RISC-V Tabanlı İşlemci Tasarımı

Bedirhan İleri, Özgür Çakmak

Fenerbahçe Üniversitesi

Bilgisayar Mühendisliği

İstanbul, Türkiye

e-mail: {bedirhan.ileri, ozgur.cakmak}@fbu.edu.tr,

**Özetçe—** Bu proje kapsamında başlangıç tasarım verilen bir RISC-V işlemcisinin ALU ve instruction decoder blokları temel SystemVerilog dili özellikleri kullanılarak tasarım ve doğrulama çalışmaları yapılacaktır.

**Anahtar Kelimeler —** RISC-V, ALU, INSTRUCTION DECODER

**Abstract—** Within the scope of this project, the ALU and instruction decoder blocks of a RISC-V processor, whose initial design is given, will be designed and verified using the basic SystemVerilog language features.

**Keywords —** RISC-V, ALU, INSTRUCTION DECODER

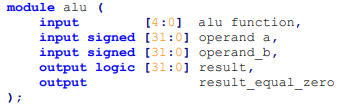
# Giriş

Risc-v tabanlı işlemcinin SystemVerilog dili ile RTL tasarımı ve tasarlanan işlemci üzerinde makine dili ile yazılan çeşitli kod parçacıkları yazılacaktır. İşlemcinin her adımı farklı modüllerle yapılacaktır. Proje sonunda basit bir işlemcideki ram, kontrol ünitesi ve saklayıcıların bir arada çalışıp, makine dilindeki kod parçacıklarını nasıl yürütebildiği gözlemlenecektir. Risc-v tabanlı bir işlemcinin ALU’sunun destekleyeceği 11 adet işlem vardır. Bu komutlar test yazılımları kullanarak test edilecektir. Bu projenin amacı temel bir işlemcinin çalışma prensibini öğretmektir.

# Sistem Mimarisi

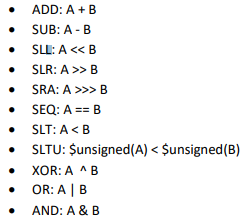
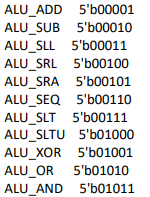
RISC-V, yerleşik indirgenmiş komut seti bilgisayar ilkelerine dayanan açık standart bir komut seti mimarisidir. Diğer ISA tasarımlarının çoğundan farklı olarak, RISC-V, kullanım için ücret gerektirmeyen açık kaynak lisansları altında sağlanır.

Aşağıda ALU ünitesinin giriş ve çıkış sinyalleri gösterilmiştir.

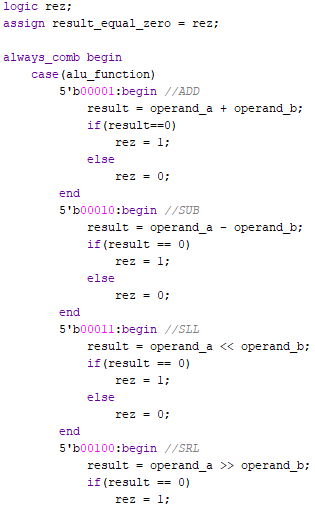


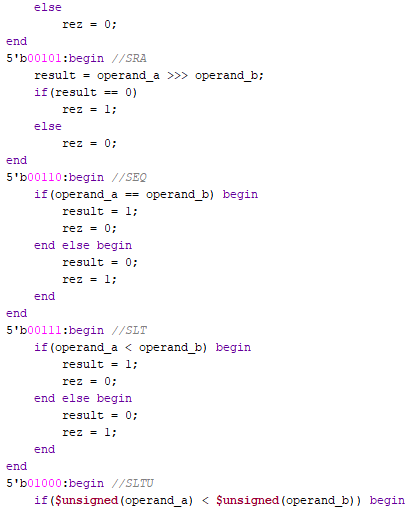
İşlemcinin ALU’sunun destekleyeceği 11 adet işlem vardır. Bu işlemlerden hangisinin yapılacağı alu\_function girişinden gelmektedir. İşlemlere göre a ve b sayıları, result isminde sonuç çıkışı ve sonuç eğer sıfır ise, ayrı bir çıkış olarak sonucun sıfır olması durumunda 1 olan bir çıktı vardır.

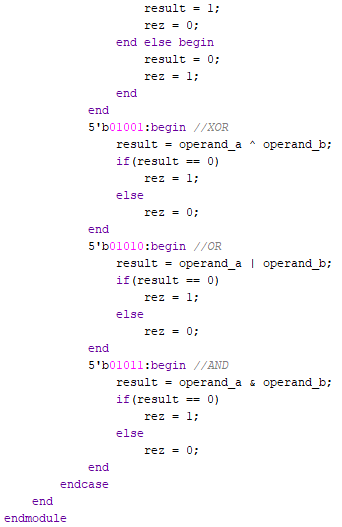
Aşağıdaki tabloda ALU’nun desteklediği işlemler, operasyon kodları ve operasyonların açıklamaları verilmiştir.



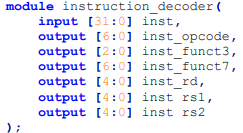
Aşağıdaki kodda rez diye bir logic tanımladık, result\_equal\_zero değişkenine eşitledik. Operasyon kodların sırasına göre 1 girişinde ADD, 2 girişinde SUB, 3 girişinde SLL, 4 girişinde SLR, 5 girişinde SRA, 6 girişinde SEQ, 7 girişinde SLT, 8 girişinde SLTU, 9 girişinde XOR, 10 girişinde OR, 11 girişinde AND işlemini gerçekleştirdik. Ayrıca her giriş için result sıfıra eşitse rezi 1, aksi takdirde rezi sıfıra eşitledik.







Aşağıda Instruction Decoder ünitesinin giriş ve çıkış sinyalleri gösterilmiştir.



Instruction decoder modülünde giriş olarak 32 bitlik instruction word’u alınmaktadır. Çıkışta ise instruction’un parse edilmiş hali, yani decode edilmiş hali çıkış olarak verilmektedir.

• Opcode, instruction’un ilk 7 bitini yani [6:0]’ı temsil etmekte

• Func3, instruction’un 14-12 bitleri arasını [14:12];

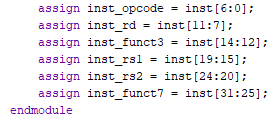
• Func7, instruction’un 31-25 bitleri arasını [31:25];

• Rd, instruction’un 11-7 bitleri arasını [11:7];

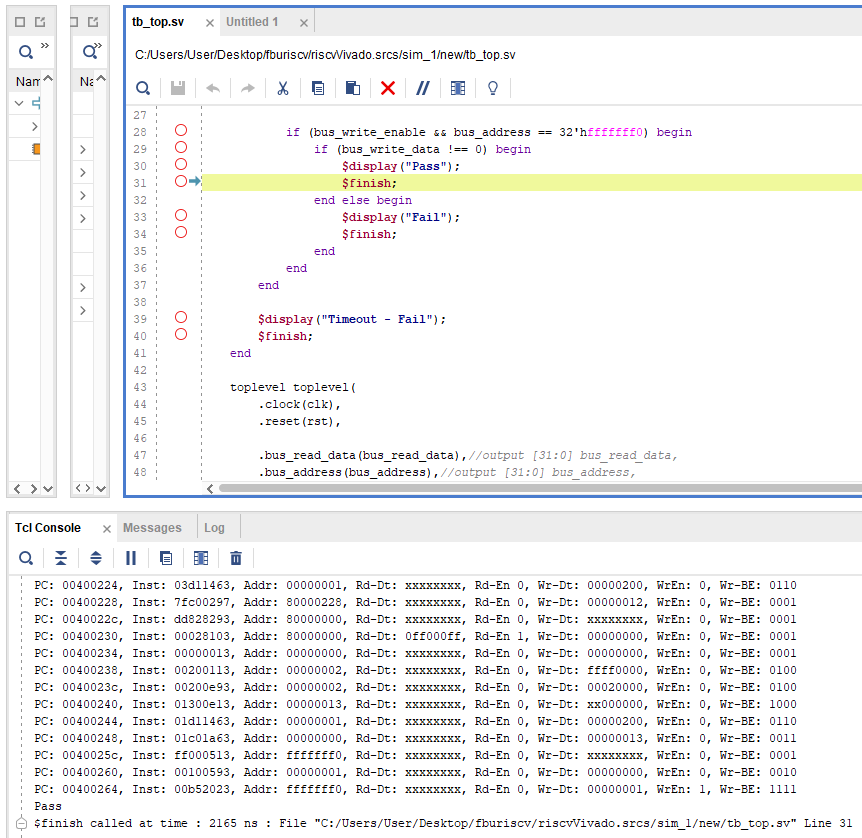
• RS1, instruction’un 19-15 bitleri arasını [19:15];

• RS2, instruction’un 24-20 bitleri arasını [24:20];

Buna göre instruction sinyalini parçalayarak ilgili sinyallerin üzerlerine atama yapılmıştırr.



Aşağıdaki tb\_top dosyasında hazır olan kodu çalıştırdığımızda pass çıktısını vermektedir. O halde kodumuz doğru çalışmıştır.



# Kullanılan Yazılım

Tasarımları yapabilmek için Xilinx tarafından geliştirilen Vivado Design Suite yazılımını kullandık. Vivado Design Suite, HDL tasarımlarının sentezi ve analizi için üretilmiş bir yazılım paketidir ve Xilinx ISE’nin yerine çip geliştirme ve üst düzey sentez sistemi için ek özellikler sunar. Vivado, tüm tasarım akışının baştan aşağı yeniden yazılmasını ve yeniden düşünülmesini temsil eder. SystemVerilog dili kullanılarak Vivado üzerinde tasarımlar yapıldı. IEEE 1800 olarak standartlaştırılmış SystemVerilog ise elektronik sistemleri modellemek, tasarlamak, simüle etmek, test etmek ve uygulamak için kullanılan bir donanım açıklaması ve donanım doğrulama dilidir.

# Sonuçlar

Geliştirilen RISC-V işlemcisi ADD, SUB, SLL, SLR, SRA, SEQ, SLT, SLTU, XOR, OR ve AND işlemlerini destekliyor. Bu proje ile birlikte RISC-V işlemcisinin tasarımını, nasıl çalıştığını, hangi işlemleri yapabildiğini ve mimarisini öğrenmiş olduk. SystemVerilog dilini iyice kavrayıp kendimizi geliştirdik. Bu işlemcide tamamlanmış ALU ve Instruction\_decoder tasarımını tamamladık. Bu tasarladığımız işlemciyi Vivado programında test edip doğru çalışıp çalışmadığını kontrol ettik ve tasarımın başarıyla çalıştığını gördük.

##### Proje Ekibi

Ben Bedirhan İleri, 06/12/2001 tarihinde Üsküdar’da doğdum. Prof. Dr. Nabi Avcı Anadolu Lisesi mezunuyum. Fenerbahçe Üniversitesi’nde Bilgisayar Mühendisliği 3. sınıf öğrencisiyim.

Ben Özgür Çakmak, 30/05/2002 tarihinde Ümraniye’de doğdum. Çamlıca Final Anadolu Lisesi mezunuyum. Fenerbahçe Üniversitesi’nde Bilgisayar Mühendisliği 2. sınıf öğrencisiyim.

##### Referans Dosyalar

<https://youtu.be/uyKF_3W9k6E>

<https://github.com/DeveloperBedirhan/RISC-V_Tabanli_Islemci_Tasarimi>

##### Kaynaklar

*<http://www.levent.tc/files/courses/computer_architecture/project/BLM202_proje_spesifikasyonlari.pdf>*