

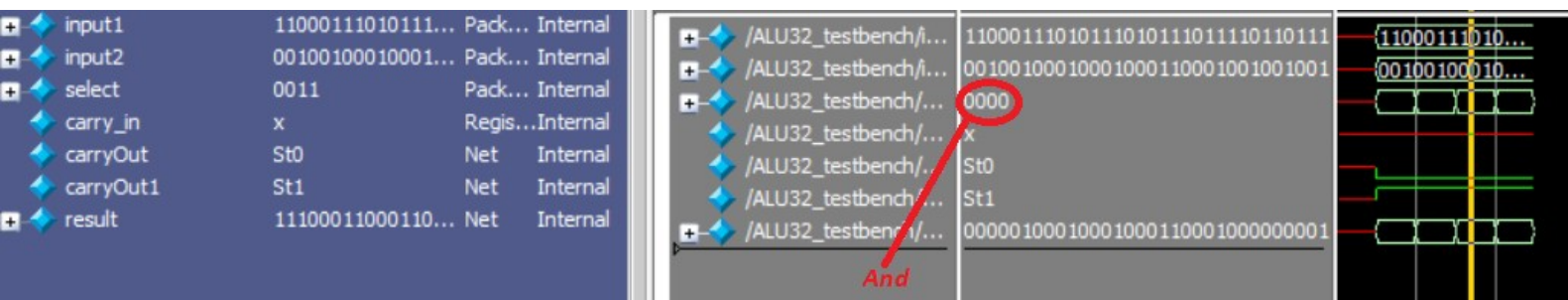
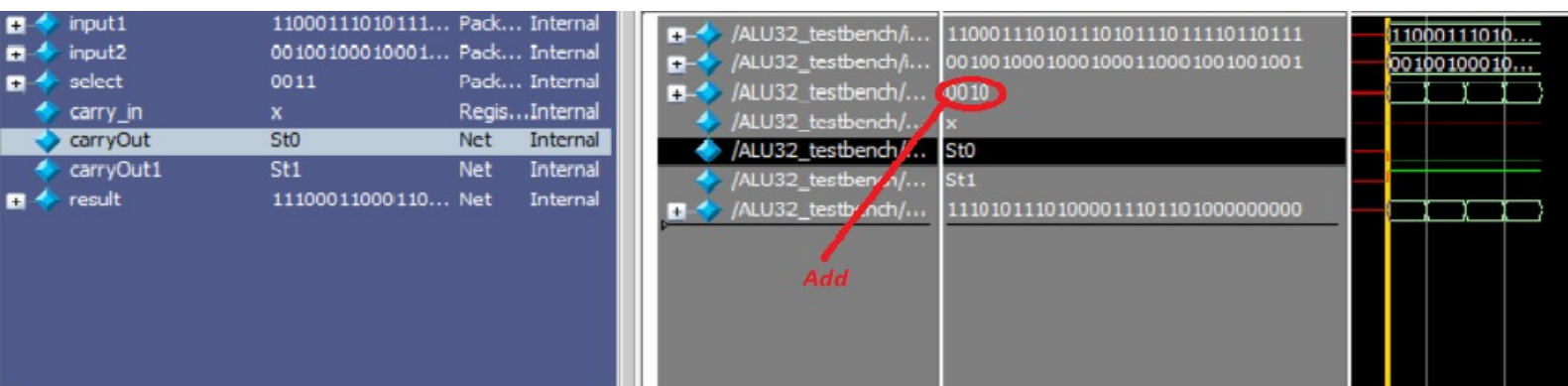
## TestBench ALU

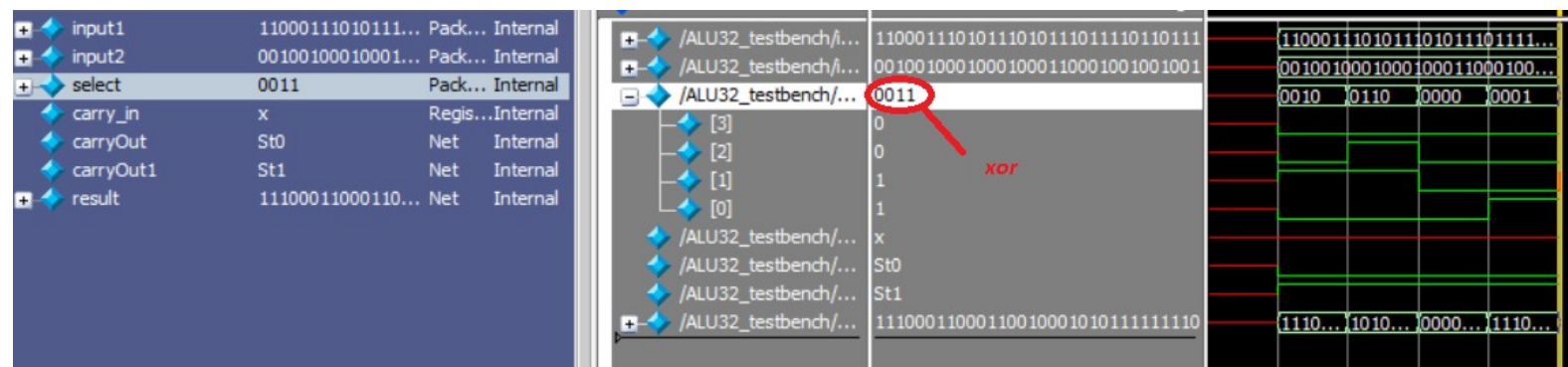
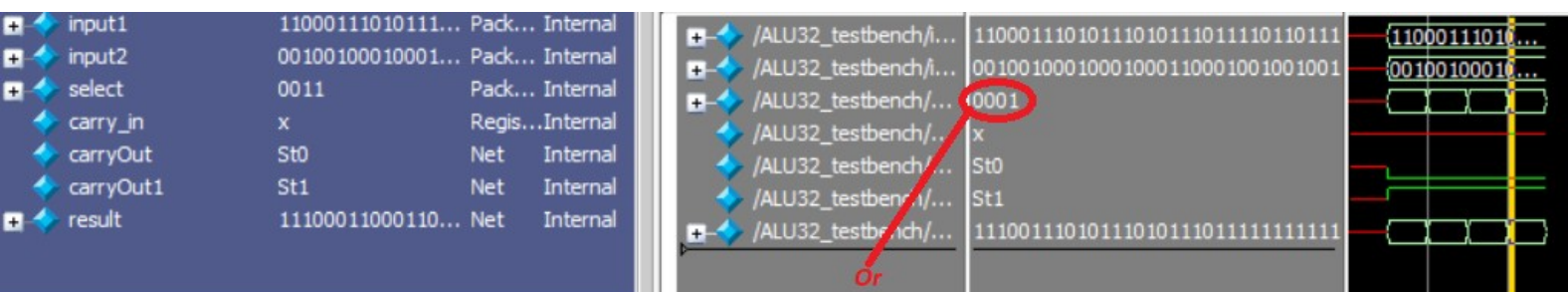
```

initial begin
#`DELAY;
input1 = 32'b11000111010111010111011110110111; input2 = 32'b00100100010001000100110001001001; select = 3'b010;
#`DELAY;
input1 = 32'b11000111010111010111011110110111; input2 = 32'b00100100010001000110001001001001; select = 3'b110;
#`DELAY;
input1 = 32'b11000111010111010111011110110111; input2 = 32'b00100100010001000110001001001001; select = 3'b000;
#`DELAY;
input1 = 32'b11000111010111010111011110110111; input2 = 32'b00100100010001000110001001001001; select = 3'b001;
#`DELAY;
input1 = 32'b11000111010111010111011110110111; input2 = 32'b00100100010001000110001001001001; select = 3'b011;
end

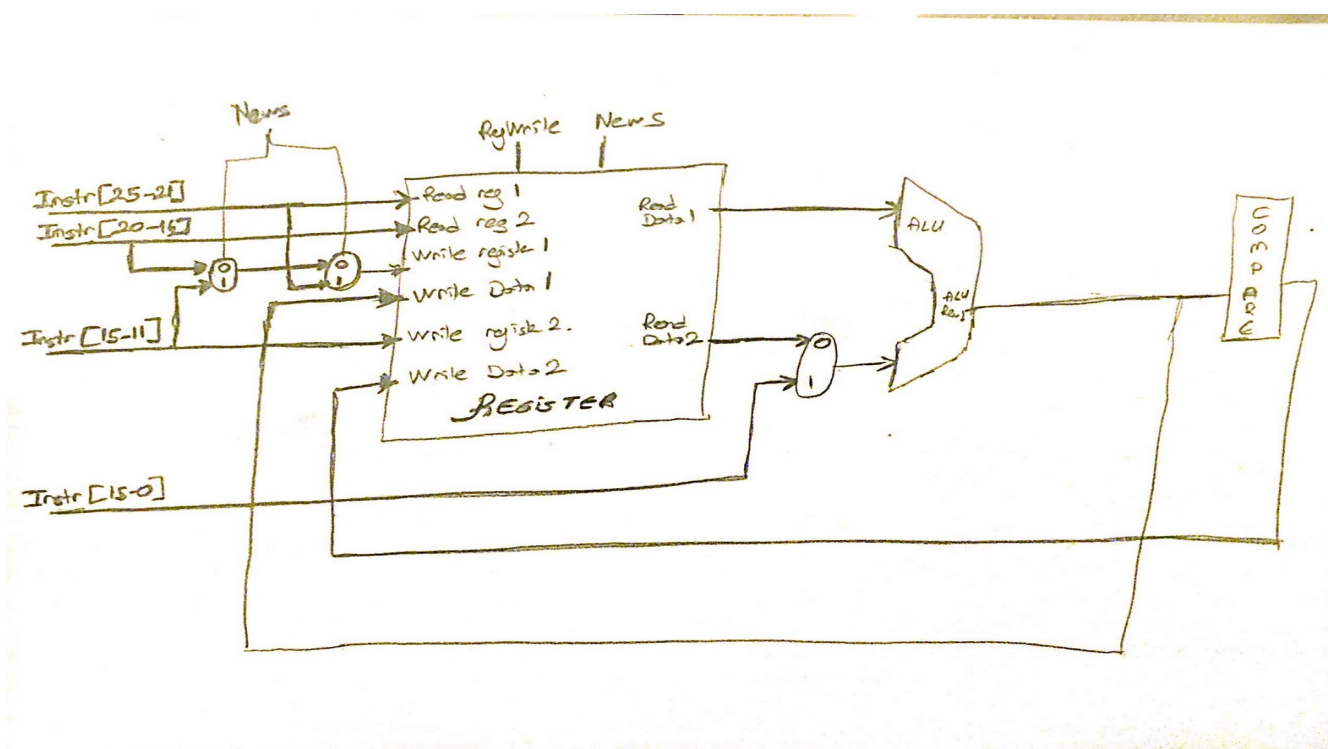
```

## Simulation





## Design datapath for New Instruction



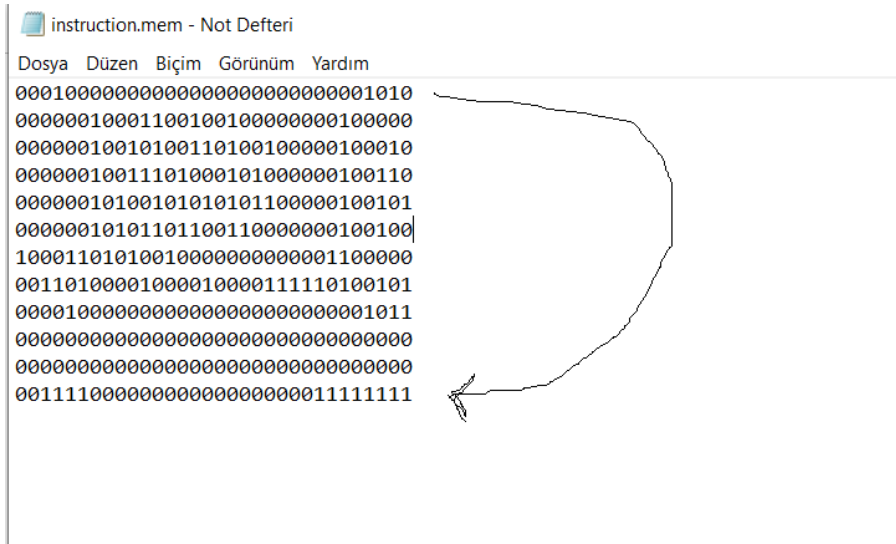
design register  
add new signal  
compare unit  
add one mux

## BEQ & BNE

000100 00000 00000 00000 00000 001010

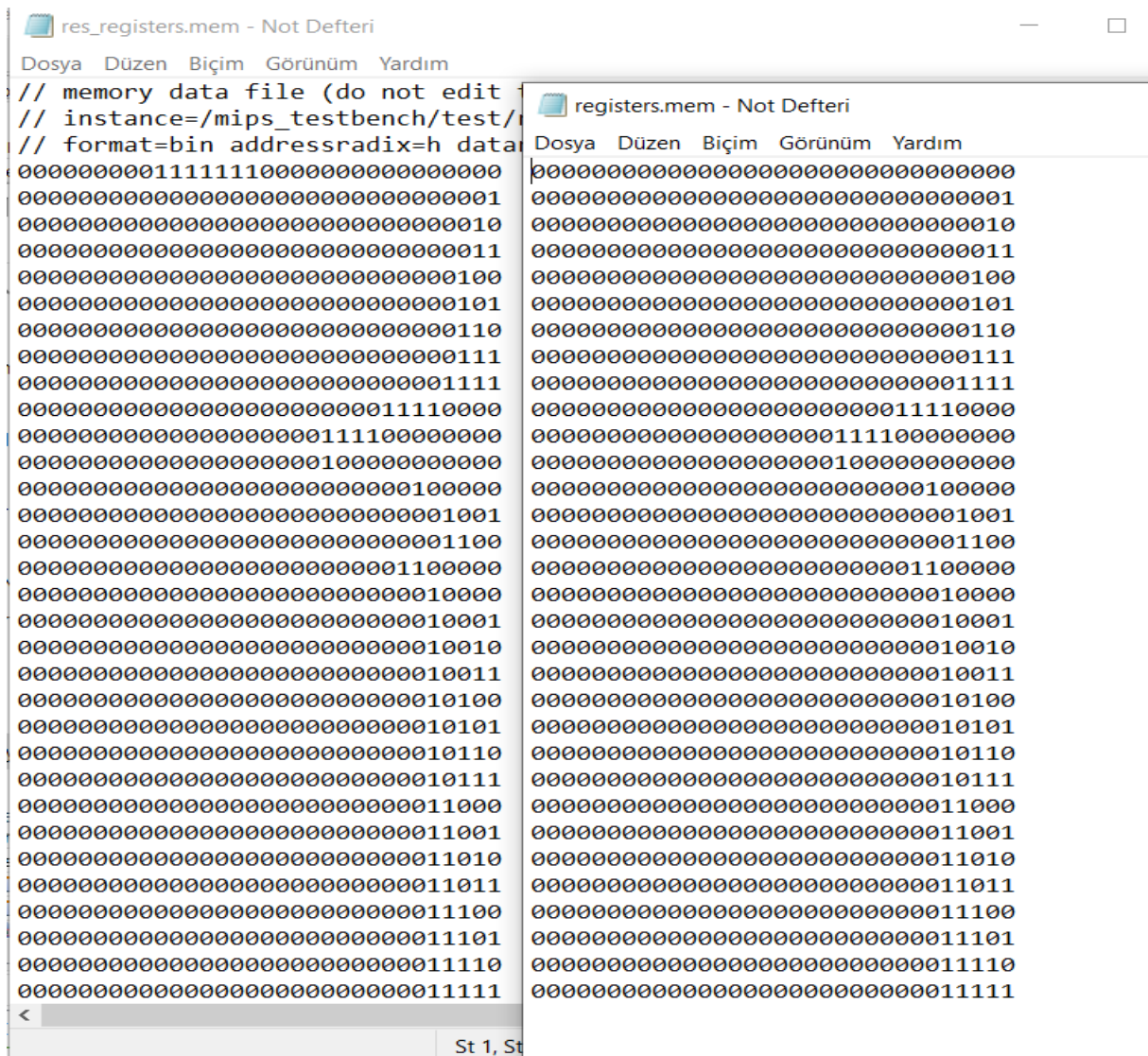
```
opcode    $rs=0    $rt=0    10
```

beq



## Result

beq in sonucu doğru olduğu için 10. registerdaki lui instrucionına gider.Aradaakilere atlar.





[illegible]

## Result

1.index 1. indexe eşit olmadığından atlama olamayacak ve instruction lar devam edecek.



res\_registers.mem - Not Defteri

[Dosya](#) [Düzen](#) [Biçim](#) [Görünüm](#) [Yardım](#)

[illegible]

registers.mem - Not Defteri

[Dosya](#) [Düzen](#) [Biçim](#) [Görünüm](#) [Yardım](#)

[illegible]

NOTE: Karışmasın diye burayı ayrı anlatmak istedim.

OP	RS	RT	RD	SHAM	FUNCT	
000100	00000	00000	00000	00000	001010	<i>beq \$0,\$0,10</i>
000000	10001	10010	01000	00000	100000	<i>addn \$8,\$17,\$18</i>
000000	10010	10011	01001	00000	100010	<i>subn \$9,\$18,\$19</i>
000000	10011	10100	01010	00000	100110	<i>xorn \$10,\$19,\$20</i>
000000	10100	10101	01011	00000	100101	<i>orn \$11,\$20,\$21</i>
000000	10101	10110	01100	00000	100100	<i>andn \$12,\$21,\$22</i>
100011	01010	01000	0000000000	1100000		<i>lw \$8,4(\$10)</i>
001101	00001	00001	0000111110	100101		<i>ori \$1, \$1, 0000111110100101</i>
000010	000000000000000000000000	01010				<i>jr \$10</i>
000000	00000	00000	00000	00000	000000	<i>boş</i>
000000	00000	00000	00000	00000	000000	<i>boş</i>
001111	00000	00000	0000000001	1111111		<i>lui \$0, 255</i>
000010	000000000000000000000000	01				<i>j \$1</i>

instruction.mem - Not Defteri

Dosya Düzen Biçim Görünüm Yardım

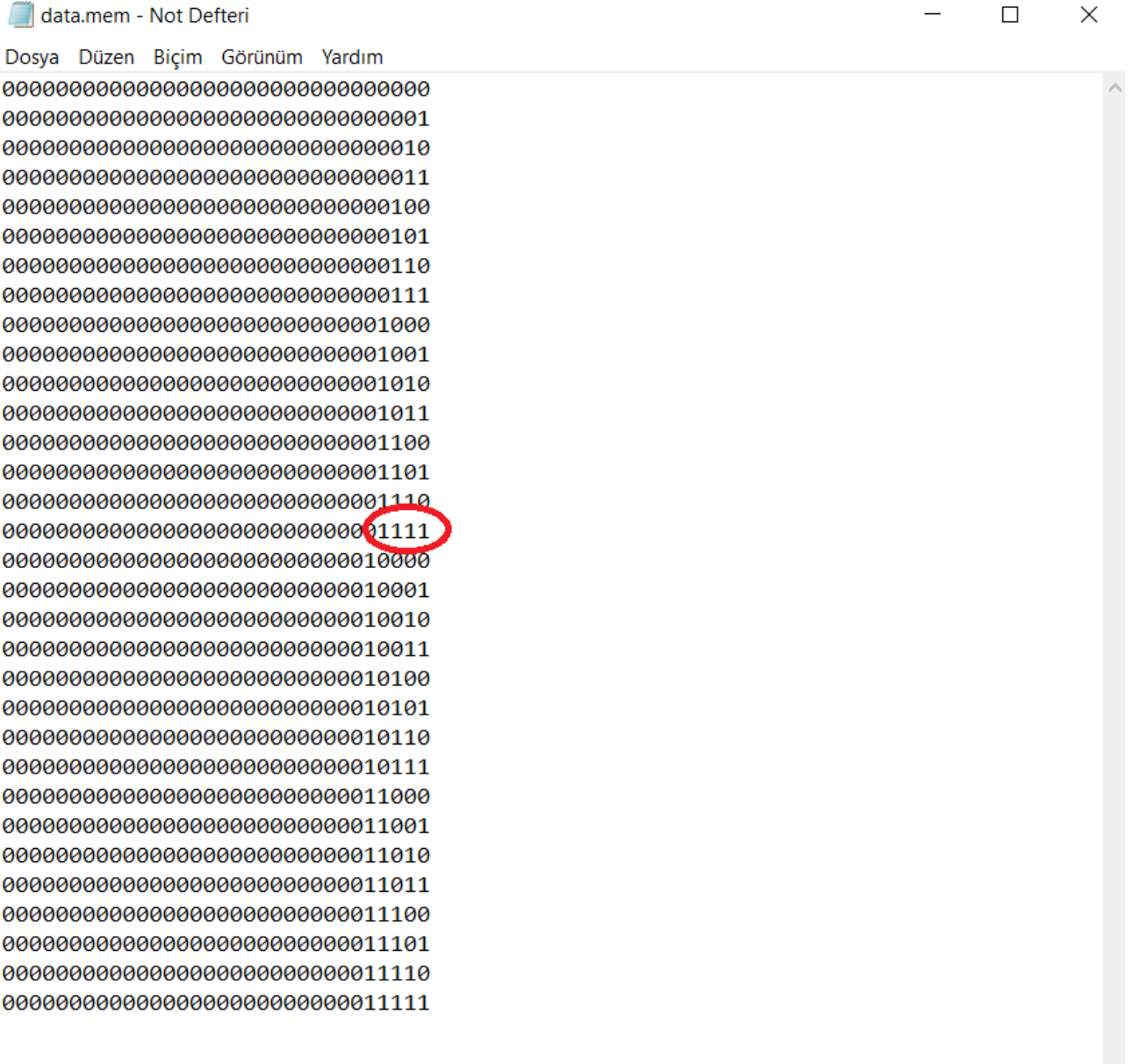
```

0001000000000000000000000000000001010
00000010001100100100000000100000
00000010010100110100100000100010
00000010011101000101000000100110
00000010100101010101100000100101
00000010101101100110000000100100
10001101010010000000000001100000
00110100001000010000111110100101
00000001010000000000000000001000
00000000000000000000000000000000
00000000000000000000000000000000
00111100000000000000000011111111
00001000000000000000000000000001

```

\*İlk instruction *beq \$0,\$0,10* 10.registara atladı *lui \$0, 255* instruction 0 yazdı daha sonra *j \$1 addn \$8,\$17,\$18* bu instruction a geri döndü ve *subn \$9,\$18,\$19, xorn \$10,\$19,\$20,orn \$11,\$20,\$21,andn \$12,\$21,\$22* ile devam eder.\$8,\$9,\$10,\$11,\$12 0'dan yükse olduğu için 0000000000000000000000000000011 yazılır.Diğer sonuçlar *\$18,\$19,\$20,\$21,\$22* yazılır.New instruction lar da böyle yazılır.

**lw \$8,4(\$10)** gelir sonra data. Mem dosyasında alınır ve **\$8 e yazar**



```
data.mem - Not Defteri
Dosya Düzen Biçim Görünüm Yardım
00000000000000000000000000000000
00000000000000000000000000000001
00000000000000000000000000000010
00000000000000000000000000000011
00000000000000000000000000000100
00000000000000000000000000000101
00000000000000000000000000000110
00000000000000000000000000000111
000000000000000000000000000001000
000000000000000000000000000001001
000000000000000000000000000001010
000000000000000000000000000001011
000000000000000000000000000001100
000000000000000000000000000001101
000000000000000000000000000001110
000000000000000000000000000001111
0000000000000000000000000000010000
0000000000000000000000000000010001
0000000000000000000000000000010010
0000000000000000000000000000010011
0000000000000000000000000000010100
0000000000000000000000000000010101
0000000000000000000000000000010110
0000000000000000000000000000010111
0000000000000000000000000000011000
0000000000000000000000000000011001
0000000000000000000000000000011010
0000000000000000000000000000011011
0000000000000000000000000000011100
0000000000000000000000000000011101
0000000000000000000000000000011110
0000000000000000000000000000011111
```

**ori \$1, \$1, 0000111110100101** instuction kendisyle or ladım  
**jr \$10 ile lui atlarım döngü tamamlanmış olur.**

# OUTPUT

res\_registers.mem - Not Defteri

Dosya Düzen Biçim Görünüm Yardım

```
// memory data file (do not edit
// instance=/mips_testbench/test/
// format=bin addressradix=h data
00000000011111110000000000000000
000000000000000000000000011110100101
0000000000000000000000000000000010
0000000000000000000000000000000011
00000000000000000000000000000000100
00000000000000000000000000000000101
00000000000000000000000000000000110
00000000000000000000000000000000111
000000000000000000000000000000001111
0000000000000000000000000000000011
0000000000000000000000000000000011
0000000000000000000000000000000011
0000000000000000000000000000000011
0000000000000000000000000000000011
000000000000000000000000000000001001
000000000000000000000000000000001100
00000000000000000000000000000000110000
0000000000000000000000000000000010000
00000000000000000000000000000000100011
00000000000000000000000000000000100001
0000000000000000000000000000000010011
0000000000000000000000000000000010101
0000000000000000000000000000000010100
0000000000000000000000000000000010110
0000000000000000000000000000000010111
0000000000000000000000000000000011000
0000000000000000000000000000000011001
0000000000000000000000000000000011010
0000000000000000000000000000000011011
0000000000000000000000000000000011100
0000000000000000000000000000000011101
0000000000000000000000000000000011110
0000000000000000000000000000000011111
```

registers.mem - Not Defteri

Dosya Düzen Biçim Görünüm Yardım

```
00000000000000000000000000000000
00000000000000000000000000000001
00000000000000000000000000000010
00000000000000000000000000000011
000000000000000000000000000000100
000000000000000000000000000000101
000000000000000000000000000000110
000000000000000000000000000000111
0000000000000000000000000000001111
00000000000000000000000000000011110000
000000000000000000000000000000111100000000
00000000000000000000000000000010000000000
000000000000000000000000000000100000
0000000000000000000000000000001001
0000000000000000000000000000001100
000000000000000000000000000000110000
00000000000000000000000000000010000
00000000000000000000000000000010001
00000000000000000000000000000010010
00000000000000000000000000000010011
00000000000000000000000000000010100
00000000000000000000000000000010101
00000000000000000000000000000010110
00000000000000000000000000000010111
00000000000000000000000000000011000
00000000000000000000000000000011001
00000000000000000000000000000011010
00000000000000000000000000000011011
00000000000000000000000000000011100
00000000000000000000000000000011101
00000000000000000000000000000011110
00000000000000000000000000000011111
```

## EKSİKLER

- SW yi kullanmadım ama çalışıyor.
- Testbench de sadece register ve sonuc register var.Memory ve instruction dosya okuması hata verdi o yüzden moduller içinde readmem ya da writememh kullandım.
- mux32bit5\_1 behavior kullandım değiştirmeyte fırsat olmadı.