

**Gebze Technical University**

**Computer Engineering**

**CSE 331**

**ASSIGNMENT 4 REPORT**

**FERHAT ÇELİK**

**151044014**

Course Assistant:Fatma Nur Esirci



R

op rs rt rd shamt func  
6 5 5 5 5 6

xor, slt, jr, add, sub, and  
or, sra, srl, sll, sltu

opcode  
000000

func  
dest:rt  
add:100000

and	000000
or	000001
xor	000010
add	000011
sub	000100
slt	000101
sra	000110
srl	000111
sll	001000
sltu	001001
jr	001010

I

op rs rt add/im  
6 5 5 16

xori, sltiu, lw, lh, lb  
sw, sb, beq, bne, addi  
addiu, andi, ori, slti  
lui

andi	000001
ori	000010
xori	000011
addi	000100
addiu	000101
slti	000110
sltiu	000111
lw	100011
lh	100001
lb	100000
lui	001111
sw	101011
sb	101000
beq	001110
bne	010000

J

op target address  
6 26

j, jal,

j 010001

jal 010010

frht 11111 → skip next inst.

ALU

and	000
or	001
add	010
xor	011 → new
shifter	100 → new
shiftl	101 → new
sub	110
slt	111

Marine Propulsion Systems

celik 111110 → rs (signim) isc rs xor imm  
degilse 0

Transmissions - Hybrid Ready Transmissions - Control Systems - Joystick Maneuvering Systems

Fixed Pitch Propellers - SPP - Sail Drive Systems - Surface Drive Systems - FPP and CPP Tunnel Thrusters

Well Mounted / Deck Mounted / Stern Mounted / Retractable / Contra Rotation / Shallow Draught Azimuth Thrusters

## Eklenen yeni 2 instruction:

frht (j type) → 1 instruction ileri atlar.Yani bir sonraki instruction'ı atlayarak sonrakine geçer.

Celik (i type) → eğer rs signimm değerinden küçükse rs ile signimm değerini xorlar.Değilse 0 döner.

## Yeni eklenen modüller

İkinci ödevde yaptığım 32 bitlik alu modülünü kullandım.Ancak xor ve shift modüllerini de ekledim.

**Alucontrol** → r type ise function code'a bakarak i type ise opcode'a göre aluopcode üretiyor.

### I Type

```
and (i_type_op[0],~and_,~add_,~sub_);  
and (i_type_op[1],~and_,~or_);  
and (i_type_op[2],~and_,~or_,~add_,~xor_);
```

### R Type

```
and (r_type_op[0],~r_and,~r_add,~r_shiftr,~r_sub);  
and (r_type_op[1],~r_and,~r_or,~r_shiftr,~r_shiftl);  
and (r_type_op[2],~r_and,~r_or,~r_add,~r_xor);
```

32 bit 2:1 mux ile instruction tipine göre aluop belirlenir.

**instruction\_memory** → 32 bit bir memory.Program counter 'a göre sırayla instructionları gönderir.

**jump\_address** → jump instructionı geldiği zaman 26 bitlik jump adresini program counter ile birleştirip jump edilecek adresi oluşturur.

**next\_pc** → program counter'ı clock atımlarına göre arttırır.Jump veya brach instructionları geldiğinde pc 'ı hanginin geldiğine göre değiştirir.



## Test Sonuçları

## Before

## After