve bir testbenchin nasıl geliştirleceğini öğrenmiş olduk.

SystemVerilog diline olan bilgimiz arttı ve Vivado uygulamasını kullanmayı daha iyi öğrenmiş olduk.

##### Proje Ekibi

Serhat Erdoğan

03.06.2000 tarihinde Doğdu 2018 yılında Yalova Uğur Okulları’ndan mezun oldu. Fenerbahçe Üniversitesi’nde Bilgisayar Mühendisliği bölümünde lisans eğitimi almakta.İngilizce biliyor. Orta seviye C, c++ ve python dili ve HTML, CSS, Javascript ve django biliyor.

Hüseyin Berk Işıldak

25.04.2000 tarihinde doğdu.2018 yılında Yusuf Kemalettin Perin Fen Lisesi'nden mezun oldu. Fenerbahçe Üniversitesi Bilgisayar Mühendisliği bölümünde lisans eğitimi almakta. Orta seviye C, c++ ve python dili biliyor.

##### Referans Dosyalar

<https://github.com/serhaterdogan77/FB-CPU-SystemVerilog-Testbench.git>

https://www.youtube.com/watch?v=DUhMv-ywyqs&ab\_channel=BerkI%C8%99%C4%B1ldak

##### Kaynaklar

[1] http://www.levent.tc/courses/electronic-circuits