Gebze Technical University
Computer Engineering

CSE 331

ASSIGNMENT 4 REPORT

CELAL CAN KAYA
161044014

Course Assistant:Fatma Nur Esirci

ŞEMATİK TASARIMLAR

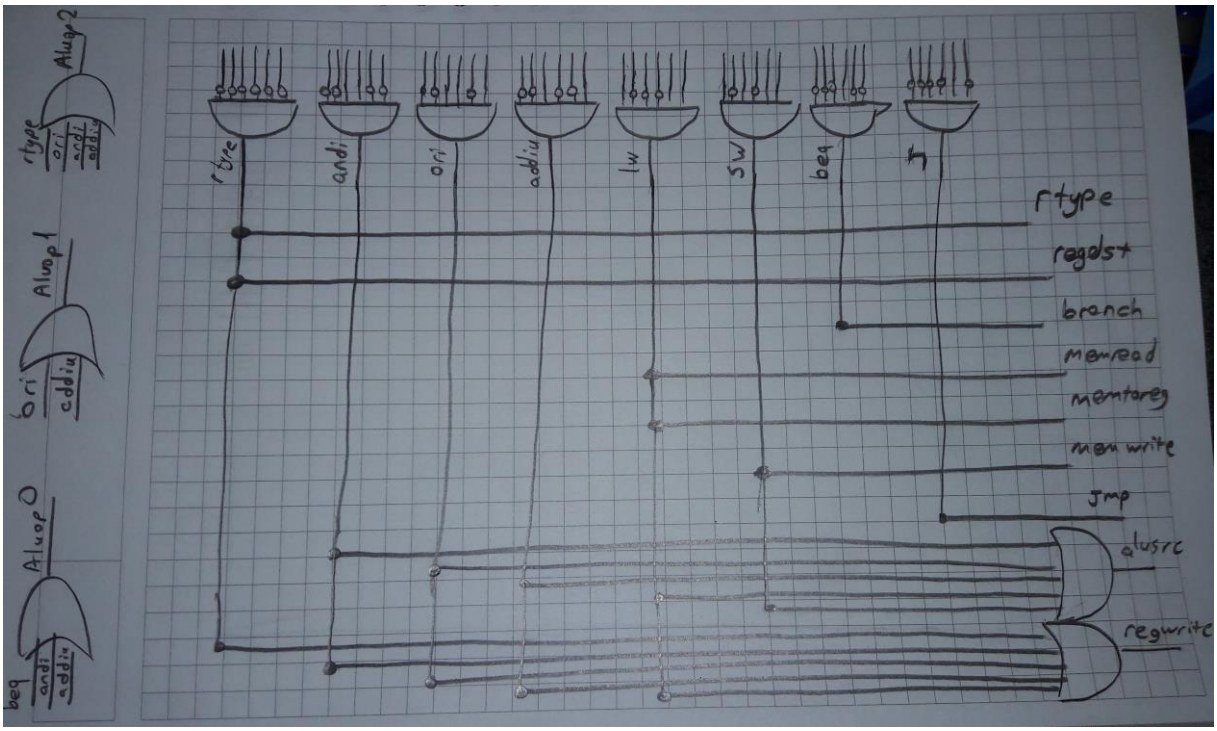
INSTRUCTION	ALUOP	ALUOP1	ALUOP	F5	F4	F3	F2	F1	F0	OPERATION
lw	0	0	0	X	X	X	X	X	X	010
sw	0	0	0	X	X	X	X	X	X	010
j	0	0	0	X	X	X	X	X	X	XXX
beq	0	0	1	X	X	X	X	X	X	100
add	1	0	0	1	0	0	0	0	0	010
addu	1	0	0	1	0	0	0	0	1	010
sub	1	0	0	1	0	0	0	1	0	100
subu	1	0	0	1	0	0	0	1	1	100
and	1	0	0	1	0	0	1	0	0	000
or	1	0	0	1	0	0	1	0	1	001
nor	1	0	0	1	0	0	1	1	1	111
sll	1	0	0	0	0	0	0	0	0	110
srl	1	0	0	0	0	0	0	1	0	101
sltu	1	0	0	1	0	1	0	1	1	100
andi	1	0	1	X	X	X	X	X	X	000
ori	1	1	0	X	X	X	X	X	X	001
addiu	1	1	1	X	X	X	X	X	X	010

OP[0] = [ALUOP1 & ~ALUOP2] + [[ALUOP2 & ~ALUOP1 & ~ALUOP0] & [[F2 & F0] + [F1 & ~F5]]]

OP[1]=[~ALUOP2 & ~ALUOP0] + [ALUOP1 & ALUOP0] + [[ALUOP2 & ~ALUOP1 & ~ALUOP0] & [[~F2 & ~F1] + [F2 & F1]]]

OP[2] = [~ALUOP2 & ALUOP0] + [[ALUOP2 & ~ALUOP1 & ~ALUOP0] & [F1 + ~F5]]

Şekil 1- ALU Control Unit



Şekil 2- Control Unit

VERİLOG MODÜLLERİ

Mips32_single_cycle – Top level modülüm.Input olarak sadece clock alıyor ve instructionları sırasıyla okuyup registers.mem ve data.mem üzerinde değişiklikler yapıyor.

Control_unit – 5 Bitlik Op code'u kullanarak datapath için gerekli olan sinyalleri üretiyorum ve Alunun yapacağı operasyonu seçmesi için 3 adet ALUOP sinyali üretip alu control unite gönderiyorum.

Alu_control_unit – 5 Bitlik Function code'un ve 3 Bitlik ALUOP'un doğruluk tablosunu kullanarak Select bitinin hangi değişkenlere bağlı olduğunu bulup formülünü çıkardım.Daha sonra o formülleri kullanarak alu için select bitlerini üreterek Alu'ya gönderdim.

Next_program_counter – Program counter'ı jump ya da branch oluyorsa ona göre arttırıyorum.Eğer ikiside yoksa bir sonraki instructionu almak için program counter'ı 1 arttırıyorum.

Data_memory – "data.mem" dosyasındaki memory dosyalarını memory'nin içine atıyorum ve memory'e bir şey yazılacağı veya okunacağı zaman o işlemleri gerçekleştiriyorum.

Instruction_memory – "instruction.mem" dosyasındaki instructionlar instruction'ın içine atıyorum ve program counter değiştiği zaman program counter'ın işaret ettiği instructionı işleme sokuyorum.

Sign_extender – 16 Bitlik immediate'yi 32 bit yapıyorum

Extend_shamt – 5 Bitlik shamt'yi 32 bit yapıyorum.

Extend_sltur – 1 Bitlik sltur'u 32 bit yapıyorum.

Mips_registers – registers.mem dosyasındaki registerları "registers" ın içine atıyorum.Regdst sinyaline göre Rd'nin alınıp alınmayacağını kontrol ediyorum.Daha sonraki işlemler yapıldıktan sonra gelen Write data'yı Regwrite == 1 oldugu zaman write_reg'in gösterdiği registera yazıyorum.

Alu32 – 8:1 Mux Kullanarak seçilmiş olan operasyonu output olarak veriyorum.

_2mux – 1 Bitlik 2:1 Mux

_32Bit_2mux – 32 Tane 2:1 Mux Kullanılarak yapılmış 32 Bitlik 2:1 Mux

_32Bit_4mux – 3 Tane 32 Bitlik 2:1 Mux Kullanılarak yapılmış 32 Bitlik 4:1 Mux

_32Bit_8mux – 2 Tane 32 Bitlik 4:1 Mux ve 1 Tane 32 Bitlik 2:1 Mux Kullanılarak yapılmış 8:1 Mux. Alu'da select bitine göre hangi işlemin yapılacağını seçmek için yazdım.

_32Bit_and – 32 Bitlik And

_32Bit_or – 32 Bitlik Or

_32Bit_xor – 32 Bitlik Xor

_32Bit_nor – 32 Bitlik Nor

_32Bit_right_shift – 2:1 Lik Muxlar kullanılarak yapılmış Aritmetik right shift modülü. Kaç bit kaydırılacağını seçmek için B inputunun ilk 5 bitine bakıyorum. İlk 5 Bitten sonraki bitler 32'nin tam katı olacağından onları shifte dahil ettiğimizde sonuca herhangi bir etki yapmayacağından dolayı sadece ilk 5 bit üzerinden işlem yaptım.

_32Bit_left_shift – 2:1 Lik Muxlar kullanılarak yapılmış Logical left shift modülü. Kaç bit kaydırılacağını seçmek için B inputunun ilk 5 bitine bakıyorum. İlk 5 Bitten sonraki bitler 32'nin tam katı olacağından onları shifte dahil ettiğimizde sonuca herhangi bir etki yapmayacağından dolayı sadece ilk 5 bit
üzerinden işlem yaptım.

Half_adder – 2 Tane 1 Bitlik Sayıyı toplayıp, toplamı ve carry bitini veren modül

Full_adder – Half adder kullanılarak yapılmış 2 Tane 1 Bitlik Sayı ve Carry bitini toplayıp, toplamı ve carry bitini veren modül

_32Bit_adder – Full adderlar kullanılarak yapılan ve M Inputu 0 verildiğinde 32 bit toplama yapan, M Inputu 1 verildiğinde 32 bit çıkarma yapan modül

SİMÜLASYON SONUÇLARI

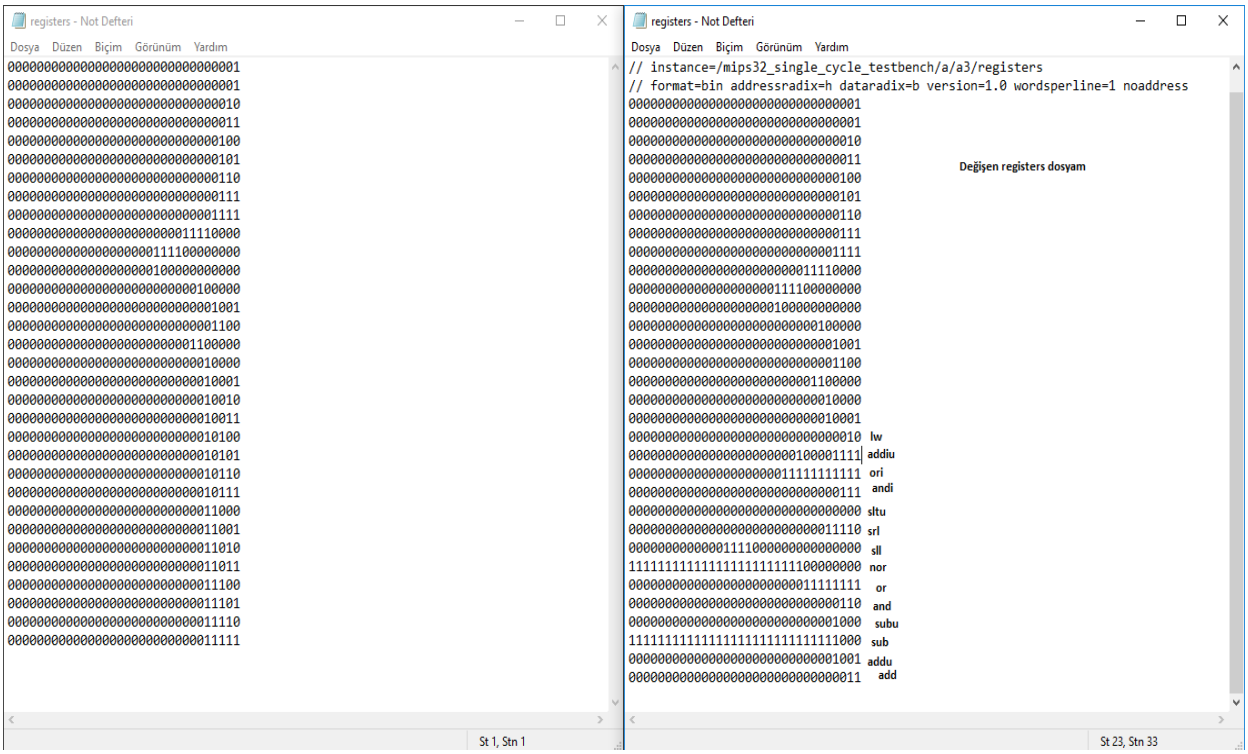
Modelsim

```
# Instruction: 10001100010100100000000000000000, S: 010, AluR: 00000000000000000000000000000010 Program C:00000000000000000000000000001101, Zero: 0
# Instruction: 10101100000000110000000000000000, S: 010, AluR: 00000000000000000000000000000001 Program C:00000000000000000000000000001110, Zero: 0
# Instruction: 00010000001000010000000000000111, S: 100, AluR: 00000000000000000000000000000000 Program C:00000000000000000000000000001111, Zero: 0  1
# Instruction: 00010000001000100000000000000011, S: 100, AluR: 11111111111111111111111111111111 Program C:00000000000000000000000000010110, Zero: 1
# Instruction: 00001000000000000000000000011111, S: 010, AluR: 00000000000000000000000000000010 Program C:00000000000000000000000000010111, Zero: 0  2
# Instruction: 00000000001000101111100000100000, S: 010, AluR: 00000000000000000000000000000011 Program C:00000000000000000000000000011111, Zero: 0
```

1 -> Branch yapıyorum. İlk register ve ikinci register contenti aynı olduğu için branch yapılıyor.

2 -> Jump yapıyorum. Program counterın değişiminden görebilirsiniz.

“Registers”.mem dosyası



Sw instructionu için ise data.mem dosyasını açıp ilk satırı kontrol edebilirsiniz.4. Register’ı ilk satırdaki memory’e yazıyorum.

Tüm modüllerim testlerime göre düzgünce çalışıyor. Sonuçları “registers.mem” dosyasına ve “data.mem” dosyasına tekrar yazdım. Testbench’imde clock’u elle arttırdım, zamanım kalmadığı için otomatik şekilde artması için düzenleyemedim.Bu nedenle modelsimde çalıştırdıktan sonra en altta xxxxxxxxxxxxxx’li outputlar olucak, instructionlar bittiği için xxxxxxxxx gösteriyor. Gelen inputlar sürekli olarak değişmemesi için her instruction’un sonucunu sırayla 32. Registerdan itibaren geriye doğru yazdırdım.Inputlar olarakta genel olarak “registers.mem”dosyasındaki ilk registerları kullandım. “registers.mem” ve “data.mem” dosyasının içeriğini değiştirmeden yolladım.Modelsim üzerinden çalıştırdıktan sonra tekrar “registers.mem” ve “data.mem” dosyasına bakarak değişimleri görebilirsiniz.
Ödev düzgün çalışmadığı için 1 gün geç gönderip ödevi düzelttim.Yoğun bir dönem olduğu için yorum satırı ekleyemedim ve ayrıca moodle’da gönderim kapandığı için mail olarak atıyorum.