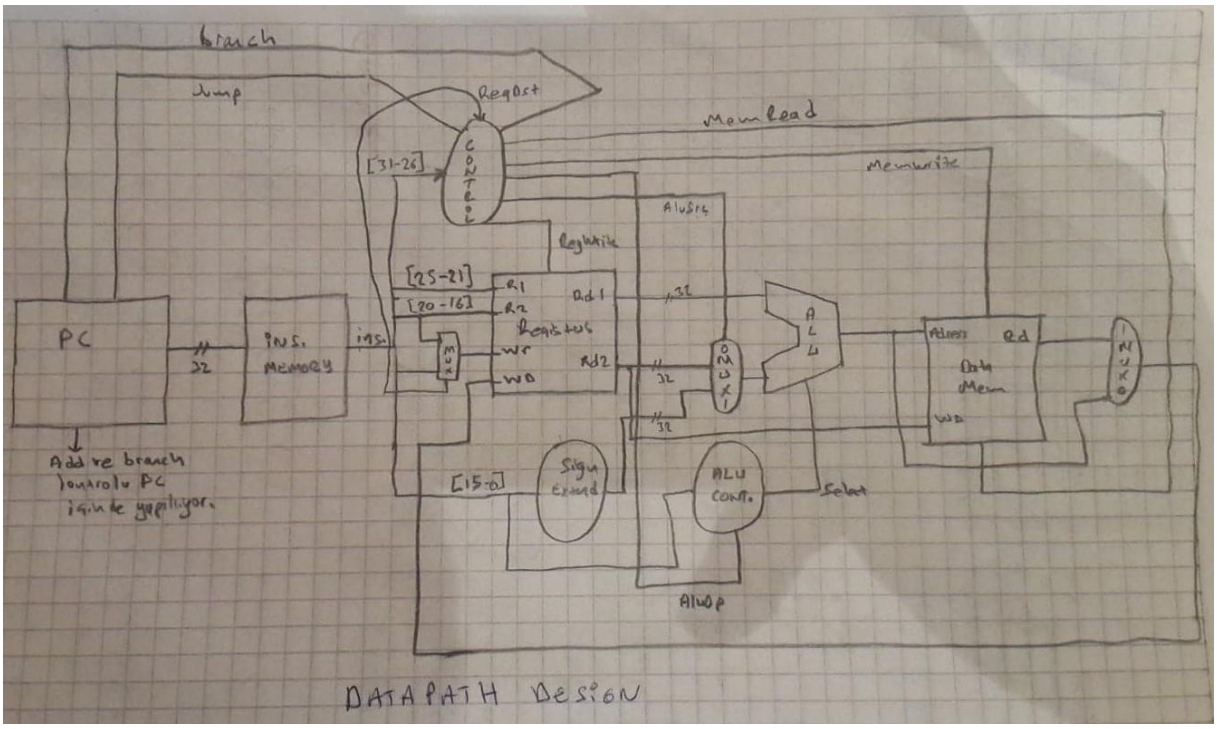
HOMEWORK 4– 151044085 – EMRE KAVAK

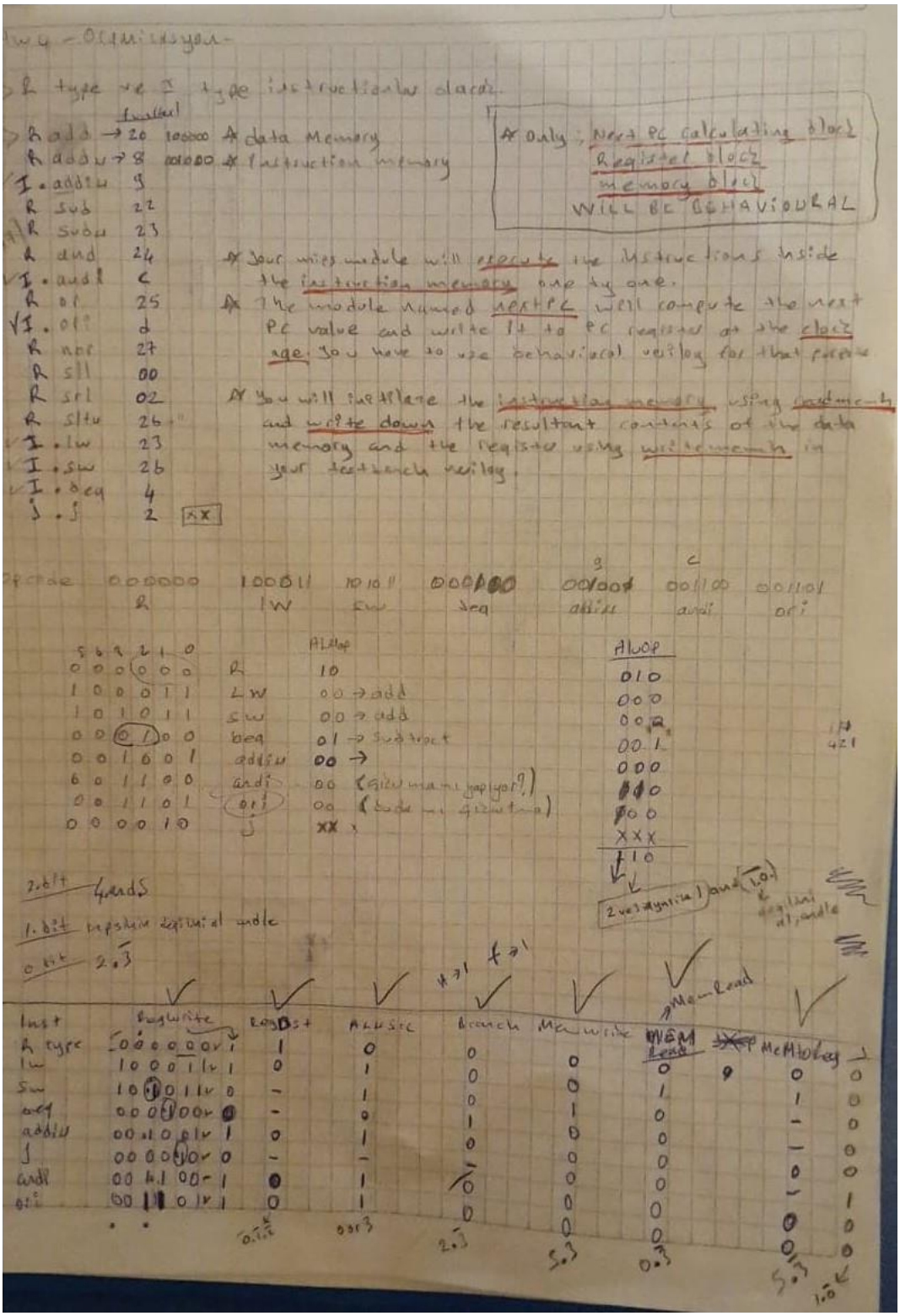
MODÜLLER

1-) mips32_single_cycle : Bu modül parametre olarak 2 tane clock sinyali alıyor.Birisi long, diğeri short clocklar (30ns,10ns). Alınan sinyaller diğer modüllere gönderiliyor. Bu modül içinde diğer tüm modüller çağırılarak tüm datapath kontrol edilmiş oluyor. Datapath PC den başlıyor ve başlangıç olarak instruction memoryden çekeceği adress 0 olarak assign edilmiş durumda. PC den çıkan adress ile instruction memoryden instruction çekiliyor. Daha sonra instruction field lerine ayrılarak control modülüne, register modülüne gönderiliyor. Control unitten çıkan sinyallere göre register modulunden contentler, muxların select bitleri, alunun select bitleri gönderiliyor ve işlem sonuçları instructiona göre dosyalara yazılıyor.



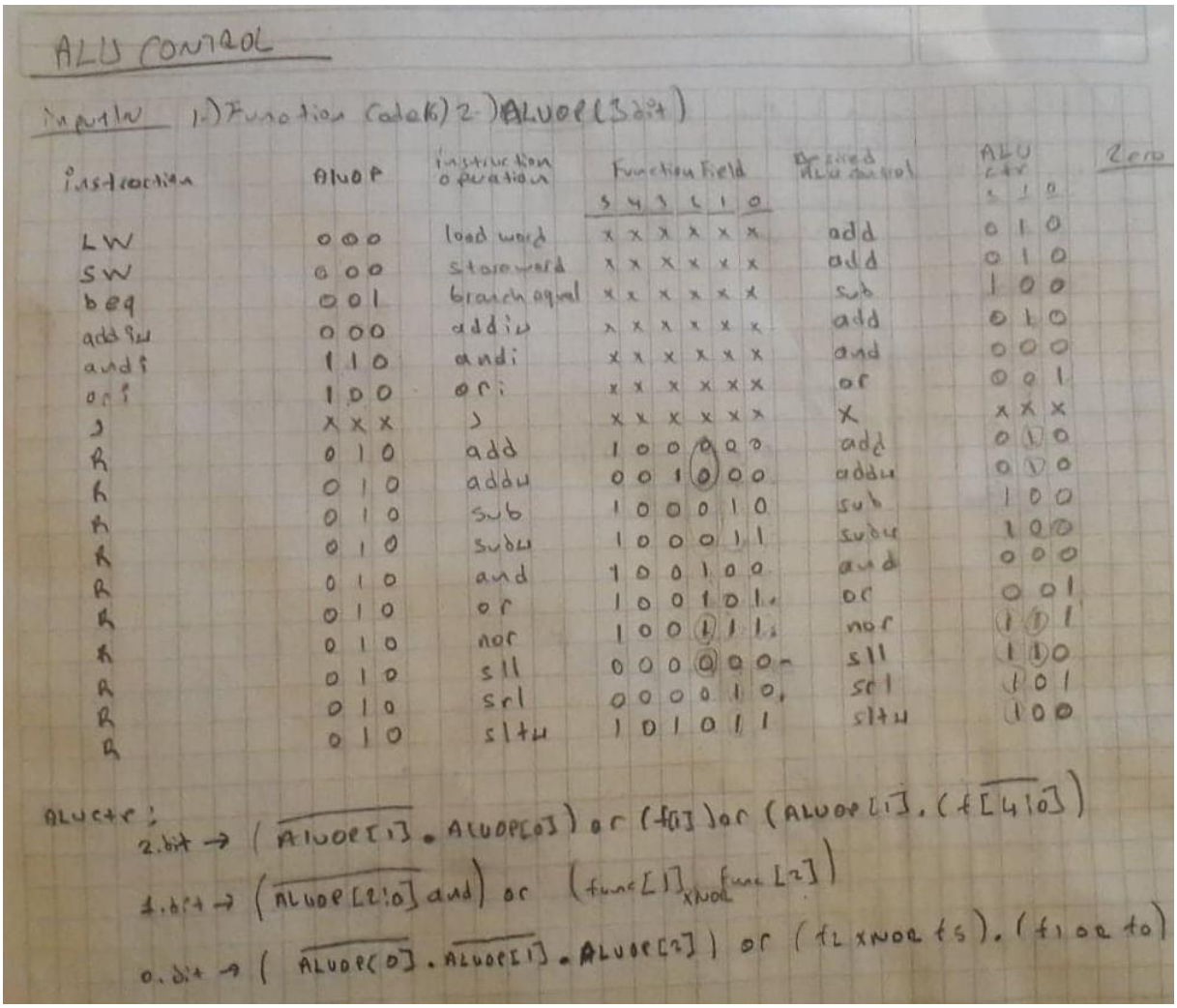
2- PC : Bu modül içinde instruction memoryden çekilecek instructionun adresi belirleniyor. Adres başlangıç olarak 0 atanmış durumda. Daha sonra parametrelerin durumuna göre adress indexi artıyor. Bu modül içinde ayrıca sign extende yapılıyor.

3- Control : Control modulunde , gelen opcode göre çıkış sinyalleri belirleniyor. Bu modülün sinyal belirleme durumları aşağıda ki resimde mevcuttur.

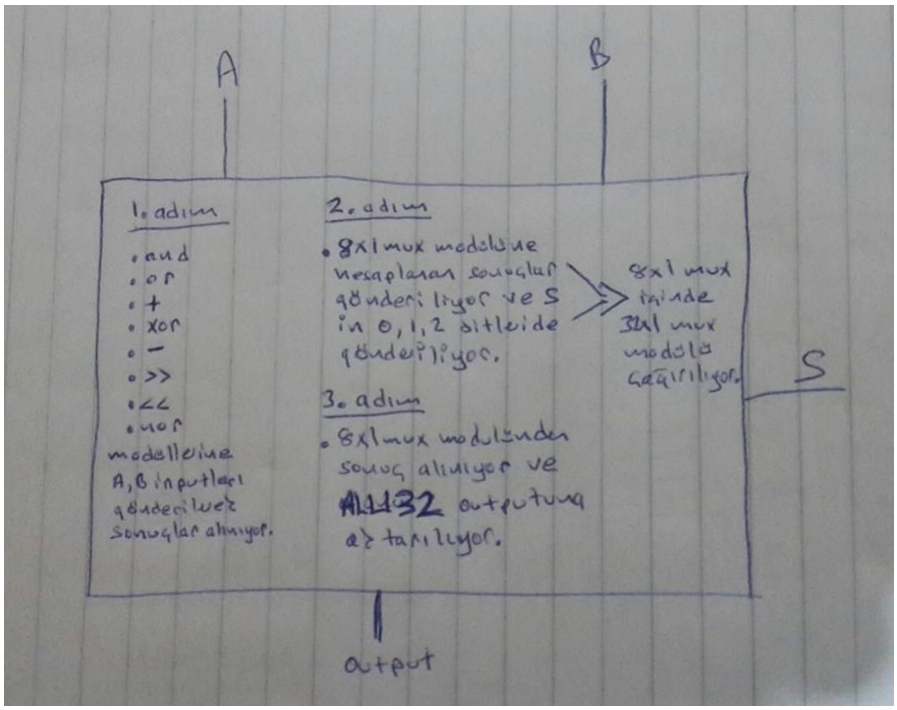


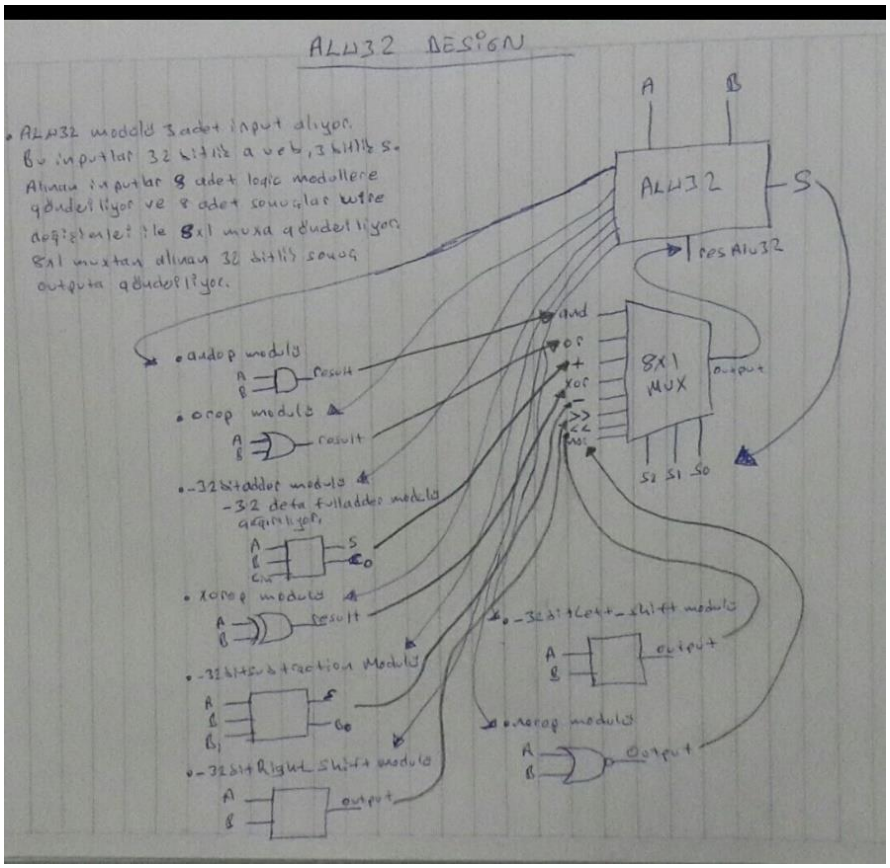
Yukarıda ki resimde sadeleştirme sonuçları ve control modulunun sinyalleri mevcuttur.

4- Alu control : Bu modülde, control unıtten gelen 3 bitlik Aluop ile instructiondan gelen function a göre Alunun select biti seçiliyor. Aşağıda kiResimde gerekli sadeleştirmeler mevcuttur.



5-) alu32 : Bir önceki proje ile hemen hemen aynı modül olup, xor ve sra işlemleri kaldırılmış, yerine srl ve sltu işlemleri eklenmiştir. Bu modülü hatırlayacak olursak, 32 bitlik 2 adet **a** ve **b**, 3 bitlik **s(select)** inputları alınmaktadır. Bu inputlar alındıktan sonra modül içinde yapmamız istenen **8 adet(and,or,+ , -,>>,<<,nor)** işlemin modülleri çağırılmaktadır. Çağırılan bu modüllerin sonuçları **wire** değişkenlerinde tutularak , **_8x1mux** modülüne gönderilmektedir. _8x1 modülü 8 adet 32 bitlik , 1 adet 3 bitlik input almaktadır. Alınan bu inputlar **_32bitmux** modülüne, sırasıyla I0,I1…I7, şeklinde gönderiliyor ve s bitinin sırasıyla s[0], s[1] ve s[2] bitleri gönderilerek hangi inputun outputa gönderileceği belirleniyor. _8x1mux modülünde en son çağırılan _32bitmux modülünün outputu alınarak, _8x1mux modülünün outputuna gönderiliyor. Oluşan bu output, son olarakda alu32 icinde ki resultAlu32 outputuna atanarak istenilen işlemin sonucu elde edilmiş oluyor.


A
B
1. adım
• and
• or
• +
• xor
• −
• >>
• <<
• nor
modellerine
A,B inputları
gönderilerek
sonuçlar alınıyor.
2. adım
• 8x1 mux modülüne
hesaplanan sonuçları
gönderiliyor ve S
in 0,1,2 bitleride
gönderiliyor.
8x1 mux
içinde
32x1 mux
modülü
çağırılıyor.
S
3. adım
• 8x1 mux modülünden
sonuç alınıyor ve
ALU32 outputuna
atanıyor.
output


ALU32 DESIGN
• ALU32 modülü 3 adet input alıyor.
Bu inputlar 32 bitlik a ve b, 3 bitlik s.
Alınan inputlar 8 adet logic modüllere
gönderiliyor ve 8 adet sonuçlar wire
değişkenleri ile 8x1 muxa gönderiliyor.
8x1 muxtan alınan 32 bitlik sonuç
outputa gönderiliyor.
A
B
ALU32
S
res ALU32
• and op modülü
result
• or op modülü
result
• 32 bit adder modülü
-32 defa fulladder modülü
çağırılıyor.
S
Cout
• xor op modülü
result
• 32 bit Subtraction Modülü
• 32 bit Right Shift modülü
output
• 32 bit Left Shift modülü
output
• nor op modülü
Output
and
or
+
xor
−
>>
<<
nor
8x1
MUX
output
S2 S1 S0

6-) İnsturction_Memory : Bu modülde instructionlar tutulmaktadır. Gelen adress bilgisine göre output olarak 32bitlik instructionu atamaktadır.

7-) data_memory : Bu modül parametre olarak, alunun resultu yani adress, rt nin contenti yani write data, MemWrite sinyalleri alınmaktadır. Output olarak read ata vermekte ve çıkısında ki muxun sonucuna göre, instruction sonucu işlem görmektedir.

Before Registers.mem

00000000000000000000000000000000

00000000000000000000000000000001

00000000000000000000000000000010

00000000000000000000000000000011

00000000000000000000000000000100

00000000000000000000000000000101

00000000000000000000000000000110

00000000000000000000000000000111

00000000000000000000000000001111

00000000000000000000000011110000

00000000000000000000111100000000

00000000000000000001000000000000

00000000000000000000000000100000

00000000000000000000000000001001

00000000000000000000000000001100

00000000000000000000000001100000

00000000000000000000000000010000

00000000000000000000000000010001

00000000000000000000000000010010

00000000000000000000000000010011

00000000000000000000000000010100

00000000000000000000000000010101

00000000000000000000000000010110

00000000000000000000000000010111

00000000000000000000000000011000

00000000000000000000000000011001

00000000000000000000000000011010

00000000000000000000000000011011

00000000000000000000000000011100

00000000000000000000000000011101

00000000000000000000000000011110

00000000000000000000000000011111

MODELSİM SONUCLARI

opcode = 000000 | rs = 00100 | rt = 00101 | rd = 00110 | shampt = 00000 | func = 100000

| Result = 00000000000000000000000000001001

opcode = 000000 | rs = 00110 | rt = 00111 | rd = 01000 | shampt = 00000 | func = 100001

| Result = 00000000000000000000000000010000

opcode = 000000 | rs = 00111 | rt = 01000 | rd = 01001 | shampt = 00000 | func = 100010

| Result = 11111111111111111111111111110111

opcode = 000000 | rs = 01001 | rt = 01000 | rd = 01010 | shampt = 00000 | func = 100011

| Result = 11111111111111111111111111100111

opcode = 000000 | rs = 01001 | rt = 01000 | rd = 01011 | shampt = 00000 | func = 100100

| Result = 00000000000000000000000000010000

opcode = 000000 | rs = 01001 | rt = 01000 | rd = 01100 | shampt = 00000 | func = 100101

| Result = 11111111111111111111111111110111

opcode = 000000 | rs = 01001 | rt = 01000 | rd = 01101 | shampt = 00000 | func = 100111

| Result = 00000000000000000000000000001000

opcode = 000000 | rs = 00000 | rt = 01001 | rd = 01110 | shampt = 00001 | func = 000000

| Result = 11111111111111111111111111110111

opcode = 000000 | rs = 00000 | rt = 01001 | rd = 01111 | shampt = 00011 | func = 000010

| Result = 00111111111111111111111111111101

opcode = 000000 | rs = 01110 | rt = 01111 | rd = 10000 | shampt = 00000 | func = 101011

| Result = 10111111111111111111111111111010

opcode = 100011 | rs = 00101 | rt = 00100 | rd = 00000 | shampt = 00000 | func = 000000

| Result = 00000000000000000000000001010000

opcode = 101011 | rs = 00110 | rt = 00111 | rd = 00000 | shampt = 00000 | func = 000001

| Result = 00000000000000000000000000010000

opcode = 001001 | rs = 01001 | rt = 01000 | rd = 00000 | shampt = 00000 | func = 000100

| Result = 00000000000000000000000000000111

opcode = 001100 | rs = 01011 | rt = 01010 | rd = 00000 | shampt = 00000 | func = 000011

| Result = 00000000000000000000000000000000

opcode = 001101 | rs = 01100 | rt = 01110 | rd = 00000 | shampt = 00000 | func = 001000

| Result = 11111111111111111111111111101110

opcode = 000100 | rs = 01100 | rt = 01110 | rd = 11111 | shampt = 11111 | func = 111100

| Result = 11111111111111101111111111111011

ModelSım çalıştıktan sonra registers.mem

00000000000000000000000000000000

00000000000000000000000000000001

00000000000000000000000000000010

00000000000000000000000000000011

00000000000000000000000000000100

00000000000000000000000000010000

00000000000000000000000000001001

00000000000000000000000000000111

00000000000000000000000000010000

00000000000000000000000000000111

11111111111111111111111111100111

00000000000000000000000000000000

11111111111111111111111111101110

00000000000000000000000000001000

11111111111111111111111111110111

00111111111111111111111111111101

00000000000000000000000000000001

00000000000000000000000000010001

00000000000000000000000000010010

00000000000000000000000000010011

00000000000000000000000000010100

00000000000000000000000000010101

00000000000000000000000000010110

00000000000000000000000000010111

00000000000000000000000000011000

00000000000000000000000000011001

00000000000000000000000000011010

00000000000000000000000000011011

00000000000000000000000000011100

00000000000000000000000000011101

00000000000000000000000000011110

00000000000000000000000000011111

İnstruction_ memory.mem

00000000100001010011000000100000

00000000110001110100000000100001

00000000111010000100100000100010

00000001001010000101000000100011

00000001001010000101100000100100

00000001001010000110000000100101

00000001001010000110100000100111

00000000000010010111000001000000

00000000000010010111100011000010

00000001110011111100000000101011

10001100101001000000000000000000

10101100110001110000000000000001

00100101001010000000000000000100

00110001011010100000000000000011

00110101100011100000000000001000

00010001100011101111111111111100

Data_memory.mem

00000000000000000000000000000000

00000000000000000000000000000001

00000000000000000000000000000010

00000000000000000000000000000011

00000000000000000000000000000100

00000000000000000000000000000101

00000000000000000000000000000110

00000000000000000000000000000111

00000000000000000000000000001111

00000000000000000000000011110000

00000000000000000000111100000000

00000000000000000001000000000000

00000000000000000000000000100000

00000000000000000000000000001001

00000000000000000000000000001100

00000000000000000000000001100000

00000000000000000000000000010000

00000000000000000000000000010001

00000000000000000000000000010010

00000000000000000000000000010011

00000000000000000000000000010100

00000000000000000000000000010101

00000000000000000000000000010110

00000000000000000000000000010111

00000000000000000000000000011000

00000000000000000000000000011001

00000000000000000000000000011010

00000000000000000000000000011011

00000000000000000000000000011100

00000000000000000000000000011101

00000000000000000000000000011110

00000000000000000000000000011111