Project Report

Department Electrical and computer

Engineering

Summer 2022

CSE-332

Computer Organization and Architecture

NAME: Md. Mohibur Rahman Nabil

ID: 2021068642

SECTION: 03

Submitted To: **Tanjila Farah (Tnf)**

INTRODUCTION

In this project I have build a single cycle MIPS architecture with a 14-bit isa which can perform 11 operations (Arithmetic, logical, data transferring, branching and jumping).

Operands and Types of Operands

There are 3 operands and I am mentioning them as s, t and d.

Types of Operands:

1. Register Based
2. Memory Based

Operations

There is total 11 operations in total in my architecture.

We have 3 types of instruction:

1. R Type
2. I Type
3. J Type

Operations Types are:

1. Arithmetic
2. Logical
3. Bruch(conditional)
4. Jump(unconditional)
5. Data Transfer

INSTRUCTION TABLE

opcode	instruction	Type	Operation Type	alu op	msb	lsb	b in	cin	wc	mem2reg	alu src	reg dest	str	ld	branch	memEn	jump
00000	add	R type	Arithmatic	1 1	1	1	0	0	1	0	0	0	0	0	0	0	0
00001	sll	R type	Logical	0 0	0	0	0	0	1	0	0	0	0	0	0	0	0
00010	sub	R type	Arithmatic	1 1	1	1	1	1	1	0	0	0	0	0	0	0	0
00011	sw	I type	Data Transfer	1 1	1	1	0	0	0	0	1	0	1	0	0	1	0
00100	beq	I type	Branch(Conditional)	x x	x	x	0	0	0	0	0	0	0	0	1	0	0
00101	nop			x x	x	x	0	0	0	0	0	0	0	0	0	0	0
00110	and	R type	Logical	0 1	0	1	0	0	1	0	0	0	0	0	0	0	0
00111	slt	R type	Logical	1 0	1	0	0	0	1	0	0	0	0	0	0	0	0
01000	addi	I type	Arithmatic	1 1	1	1	0	0	1	0	1	1	0	0	0	0	0
01001	lw	I type	Data Transfer	1 1	1	1	0	0	1	1	1	1	0	1	0	1	0
01010	jmp	J type	Jump(Unconditional)	x x	x	x	0	0	0	0	0	0	0	0	0	0	1

ISA Design

14 bit isa

R type			
op	rs	rt	rd
5bit	3bit	3bit	3bit

I type			
op	rs	rt	immidiate
5bit	3bit	3bit	3bit

j type	
op	target
5bit	9 bit

Description of instruction

1. **add:** adds two number from register and store the result into a register
   r3 = r1+ r2
   Code: add r3 r1 r2

2. **sll:** Left shifts bits of a value
   r3 <<31
   Code: sll r3 r3 r1

3. **sub:** subtracts two number from register and store the result into a register

   r3 = r1 - r2
   Code: sub r3 r1 r2

4. **sw: stores a value from register to ram.**

Ram address 1 = r3
Code: sw r3 r0 1

5. **beq :** checks if the values are same. If same then goes to address mentioned in the offset.

If(r1 == r2)
Goto: 10
Else:
Next line

Code: beq r1 r2 10

6. **nop:** do nothing

7. **and:** and two register value and store onto another register.

R3 = R1 and r2
Code: and r3 r1 r2

8. **slt:** check if r1 is less then r2. If yes then r3 = 1 else r3 =0.

R3 = r1<r2
Code: slt r3 r1 r2

9. **addi:** adds register and immediate value and stores to another register.

R3 = r1 + 5
Code: addi r3 r1 5

10. lw: loads a value from ram and store it to the register.

R3 = value of ram address 1
Code: lw r3 r0 1

11. jmp: jumps to a specific address of code.

goto 3
Code: j 3

Components of the MIPS Architecture

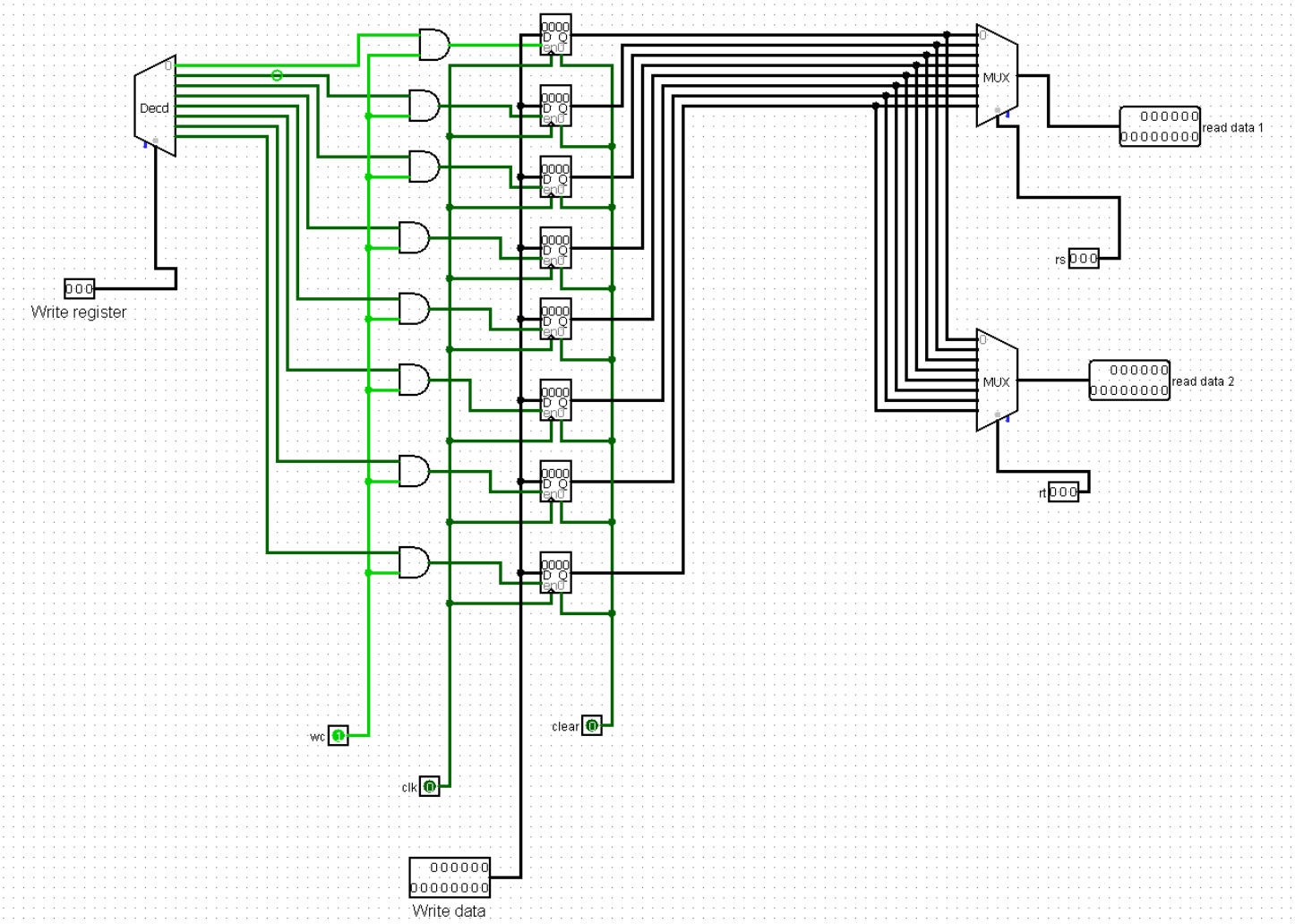

Figure: 8-bit register file

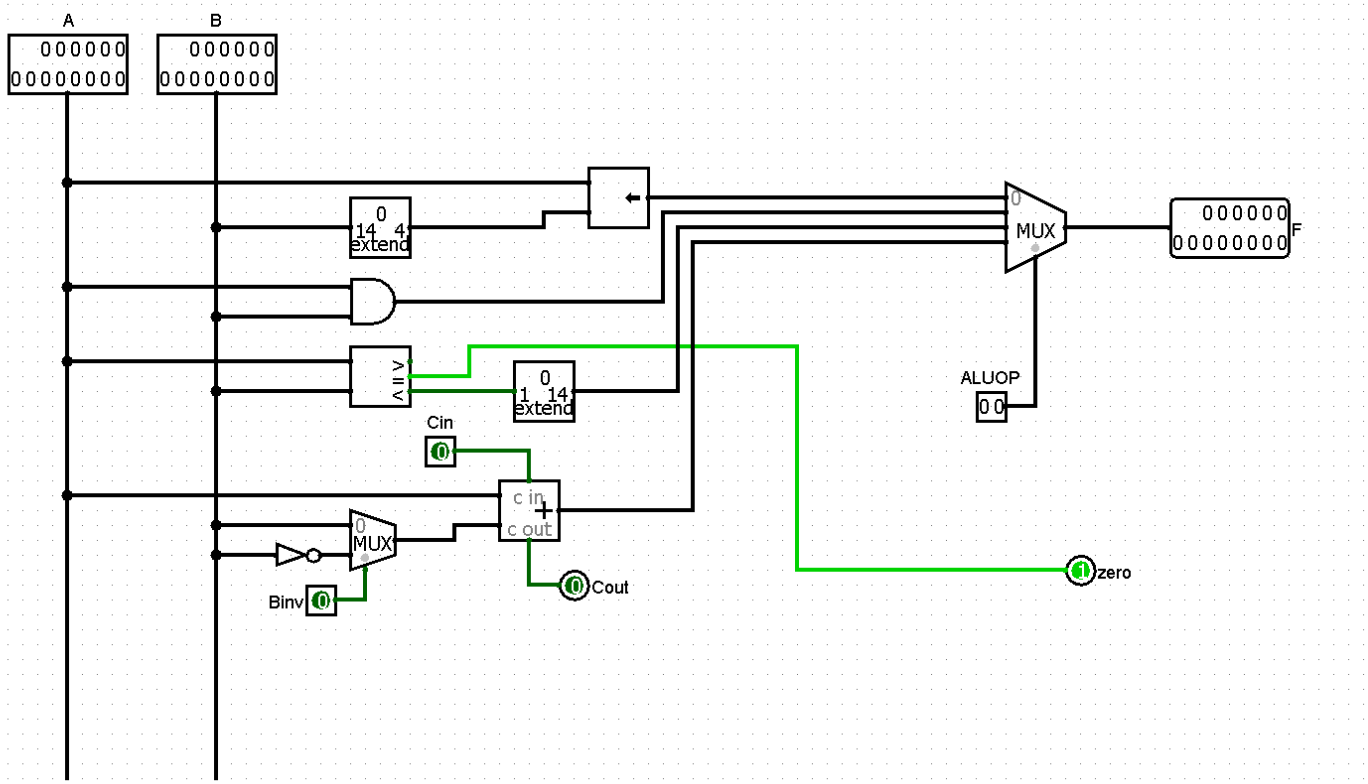

Figure: 8-bit ALU

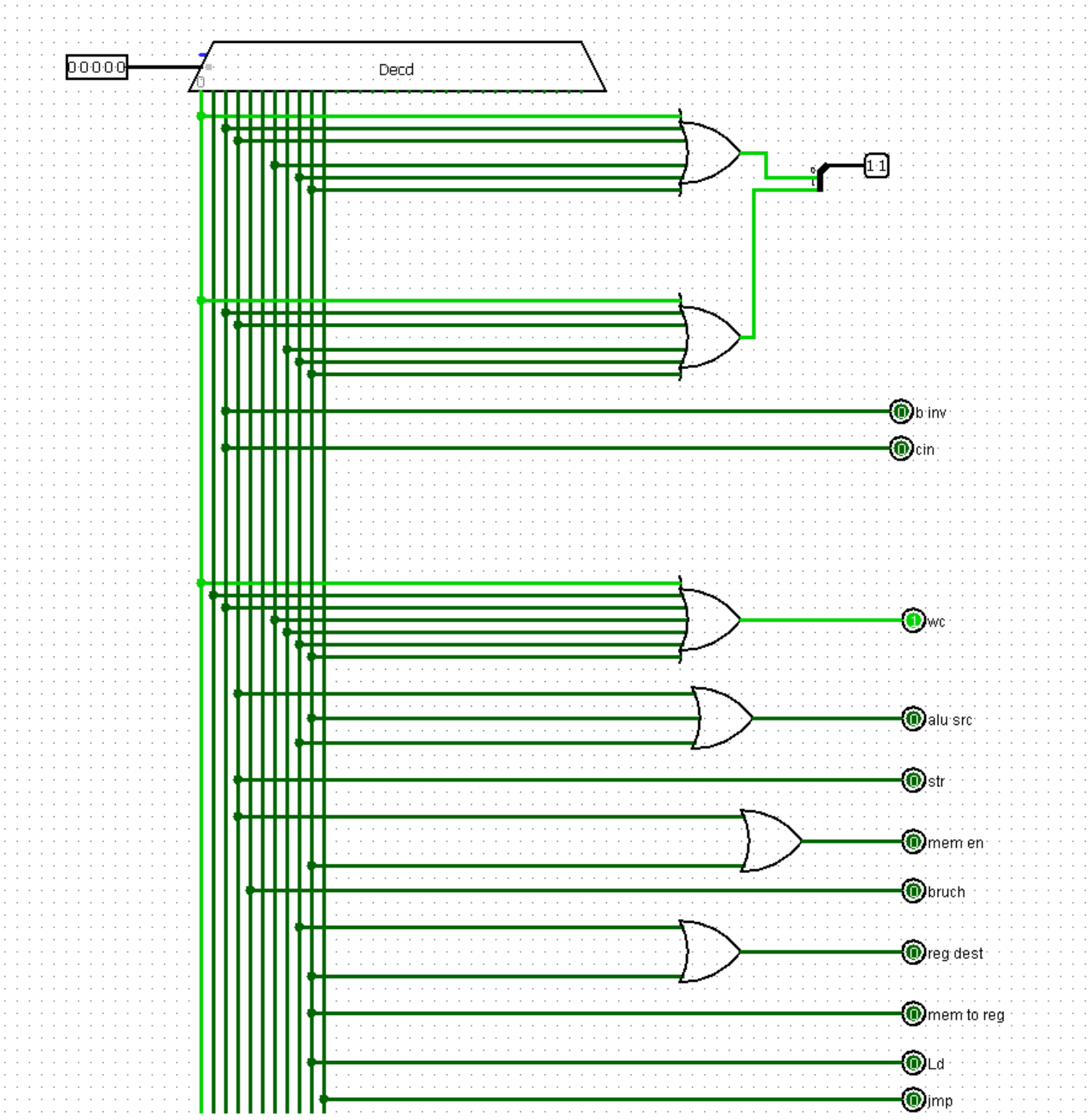

Figure: control unit

Single cycle MIPS

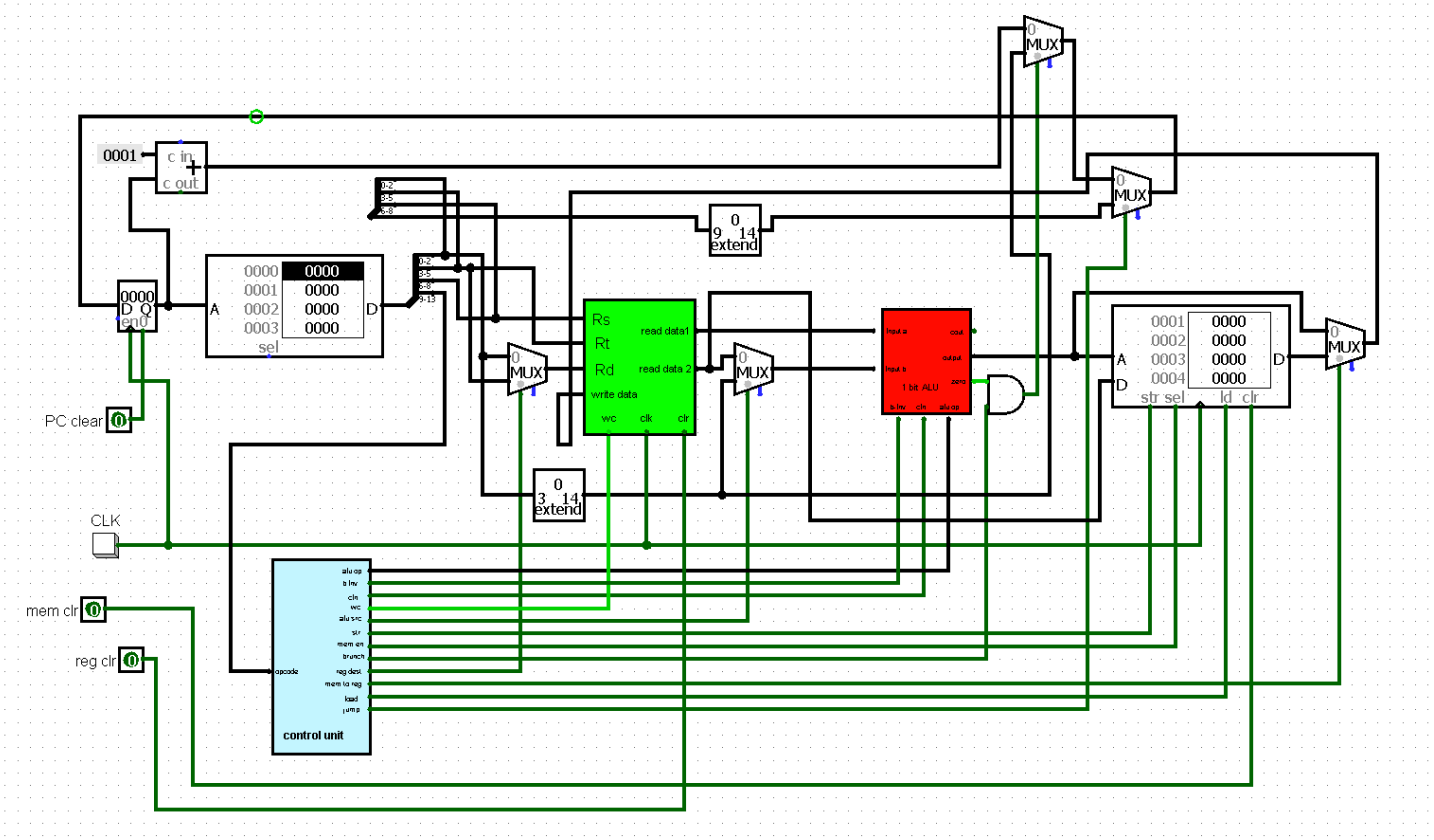

Figure: single cycle MIPS

The END

------o------