Project Report – CSE332

Topic: 16-bit Processor

NORTH SOUTH UNIVERSITY

Center of Excellence in Higher Education

The first private university in Bangladesh

Name: Md.Tahrim Faroque Tushar

ID: 1621148642

Name: Abdur Rab Dhruba

ID: 1712441642

Name: Kazi Nabiul Alam

ID: 1711217642

Name: Alisa Hossain

ID: 1522014042

Sec: 10

Date: 18-04-2019

Submitted To:

Prof. Dr. Khaleda Ali (KDA1)

Department of Electrical and Computer Engineering

1. Number of Operands:

R-format->3 operands

I-format->2 operands

2. Types Of operand:

Register-Register based or, Load-Store Based

3. Operations:

Opcode is 4 bits and functionality is 3 bits, so total 7 bits are reserved for different operations. By using those bits, we can do 128 operations. Because in binary, for example if we have 2 bits, we can $2^2 = 4$ operations. We can assign them or, select them by using multiplexers (MUX) such as 00, 01, 10, 11. So, with 7 bits we can do $2^7 = 128$ operations.

4. Operations Type:

Arithmetic Operation:

1) Addition
2) Subtraction
3) Multiplication
4) Division
5) Immediate Addition

Logical Operation:

1) AND

2) OR

3) NOT

Branch Type Operations:

1) Branch on equal

2) Branch on not equal

3) Branch on greater than

4) Branch on greater than or equal

5) Branch on less than

6) Branch on less than or equal

5 from Arithmetic, 3 from Logical and 6 from branch category

Operation	Type	Format	Opcode	funct	Instruction
ADD	Arithmetic	R-format	1010	000	ADD $r_{0,}$ r_1, r_2
SUB	Arithmetic	R-format	1010	010	SUB $r_{0,}$ r_1, r_2
AND	Logical	R-format	1010	100	AND $r_{3,}$ r_4, r_5
OR	Logical	R-format	1010	101	OR $r_{3,}$ r_4, r_5
ADDI	Arithmetic	I-format	0011	XXX	ADDI $r_{8,}$ r_7, 100
BEQ	Conditional Branch	I-format	0010	XXX	beq $r_{5,}$ r_6, 100
SLT	Logical	R-format	1010	011	slt $r_{5,}$ r_6, r_7
LW	Load-store	I-format	0000	XXX	lw $r_{5,}$ $(r_6)0$
SW	Load-store	I-format	0001	XXX	sw $r_{5,}$ $(r_6)0$

5. Formats:

R-format:

Name	**Bit Fields**				
Opcode	**r_s(1st source)**	**r_t(2nd source)**	**r_d(destination)**	**funct**	
4 bits	**3 bits**	**3 bits**	**3 bits**	**3 bits**	

I-format:

Name	**Bit Fields**		
Opcode	**r_s(1st source)**	**r_t(destination)**	**Immediate/address**
4 bits	**3 bits**	**3 bits**	**6 bits**

6. List of Registers:

Registers	Values
r_0	000
r_1	001
r_2	010
r_3	011
r_4	100
r_5	101
r_6	110
r_7	111

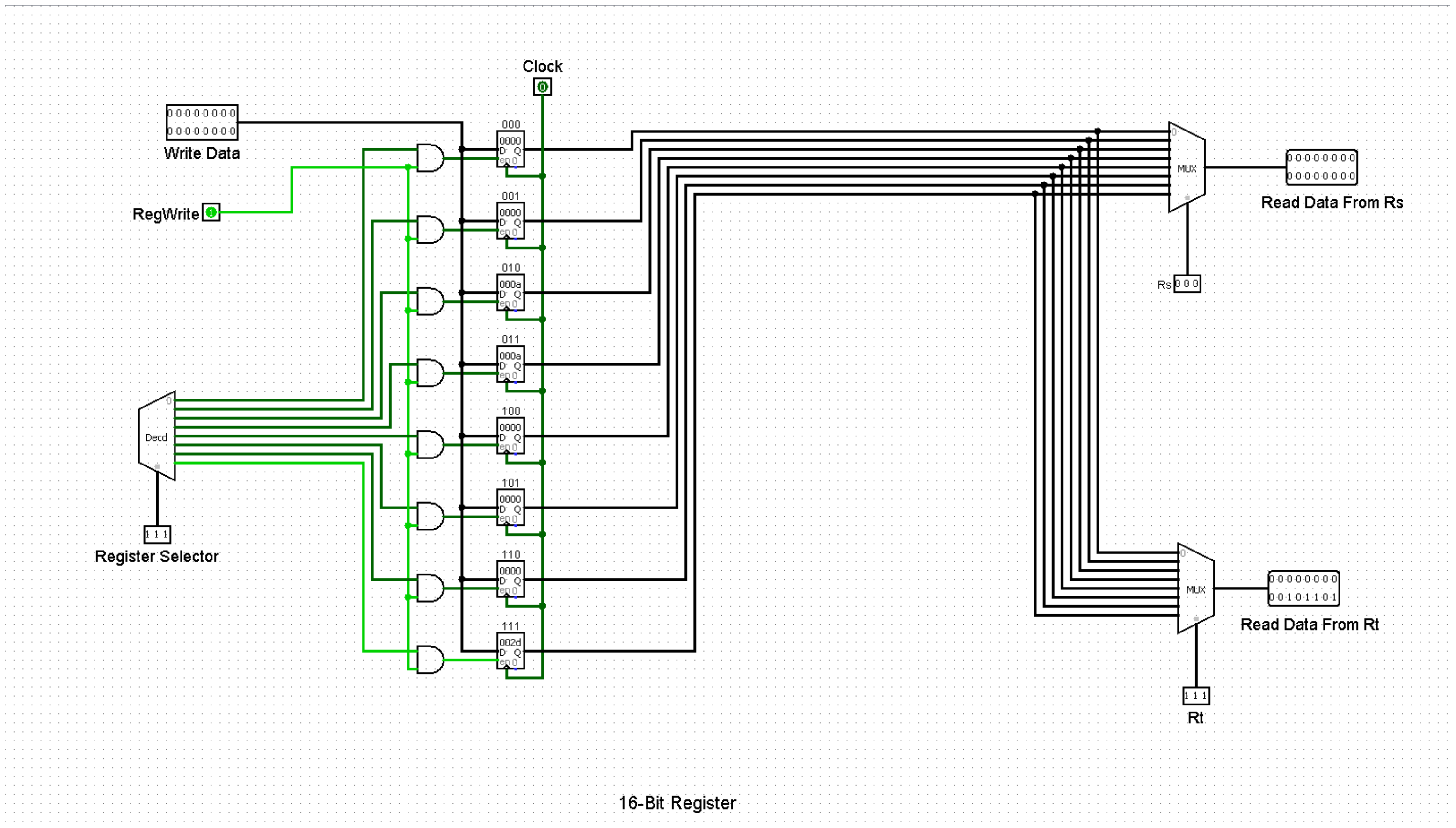


Fig: 16-bit Register

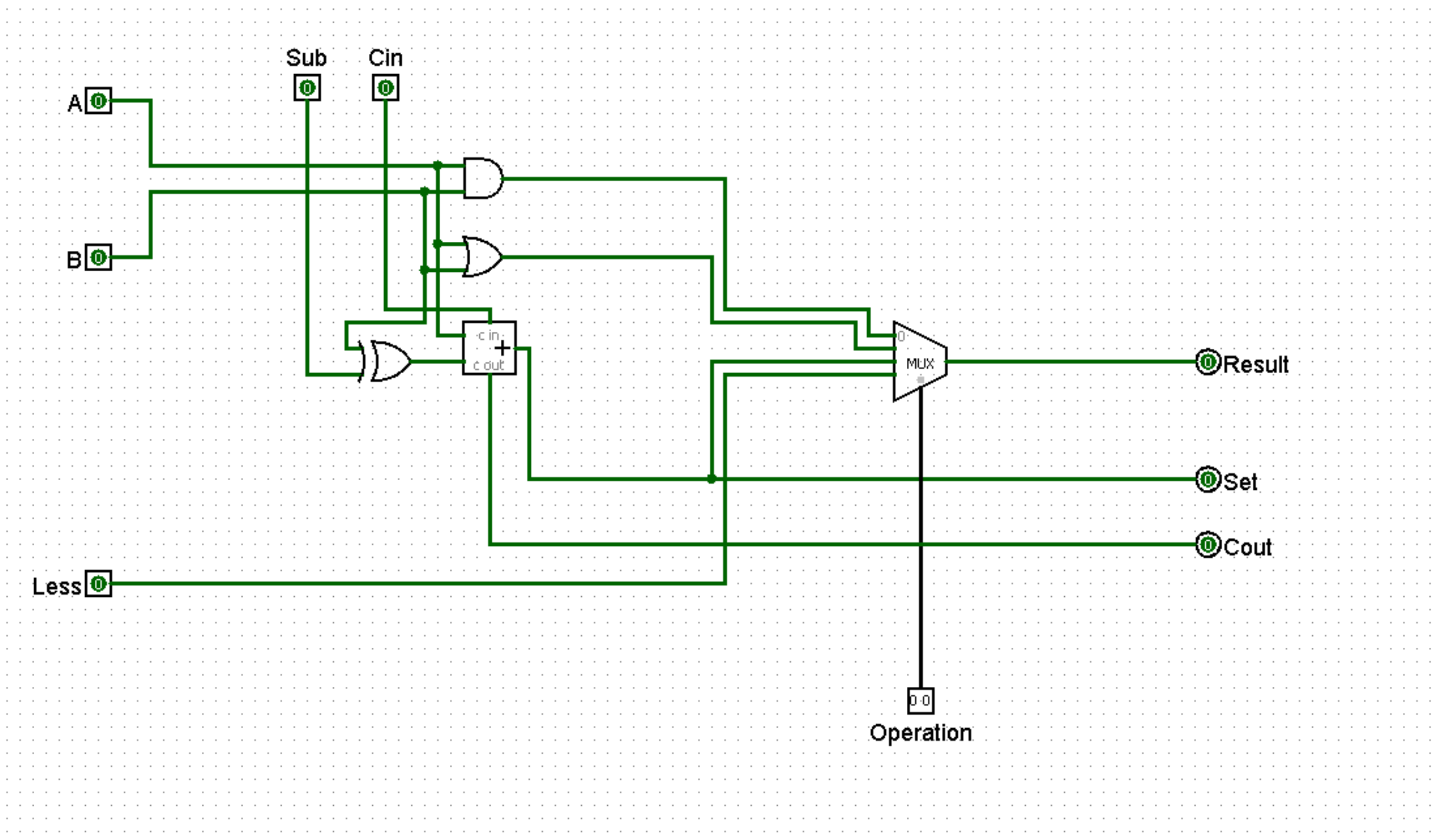


Fig: 1-bit ALU

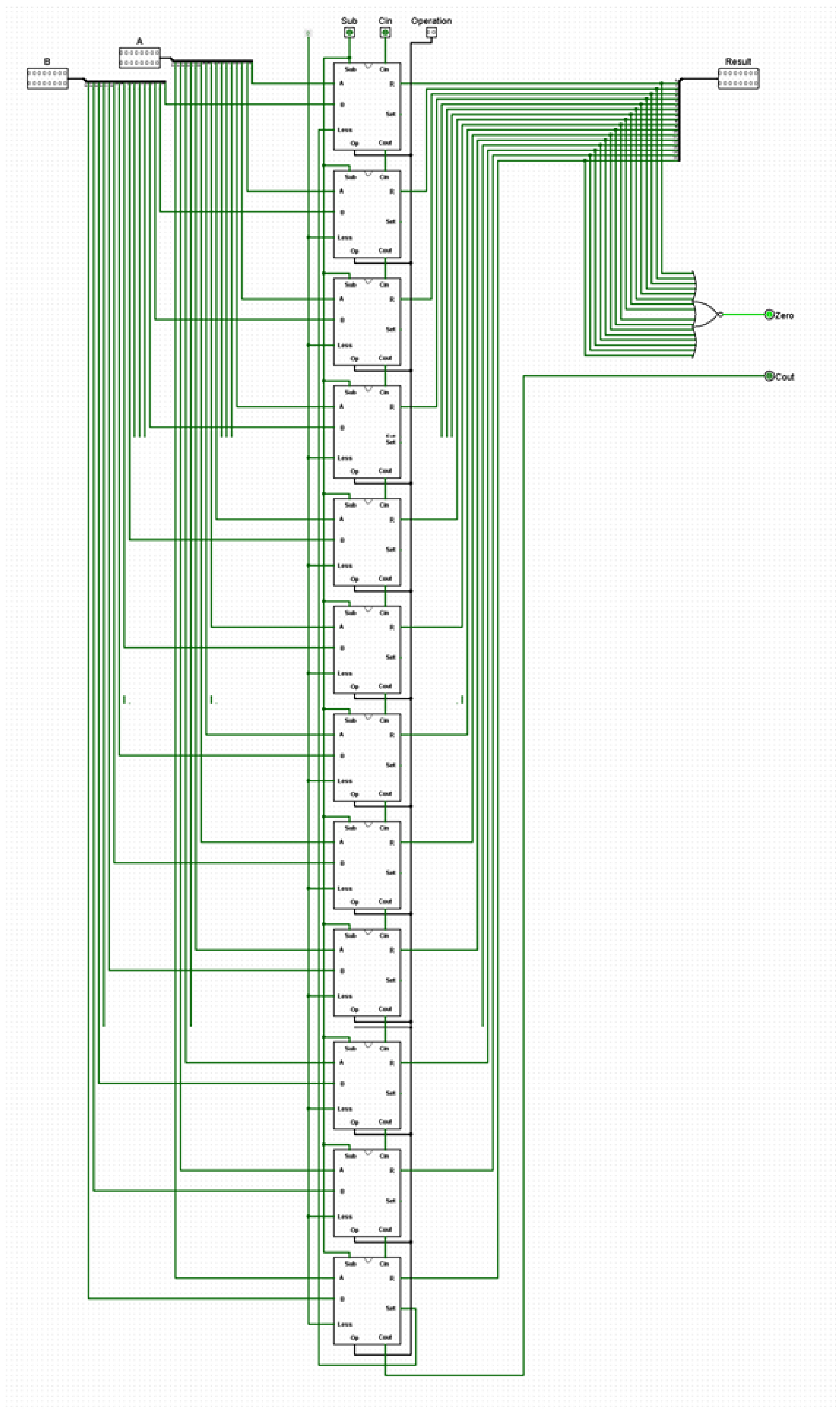
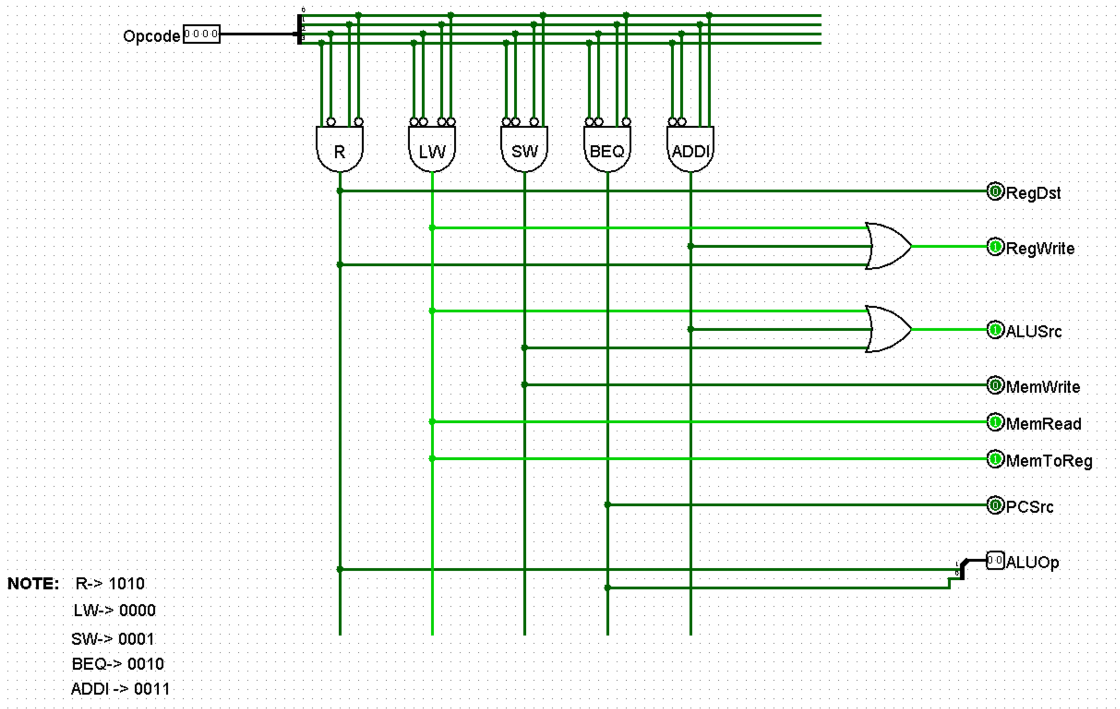


Fig: 16-bit ALU

*Control Unit:

Im	Op Code	RegDst	ALU Src	Memto Reg	Reg Write	Mem Read	Mem Wr	PC Src	ALU Op1	ALU Op2
R	1010	1	0	0	1	0	0	0	1	0
LW	0000	0	1	1	1	1	0	0	0	0
SW	0001	0	1	0	0	0	1	0	0	0
BEQ	0010	0	0	0	0	0	0	1	0	1
ADDI	0011	0	1	0	1	0	0	0	0	0



NOTE: R-> 1010
LW-> 0000
SW-> 0001
BEQ-> 0010
ADDI -> 0011

Fig: Control unit

* ALU Control Unit:

ALUOp1	ALUOp1	F2 F1 F0	Bin S1 S0	
1	0	1 0 0	0 0 0	AND
1	0	1 0 1	0 0 1	OR
1	0	0 0 0	0 1 0	ADD
1	0	0 1 0	1 1 0	SUB
0	0	x x x	0 1 0	ADD
1	0	0 1 1	1 1 1	SLT
0	1	x x x	1 1 0	SUB

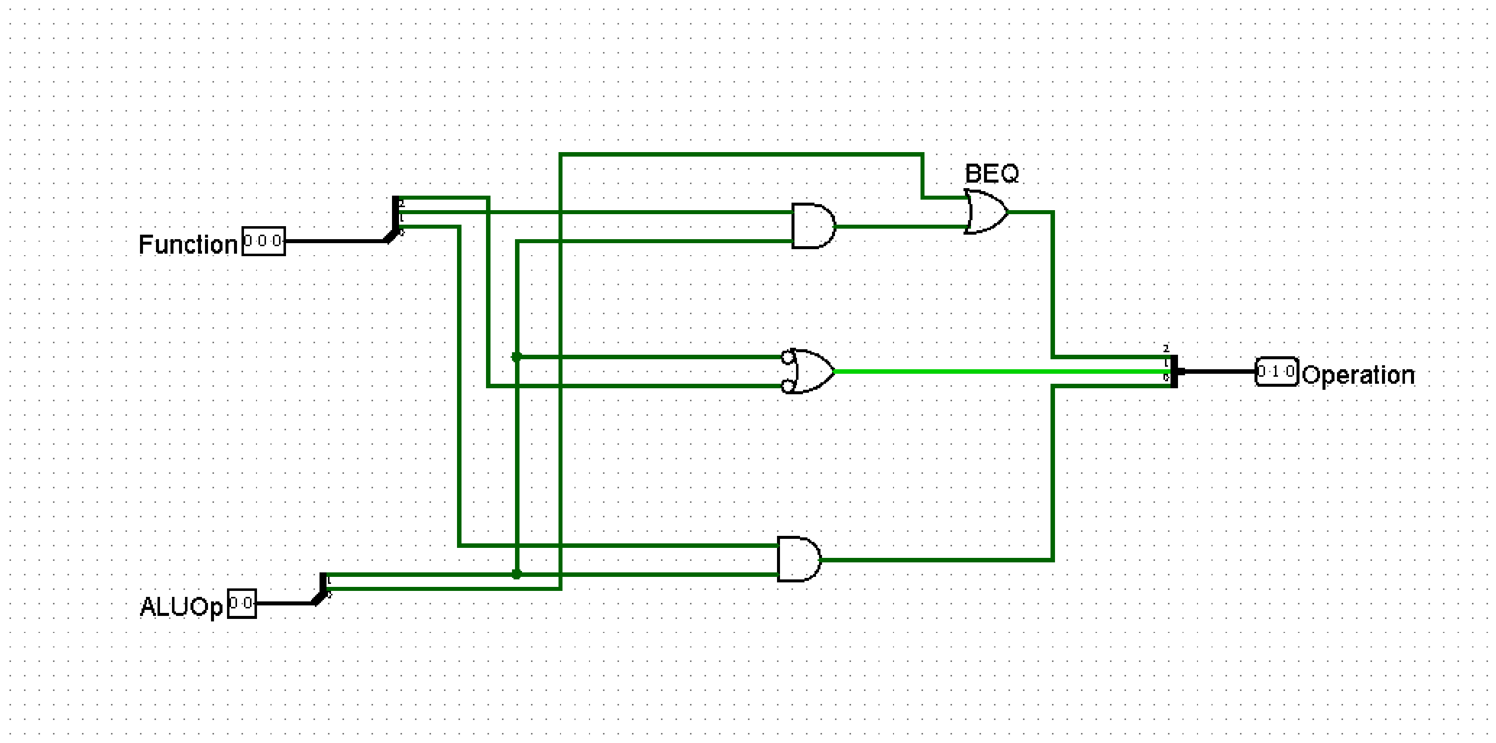


Fig: ALU Control unit

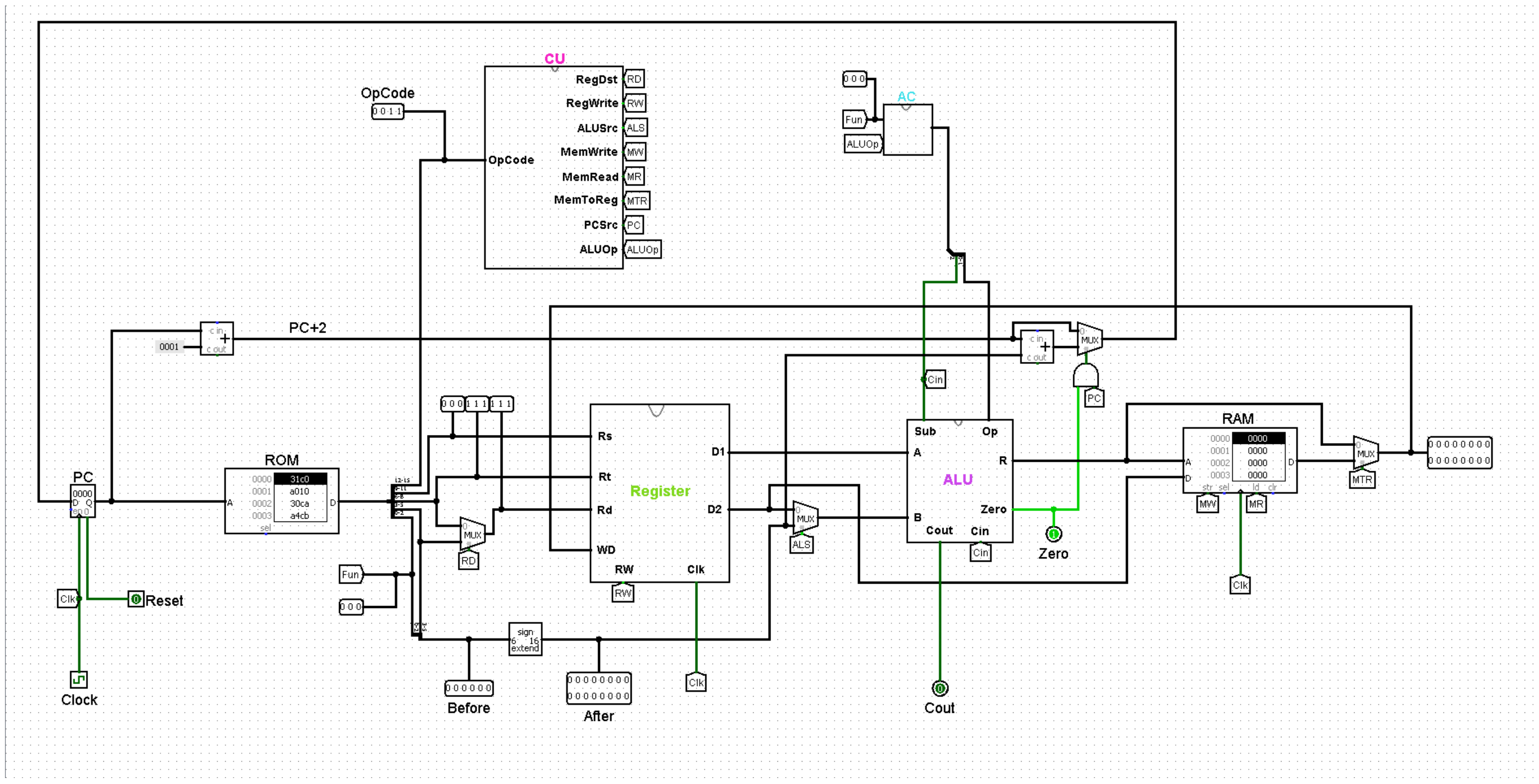
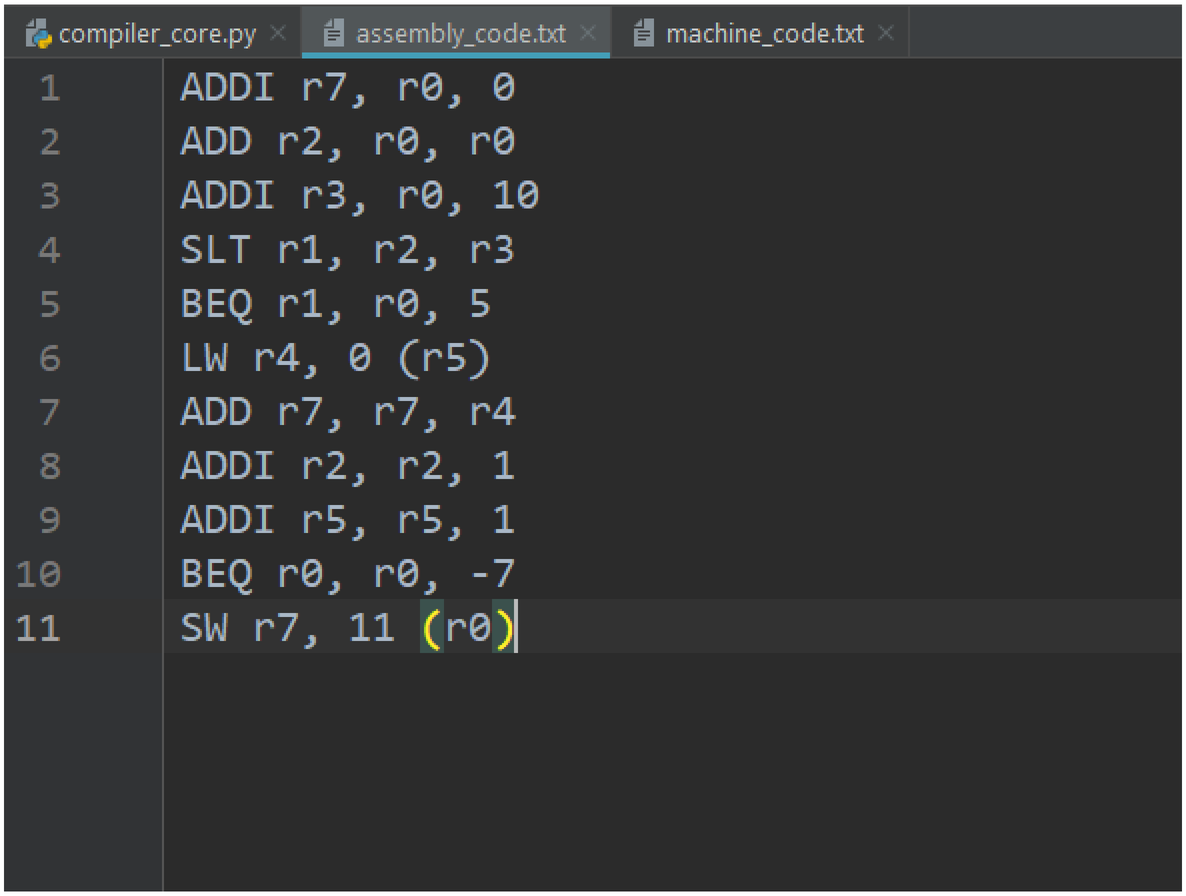
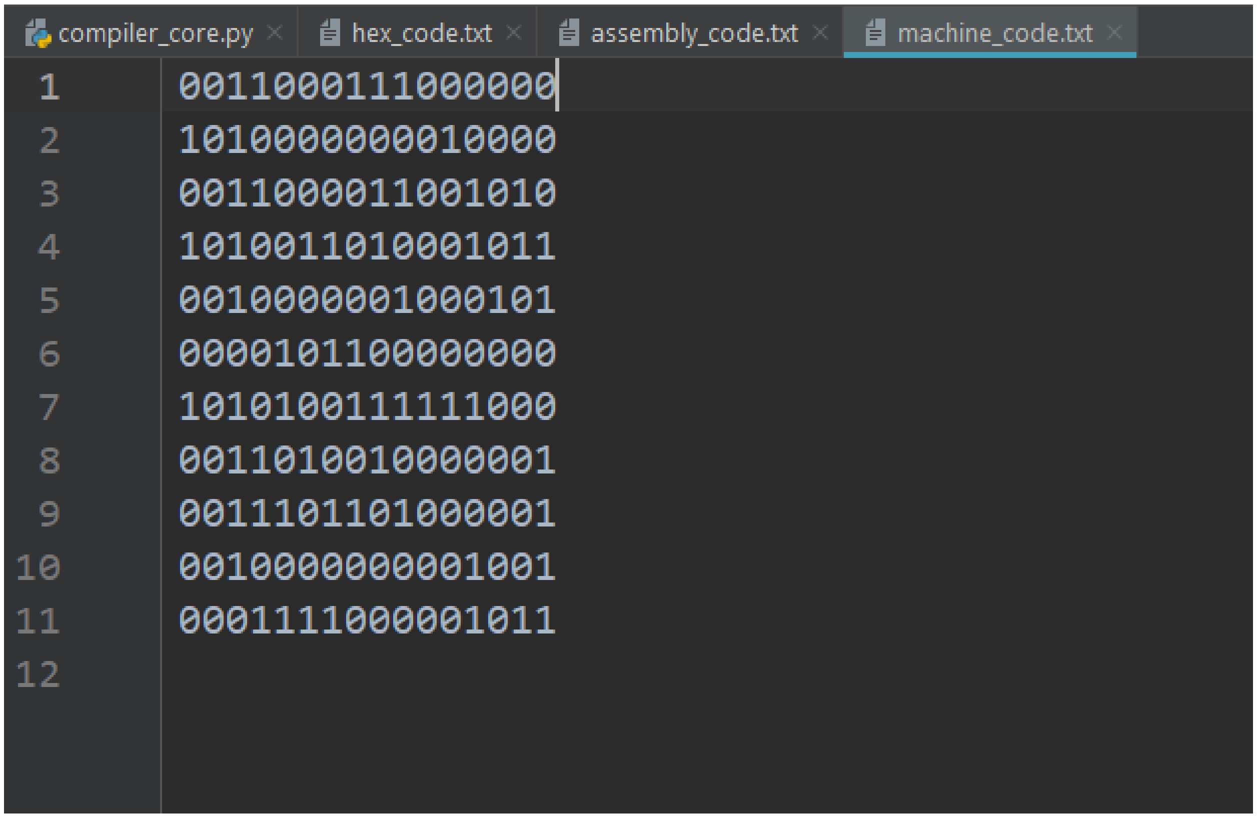


Fig: 16-bit Processor/Datapath



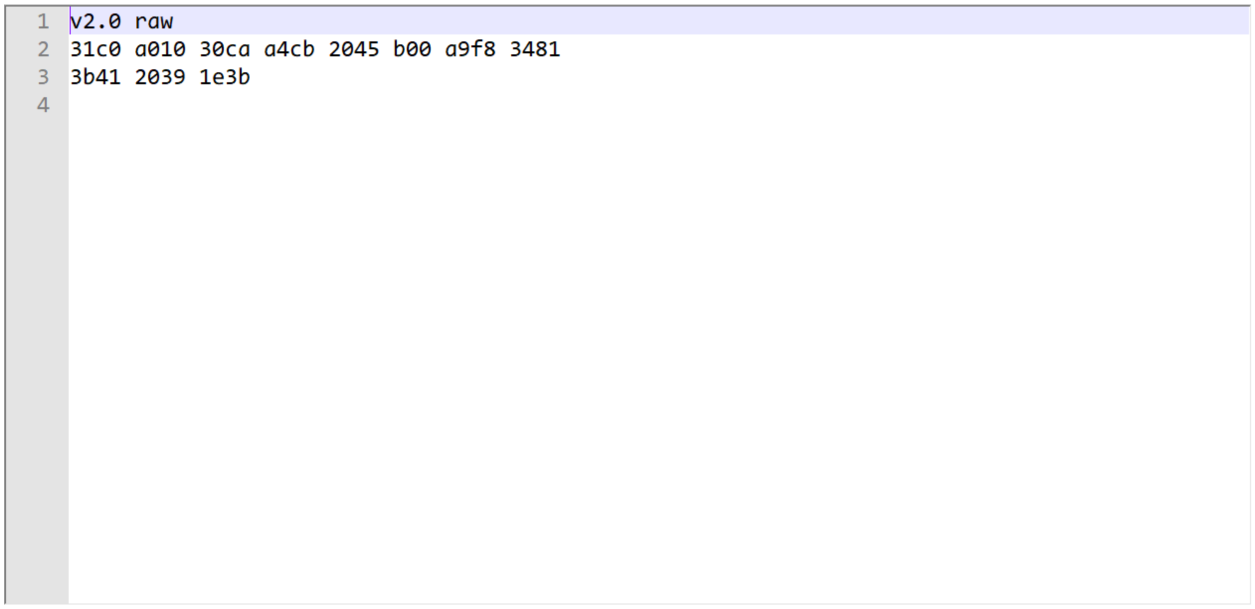
```
ADDI r7, r0, 0
ADD r2, r0, r0
ADDI r3, r0, 10
SLT r1, r2, r3
BEQ r1, r0, 5
LW r4, 0 (r5)
ADD r7, r7, r4
ADDI r2, r2, 1
ADDI r5, r5, 1
BEQ r0, r0, -7
SW r7, 11 (r0)
```

Fig: Assembly Code



```
0011000111000000
1010000000010000
0011000011001010
1010011010001011
0010000001000101
0000101100000000
1010100111111000
0011010010000001
0011101101000001
0010000000001001
0001111000001011
```

Fig: Binary Machine Code



```
v2.0 raw
31c0 a010 30ca a4cb 2045 b00 a9f8 3481
3b41 2039 1e3b
```

Fig: Hexadecimal Output

THANK YOU!