

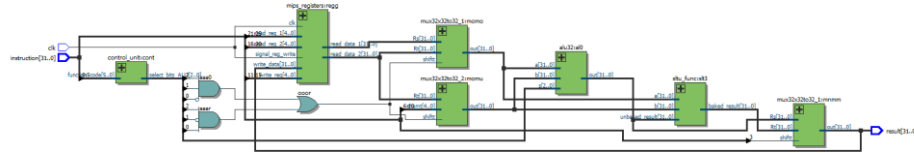
# CSE 331 Computer Organization

## Project 3 – R-type Single cycle MIPS with Structural Verilog

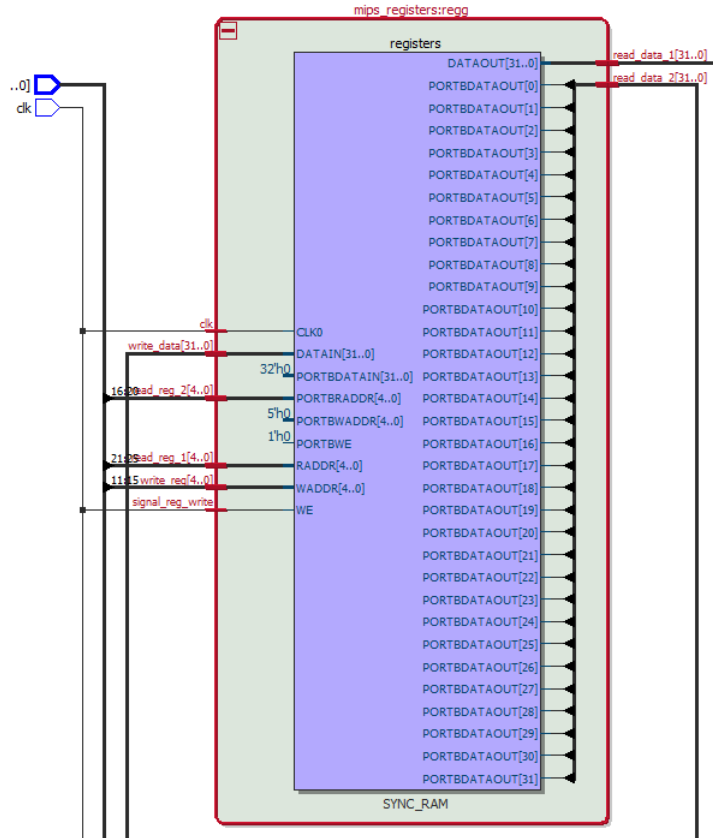
Due date: December 7, Friday 23:59 (Moodle)

### Rapor

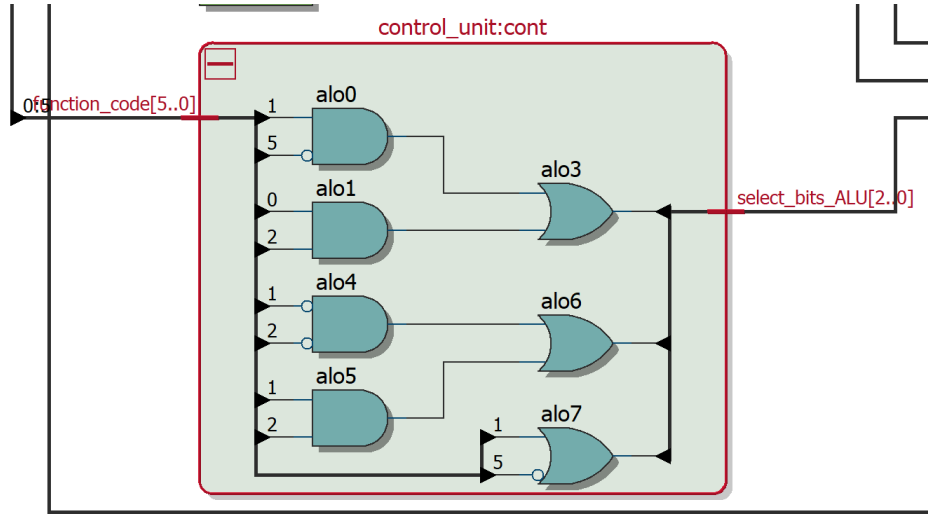
Muhammetalp ERDEM 151044006



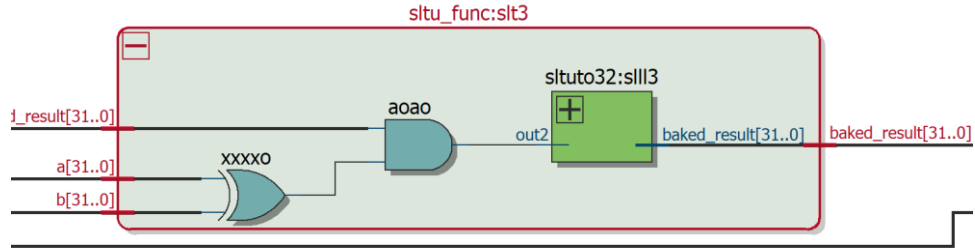
Projede kullanılan modüllerin genel görünümü üstte verildiği gibidir. Bu Raporda Proje 2'den ekstra olarak kullanılan modüller açıklanacaktır.



Mips\_registers.v : Bu modülde register işlemleri yapılmıştır. Yazma işlemi için bir sinyal kullanılmış, okuma işlemi ise direkt yapılmıştır. (Ders slaytı örnek alındı.)



**Control\_unit.v :** Bu modül instruction'ın func kısmını alıp önceki projede tasarladığımız Alu için select bit üretir. Bu dönüşüm truth tablosu ve karnough map sadeleştirmeleri ile elde edilmiştir.



**Sltu\_func.v :** Alu'dan çıkan subtract sonucunu alır. Bu sonucu kullanarak set less than unsigned sonucu üretir. İki inputun most-significant biti eşit değilse ve sonucun most-significant biti 1 ise pozitif sonuç döndürür.

**Slto32.v :** Sltu\_func.v içinde kullanılan yardımcı modül. 1 bitlik inputu alıp başına 31 adet 0 ekleyerek 32 bitlik hale dönüştürür.

**Shamtto32.v :** Shift amount'u 32 bitlik hale dönüştürür.

**Mux32x32to32\_1.v :** shift operasyonu yapılacak ise ilk inputa Rt verir, diğer türlü Rs.

**Mux32x32to32\_2.v :** shift operasyonu yapılacak ise ikinci inputa Shamt verir, diğer türlü Rt.

```
Transcript
VSIM 2> step -current
# opcode = 00, rs = 00, rt = 00, rd = 00, shamt = 00, funct = 00 , aluSel = 6
# result = 00000000, aluF = 00000000
# $rs = 00000000
# $rt = 00000000
# opcode = 00, rs = 08, rt = 09, rd = 10, shamt = 00, funct = 20 , aluSel = 2
# result = 00000005, aluF = 00000005
# $rs = 00000002
# $rt = 00000003
# opcode = 00, rs = 08, rt = 09, rd = 11, shamt = 00, funct = 21 , aluSel = 2
# result = 00000005, aluF = 00000005
# $rs = 00000002
# $rt = 00000003
# opcode = 00, rs = 08, rt = 09, rd = 12, shamt = 00, funct = 24 , aluSel = 0
# result = 00000002, aluF = 00000002
# $rs = 00000002
# $rt = 00000003
# opcode = 00, rs = 08, rt = 09, rd = 13, shamt = 00, funct = 27 , aluSel = 7
# result = ffffffff, aluF = ffffffff
# $rs = 00000002
# $rt = 00000003
# opcode = 00, rs = 08, rt = 09, rd = 14, shamt = 00, funct = 25 , aluSel = 1
# result = 00000003, aluF = 00000003
# $rs = 00000002
# $rt = 00000003
# opcode = 00, rs = 08, rt = 09, rd = 15, shamt = 00, funct = 2b , aluSel = 4
# result = 00000000, aluF = 00000000
# $rs = 00000002
# $rt = 00000003
# opcode = 00, rs = 00, rt = 09, rd = 16, shamt = 12, funct = 00 , aluSel = 6
# result = 000c0000, aluF = 000c0000
# $rs = 00000000
# $rt = 00000003
# opcode = 00, rs = 08, rt = 09, rd = 18, shamt = 00, funct = 22 , aluSel = 4
# result = ffffffff, aluF = ffffffff
# $rs = 00000002
# $rt = 00000003
# opcode = xx, rs = xx, rt = xx, rd = xx, shamt = xx, funct = xx , aluSel = x
# result = xxxxxxxx, aluF = xxxxxxxx
# $rs = xxxxxxxx
# $rt = xxxxxxxx
# opcode = 00, rs = 08, rt = 09, rd = 18, shamt = 1c, funct = 23 , aluSel = 4
# result = ffffffff, aluF = ffffffff
# $rs = 00000002
# $rt = 00000003
# opcode = xx, rs = xx, rt = xx, rd = xx, shamt = xx, funct = xx , aluSel = x
# result = xxxxxxxx, aluF = xxxxxxxx
# $rs = xxxxxxxx
# $rt = xxxxxxxx
# 10 tests completed.
.
```

Modelsim sonuçları yukarıda görüldüğü gibidir, testbench’de dosya okurken sıkıntı oluştu bu nedenle instruction ve registerları el ile girdim, çalışmaktadır.

```

begin
    clk = 0;
    clk2 = 0;
    //$readmemb("input.tv", testVectors);
    testVectors[0] = 32'h01098020;
    testVectors[1] = 32'h01098821;
    testVectors[2] = 32'h01099024;
    testVectors[3] = 32'h01099827;
    testVectors[4] = 32'h0109A025;
    testVectors[5] = 32'h0109A82B;
    testVectors[6] = 32'h0009B480;
    testVectors[7] = 32'h0009B942;
    testVectors[8] = 32'h0109C022;
    testVectors[9] = 32'h0109C723;

    //$readmemb("registers.mem", mipss.regg.registers);

    mipss.regg.registers[0] = 32'b00000000000000000000000000000000;
    mipss.regg.registers[1] = 32'b00000000000000000000000000000001;
    mipss.regg.registers[2] = 32'b00000000000000000000000000000010;
    mipss.regg.registers[3] = 32'b000000000000000000000000000000011;
    mipss.regg.registers[4] = 32'b000000000000000000000000000000011;
    mipss.regg.registers[5] = 32'b000000000000000000000000000000011;
    mipss.regg.registers[6] = 32'b000000000000000000000000000000000;
    mipss.regg.registers[7] = 32'b000000000000000000000000000000001;
    mipss.regg.registers[8] = 32'b000000000000000000000000000000010;
    mipss.regg.registers[9] = 32'b000000000000000000000000000000011;
    mipss.regg.registers[10] = 32'b000000000000000000000000000000011;
    mipss.regg.registers[11] = 32'b000000000000000000000000000000011;
    mipss.regg.registers[12] = 32'b000000000000000000000000000000000;
    mipss.regg.registers[13] = 32'b000000000000000000000000000000001;
    mipss.regg.registers[14] = 32'b000000000000000000000000000000010;
    mipss.regg.registers[15] = 32'b000000000000000000000000000000011;
    mipss.regg.registers[16] = 32'b000000000000000000000000000000011;
    mipss.regg.registers[17] = 32'b000000000000000000000000000000011;
    mipss.regg.registers[18] = 32'b000000000000000000000000000000000;
    mipss.regg.registers[19] = 32'b000000000000000000000000000000001;
    mipss.regg.registers[20] = 32'b000000000000000000000000000000010;
    mipss.regg.registers[21] = 32'b000000000000000000000000000000011;
    mipss.regg.registers[22] = 32'b000000000000000000000000000000011;

```

Teşekkürler.