TOBB Ekonomi ve Teknoloji Üniversitesi
Bilgisayar Mühendisliği Bölümü
BİL 361 – Bilgisayar Mimarisi ve Organizasyonu

2 Kasım 2023
2023 – 2024 Öğretim Yılı
Güz Dönemi
Ödev 1

[100 puan] Çok Vuruşluk RISC-V İşlemci Tasarımı

Bu ödevde RISC-V RV32I buyruk kümesi mimarisindeki bazı buyrukları yürütebilen çok vuruşluk 32-bit işlemci tasarlayacaksınız.

İşlemcinizin desteklemesi gereken buyruklar aşağıdaki tabloda belirtilmiştir. İşemcide olacak buyrukların ayrıntılı açıklamalarına ve bu tablonun tüm RISCV buyruklarını içeren haline "https://riscv.org/wp-content/uploads/2017/05/riscv-spec-v2.2.pdf" bağlantısından ulaşabilirsiniz.

RV32I Base Instruction Set

imm[31:12]				rd	0110111	LUI
imm[31:12]				rd	0010111	AUIPC
imm[20|10:1|11|19:12]				rd	1101111	JAL
imm[11:0]		rs1	000	rd	1100111	JALR
imm[12|10:5]	rs2	rs1	000	imm[4:1|11]	1100011	BEQ
imm[12|10:5]	rs2	rs1	001	imm[4:1|11]	1100011	BNE
imm[12|10:5]	rs2	rs1	100	imm[4:1|11]	1100011	BLT
imm[11:0]		rs1	010	rd	0000011	LW
imm[11:5]	rs2	rs1	010	imm[4:0]	0100011	SW
imm[11:0]		rs1	000	rd	0010011	ADDI
0000000	rs2	rs1	000	rd	0110011	ADD
0100000	rs2	rs1	000	rd	0110011	SUB
0000000	rs2	rs1	110	rd	0110011	OR
0000000	rs2	rs1	111	rd	0110011	AND
0000000	rs2	rs1	100	rd	0110011	XOR

- RISCV Buyruk Mimarisi tüm buyrukların verimli ve etkin çalışması üzerine tasarlanmıştır. Fakat problemi kolaylaştırmak adına size 15 adet buyruk ile çalışabilecek bir işlemci tasarlama görevi verilmiştir. Buyruk mimarisinde belirtilen ve bu 15 buyruk dışındaki şartlardan dolayı oluşan tüm kısıtlamaları ve kural dışı durumları yok sayabilirsiniz.

- Yapacağınız işlemci belirtilen tüm buyruklar için 3 çevrimde doğru işlemleri tamamlamalıdır. Bu aşamalar sırasıyla "**Getir**", "**ÇözYazmaçOku**" ve "**YürütGeriYaz**" olacaktır ve her biri yalnızca 1 çevrim sürmelidir. Aşamaların işlevleri aşağıda detaylıca açıklanmıştır. Kafanızı karıştıran durumlarda piazzada ödev gönderisini kullanmaya çekinmeyin.
    - **Getir:** İşlenmesi gereken buyruğun program sayacı belleğe gönderilir ve gelen buyruk bir sonraki aşama için kayıt edilir. Getir aşaması istek yapıldıktan sonra saatin yükselen kenarında program sayacını günceller.
    - **ÇözYazmaçOku:** Getirilen buyruk mikroişlemlere ayrılır (örn, AMB için yapılacak işlemin kayıt edilmesi) ve ilgili yazmaçlar okunarak bir sonraki aşama için kayıt edilir. Dallanma işlemlerini bu aşamada **sonuçlandırmayın**.
    - **YürütGeriYaz:** Çözülen mikroişlemlere göre buyruk yürütülür ve durum (yazmaçlar, ana bellek) güncellenir. Dallanma yapılıyorsa YürütGeriYaz aşaması bir sonraki buyruk için program sayacını düzgün şekilde günceller.

- İşlemci ile ilgili genel kısıtlar aşağıdaki gibidir, bu tanımlara uyduğunuzdan emin olmak için kaynaklar altındaki taslak tasarım dosyalarını kullanabilirsiniz.
  - İşlemci ve bellekte bayt adresleme kullanılır.
  - 32 adet 32 bitlik yazmaç bulunur ve değişkenin adı "**yazmac_obegi"**dir.
  - **simdiki_asama_r** yazmacı her zaman o çevrimde yürütülen aşamayı gösterir.
  - **ps_r** yazmacı her zaman işlenecek sıradaki buyruğun adresini gösterir (dallanmalarla değişebilir).
  - 2048 satırlı 32 bitlik elemanlardan oluşan ana bellek, işlemciye "**bellek_adres**" çıkışında bulunan adresteki veriyi verecektir. Ana bellekteki ilk elemanın adresi "0x8000_0000" olup programın ilk buyruğu her zaman bu adreste yer alacaktır.

  Dikkat: Çok vuruşluk bir işlemci tasarladığınız için **bellek_adres** çıkışında **Getir** aşaması için program sayacı olup buyruk getirilecek, **ÇözYazmaçOku** aşamasında ise **LOAD** ve **STORE** adresleri olup belleğe veri yazma / okuma işlemleri yapılacaktır.

  Not: Önemli bir nokta değil ancak dikkatinizi çektiği üzere programlar istenirse kendi buyruklarının üzerine veri yazarak programı bozabilirler / değiştirebilirler. İşletim sistemleri normal yazdığınız programlarda bunu zorlaştırsa da bu durum olağandır ve engellenmesine gerek yok.

[100 Puan] Yukarıda ayrıntıları verilen çok vuruşluk işlemciyi verilog dilinde "**islemci.v**" ile tek bir modül olarak tasarlayınız. "**islemci.v**" için taslak kodunu piazza kaynaklar kısmında bulabilirsiniz. Bu dosyaya giriş / çıkışlar, yazmac_obegi, simdiki_asama_r ve ps_r değişkenlerinin **isimleri** dışında istediğiniz değişikliği yapabilirsiniz (sadece isimler önemli, atamaları ve genel kodu değiştirmenizde bir sakınca yok).
işlemci.v giriş çıkışları sırasıyla aşağıdaki gibidir.

- **clk** : 1 bitlik saat girişi
- **rst** : 1 bitlik işlemcinizi başlangıç durumuna geri çeviren giriş
- **bellek_adres** : 32 bitlik belleğe giden adres çıkışı
- **bellek_oku_veri** : 32 bitlik bellekten okunan veri girişi
- **bellek_yaz_veri** : 32 bitlik belleğe yazılacak veri çıkışı
- **bellek_yaz** : 1 bitlik belleğe yazma yapılacağını gösteren çıkış

- Bellekte yazma işlemleri sıralı mantık, okuma işlemleri ise **kombinasyonel** mantıkla çalışmaktadır. Yani ilgili çıkışlar atandıktan sonra **yazma işlemleri saatin yükselen kenarında** gerçekleşirken **okuma işlemlerinde saatin yükselen kenarı beklenmeden bellek_oku_veri** girişinde olacaktır. Kaynaklar altında "**anabellek.v**" dosyasını inceleyip testlerinizde kullanabilirsiniz.

Önemli Not: Bellekten sadece 4 bayt hizalı okumalar yapılacağını varsayabilirsiniz. Yani 0x8000_0002 gibi bir adres üzerinde **okuma** veya **yazma** asla gerçekleştirilmeyecektir. Ek olarak 0x8000_0000 adresi bellekteki ilk 4 baytın başlangıcını, 0x8000_0004 ise ikinci 4 baytın başlangıcını adresleyecektir.

Ödev Gönderimi ve Formatı

Ödevinizde işlemci tasarımı için yazdığınız Verilog dosyaları ve **islemci** tasarımınızın sentez çıktısı bulunmalıdır. Ödevinizi uzak platformuna **sıkıştırmadan** yükleyeceksiniz.

Gönderdiğiniz tüm tasarımların **sentezlenebilir** verilog standartlarında yazılması beklenmektedir. Tasarımlarınızın keyfi bir FPGA kartı için sentezlenebilir olduğunu kontrol edin. Modülleriniz simülasyon üzerinden kontrol edileceğinden implementasyon ve bitstream aşamaları **gerçekleştirilmeyecek**. Bu aşamalar için "**constraints**" dosyaları oluşturmakla **vakit kaybetmeyin**.

Sentez çıktılarını almak için bir çok farklı yol var ancak daha önce yapmadıysanız aşağıdaki adımları izleyebilirsiniz:

-Proje ağacından ilgili modüle (islemci.v) sağ tıklayıp "**Set as Top**"ı seçin. Eğer zaten ilgili modül kalın fontla yazılıysa (zaten hedef olarak seçiliyse) bu adımı atlayabilirsiniz.
-Vivado'da sol kısımdaki menüden "**SYNTHESIS"** altında bulunan "**Run Synthesis**" seçeneğine tıklayın.
-Senteziniz tamamlandığında sentez raporunuzu (vds dosyasını) kayıt edin ve gönderiminize ekleyin. Raporunuz sentez bittikten sonra aşağıdaki yolda oluşacaktır.

"**{PROJE_KLASORU}/{PROJE_ISMI}.runs/synth_1/{MODUL_ISMI}.vds**"

Son Teslim Tarihi: 17 Kasım 2023, 23:59