Assignment on 4-bit MIPS Design, Simulation, and Implementation

COURSE NO: 306

COMPUTER ARCHITECTURE SESSIONAL

GROUP NO: 2

SUBSECTION: B2

GROUP MEMBERS:

1. ID: 1805092 Name: Ishika Tarin
2. ID: 1805104 Name: Swarnali Saha
3. ID: 1805110 Name: Anika Monir
4. ID:1805114 Name: Afroza Parvin Disa
5. ID:1805116 Name: Sanjida Islam Era

LEVEL / TERM : 3-1

Department of Computer Science and Engineering

Bangladesh University of Engineering and Technology (BUET)

Date of submission: 19 August 2022

1.Introduction

The main task of this experiment was to design a 4bit microprocessor, simulate it in software and implement it in hardware according to the MIPS instruction set. Following the design specification, 8bits address bus, 4bits data bus and 4 bits ALU were used. The temporary registers: $zero, $t0, $t1, $t2, $t3, $t4 were designed. The control unit was microprogrammed, and the control signals associated with the operations were stored in a special memory unit as control words. Instruction memory and data memory were considered. All instructions needed 1 clock cycle to be executed. The main components of the processor are as follows: ALU, register file, instruction memory, data memory, and a control unit.

2.Instruction Set Description

Instruction ID	Instruction Type	Instruction
A	Arithmetic	add
B	Arithmetic	addi
C	Arithmetic	sub
D	Arithmetic	subi
E	Logic	and
F	Logic	andi
G	Logic	or
H	Logic	ori
I	Logic	sll
J	Logic	srl
K	Logic	nor
L	Memory	lw
M	Memory	sw
N	Control	beq
O	Control	bneq
P	Control	j

Fig: Given Instruction Set Description

3.MIPS Instruction Format

All MIPS Instructions were 16-bits long with the following four formats:

1. R-type

Opcode	Src Reg1	Src Reg2	Dst Reg
4-bits	4-bits	4 -bits	4-bits

2. S-type

Opcode	Src Reg1	Dst Reg	Shamt
4-bits	4-bits	4 -bits	4-bits

3. I-type

Opcode	Src Reg1	Src Reg2/Dst Reg	Addr./Immdt.
4-bits	4-bits	4 -bits	4-bits

4. J-type

Opcode	Target Jump address	0
4-bits	8-bits	4-bits

4. Assigned Instruction Set

ID Serial	Instruction
K	nor
H	ori
C	sub
I	sll
L	lw
N	beq
F	andi
D	subi
M	sw
P	j
O	bneq
E	and
G	or
B	addi
J	srl
A	add

5. Control Unit Input and Outputs

	Aluop	ALU src	Bneq	Beq	Memto Reg	Memto Write	Memread	RegWrite	Rdst	S	J
K	100	0	0	0	0	0	0	1	1	0	0
H	010	1	0	0	0	0	0	1	0	0	0
C	001	0	0	0	0	0	0	1	1	0	0
I	101	1	0	0	0	0	0	1	0	1	0
L	000	1	0	0	1	0	1	1	0	0	0
N	001	0	0	1	0	0	0	0	0	0	0
F	011	1	0	0	0	0	0	1	0	0	0
D	001	1	0	0	0	0	0	1	0	0	0
M	000	1	0	0	0	1	0	0	0	0	0
P	000	0	0	0	0	0	0	0	0	0	0
O	001	0	1	0	0	0	0	0	0	0	0
E	011	0	0	0	0	0	0	1	1	0	0
G	010	0	0	0	0	0	0	1	1	0	0
B	000	1	0	0	0	0	0	1	0	0	0
J	110	1	0	0	0	0	0	1	0	1	0
A	000	0	0	0	0	0	0	1	1	0	0

6. Opcode For ALU

ALUop	Instruction
000	ADDITION
001	SUBTRACTION
010	OR
011	AND
100	NOR
101	SLL

110	SRL

7.Circuit Diagrams:

7.1: Complete circuit with all components

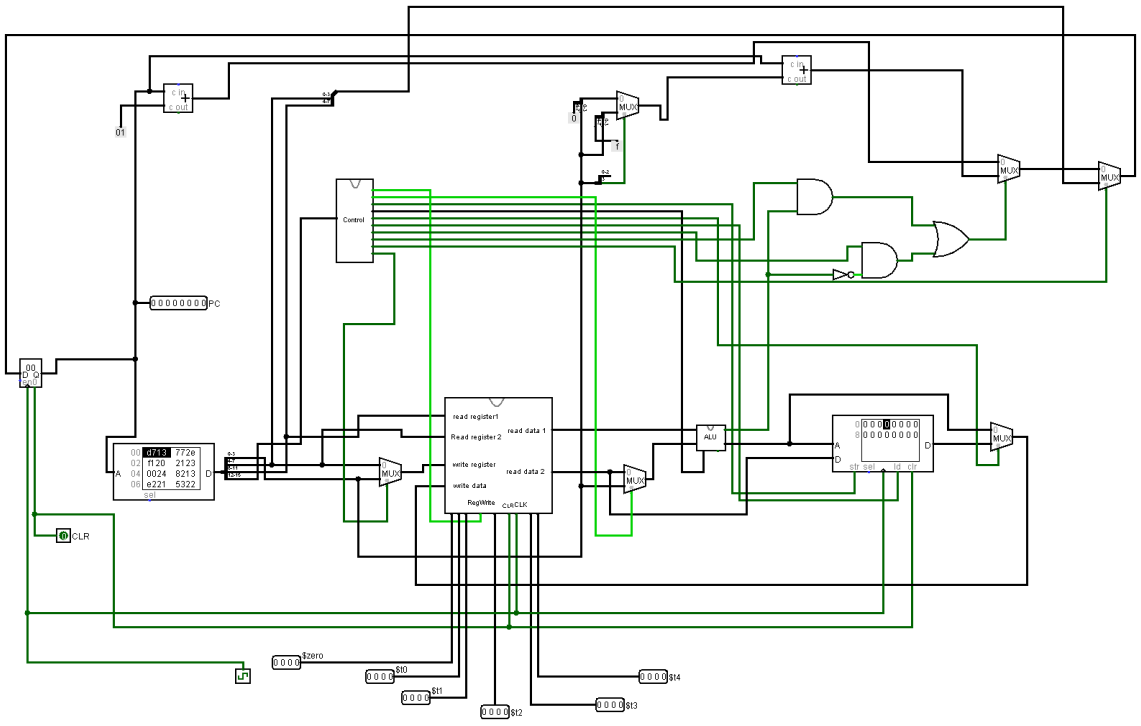

Fig: Main circuit of 4 Bits MIPS

7.2 Arithmetic logical unit

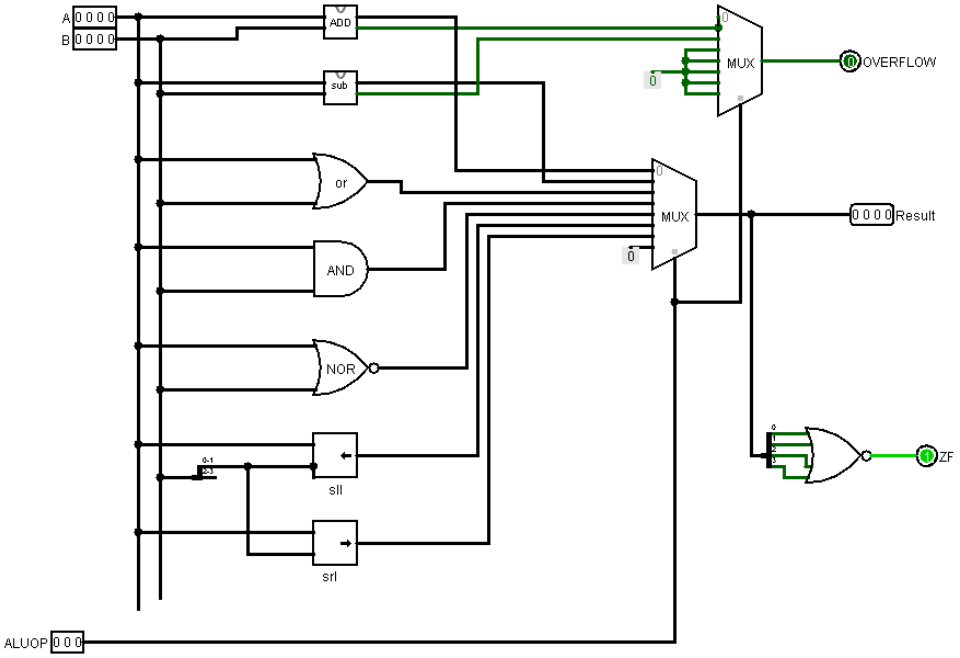

Fig: 4 Bit ALU

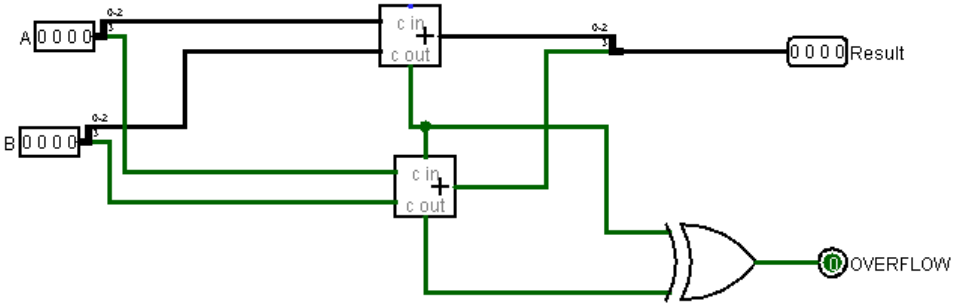

Fig: Adder Circuit

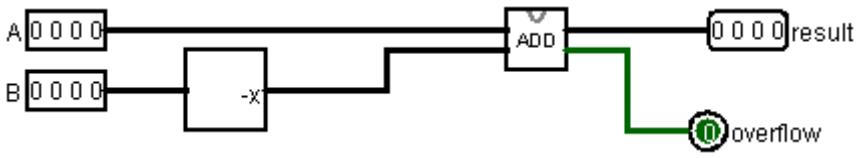

Fig: Subtractor

7.3 Registers

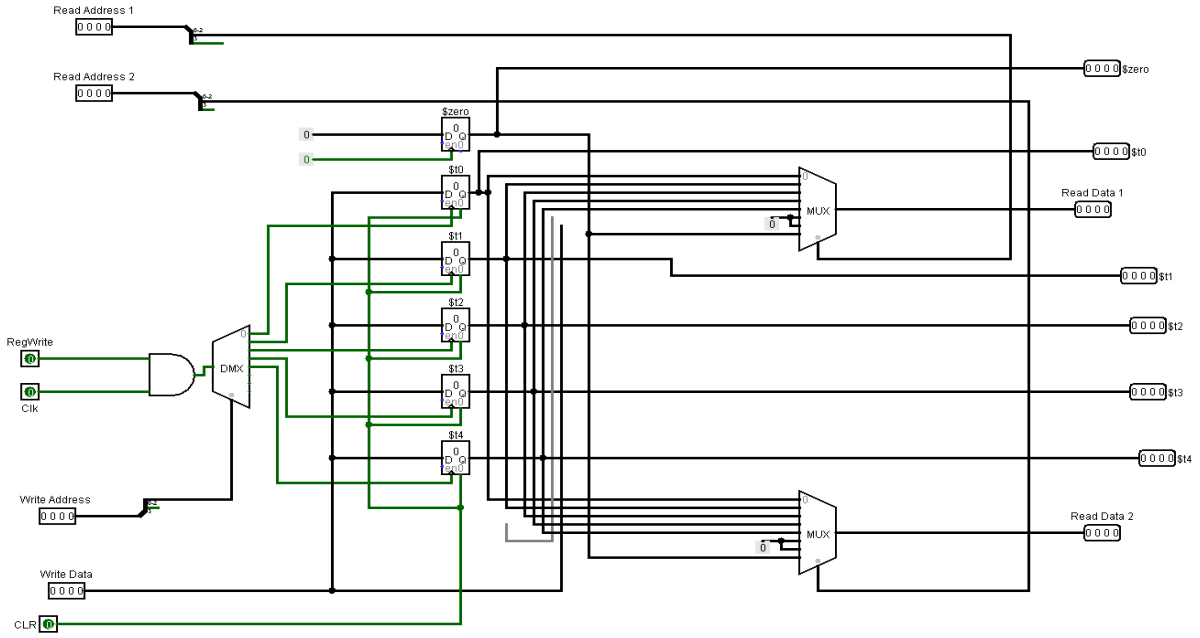

Fig: Registers

7.4 Control unit

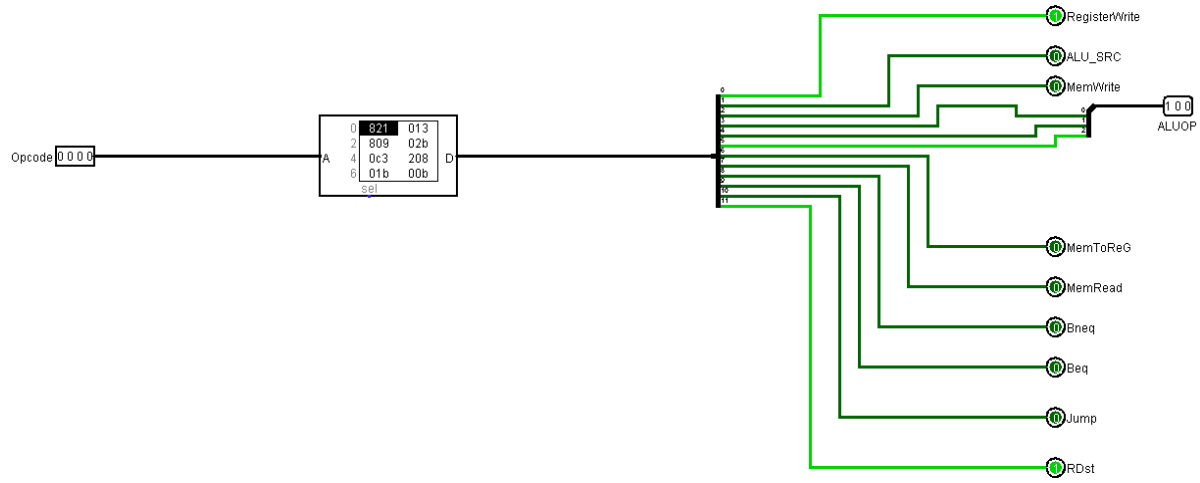

Fig: Control Unit

8.How to write and execute a program on this machine:

A program is written that takes the MIPS code and converts it to a file .The file contains the hexadecimal value of the instruction machine code. Then we need to upload it to the instruction ROM. Then the registers get the values from the ROM. The program will start as soon as the processor gets the clock pulse.

9.1 ICs and their counts(In software):

IC	COUNT
8 Bit Logical left shifter	1
8 bit Logical right shifter	1
74173 (4 bit D type register)	6
Register 8 bit	2
Adder 4 bit	4
Adder 8 bit	2
2 to 1 Mux 74157	5
8 to 1 Mux 74151	4
Negator	4
Demux	1
7432(OR)	2
7408(AND)	4
NOR	2
XOR	2

9.2 ICs and their counts(In hardware):

IC	COUNT
Adder 8 bit	2
2 to 1 Mux 74157	9
ATMEGA32	5
7432(OR)	1
7408(AND)	2
NOT	1

10.Discussion

The Processor takes a binary number (Instruction) and the circuit reads/stores the register values and memory values. The hardware implementation used Atmega32 ICs for abstraction; ROMs (used in Instruction Memory, Control Unit), Register File, ALU, RAM (used in Data Memory). A number of basic gates (AND, OR, NOT, NOR), universal gates (XOR, XNOR), and some other necessary gates (MUX, Shifter, Bit Finder, Adder, Subtractor, Bit Extender), RAM, ROM and register were also used. We checked various inputs and test cases and tried to simplify the circuit. Thus our simulation was done.