CO224 - Computer Architecture
Lab 5 - Building Simple Processor
Part 5 - Extended ISA

Group 34
E/18/354 THARAKA K.K.D.R.
E/18/349 THALISHA W.G.A.P.

EXTENDED INSTRUCTIONS

INSTRUCTION	Description	OPCODE	ALU SELECT
bne	If values in given registers are not equal, branch given no of instructions forward or backward.	8’b00010001	3’b000
sll	Apply logical shift left given times on value in 2nd register and place the result in 1st register.	8'b00001101	3'b101
srl	Apply logical shift right given times on value in 2nd register and place the result in 1st register.	8'b00001110	3'b101
sra	Apply arithmetic shift right given times on value in 2nd register and place the result in 1st register.	8'b00001111	3'b101
ror	Apply rotate right given times on value in 2nd register and place the result in 1st register.	8'b00111111	3'b111
mul	Multiply value in 2nd register by value in 3rd register and place in 1st register.	8'b00001001	3'b100

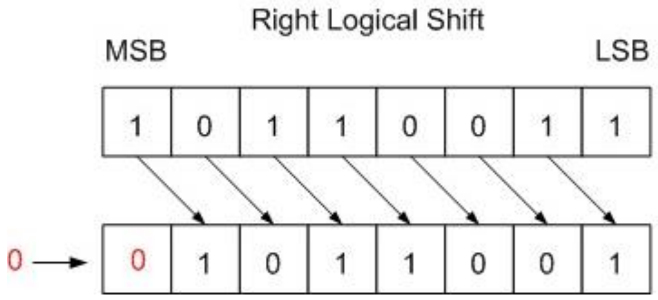
srl

DESCRIPTION

Apply logical shift right given times on value in 2nd register and place the result in 1st register.

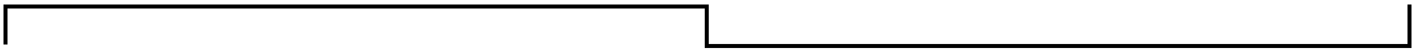
ASSEMBLY EXAMPLE

```
srl 3 2 0x01
```

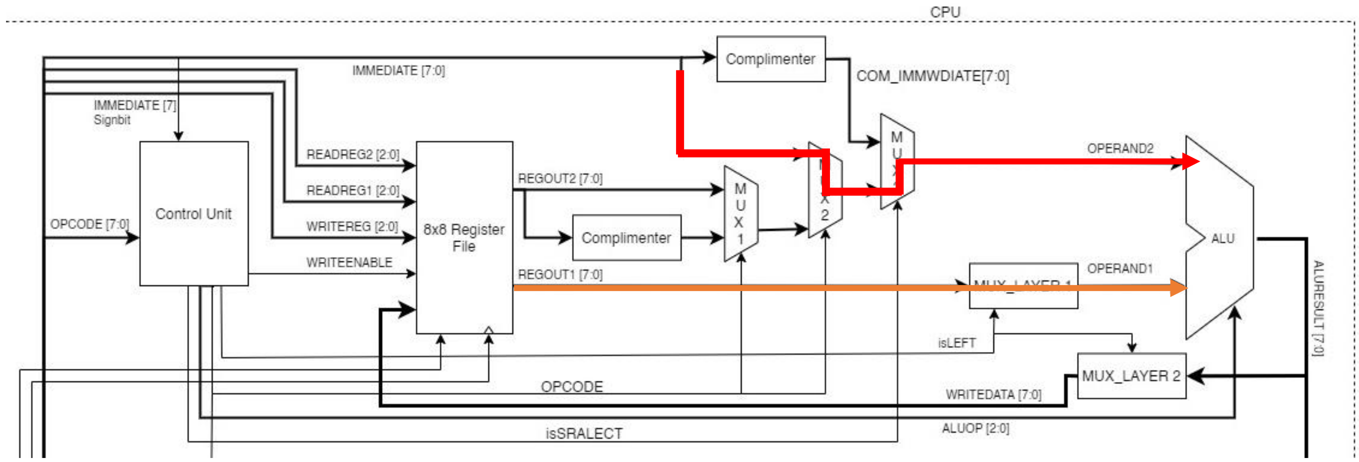


TIMING DIAGRAM

srl / sll:

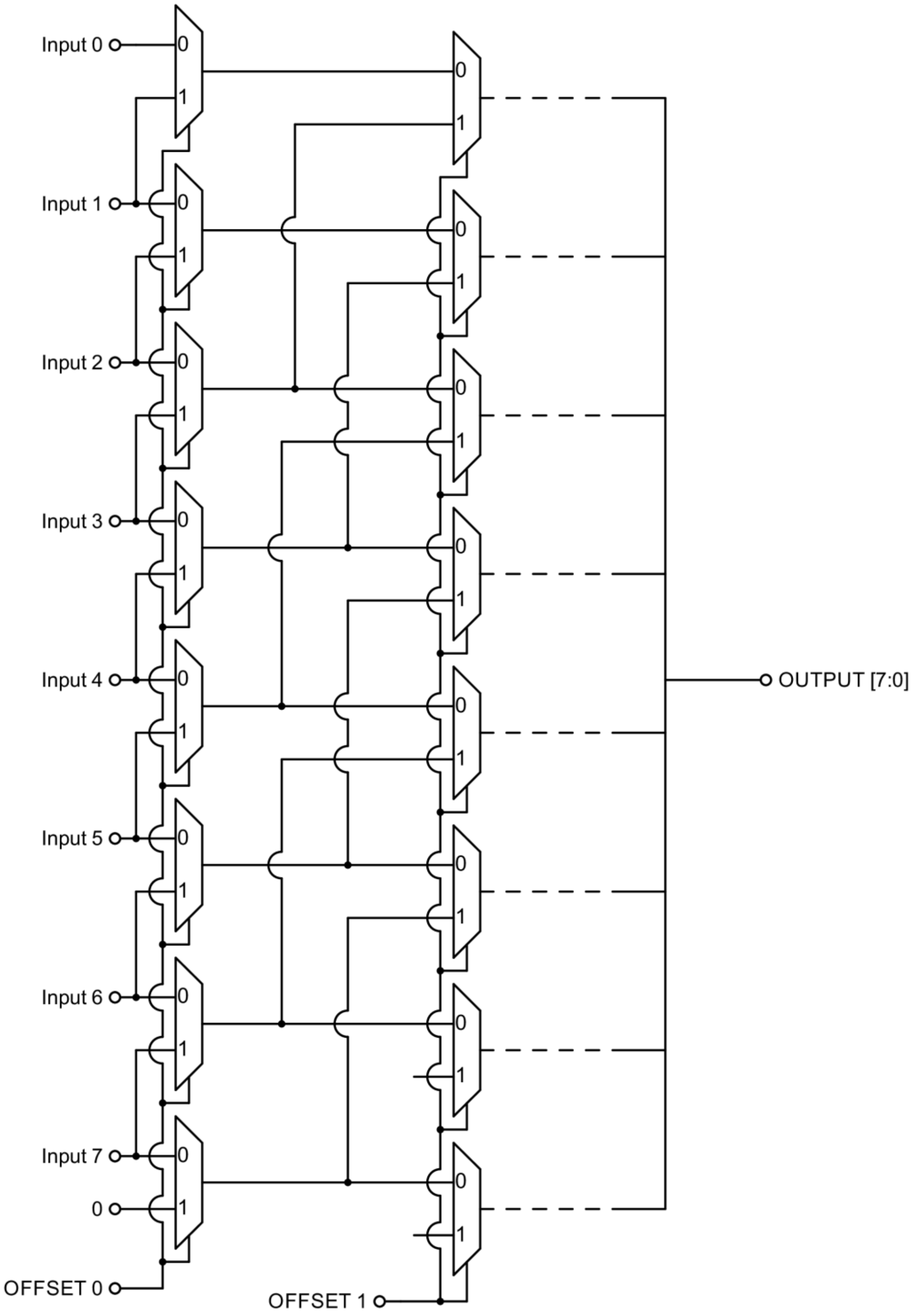


Pc Update	Instruction Memory Read	Register Read	ALU	
#1	#2	#2	#1	
	Pc+4 Adder	Decode		
	#1	#1		
Register Write				
#1				



srl MODULE IMPLEMENTATION

Only 2 MUX layers are shown. There are 8 similar MUX layers



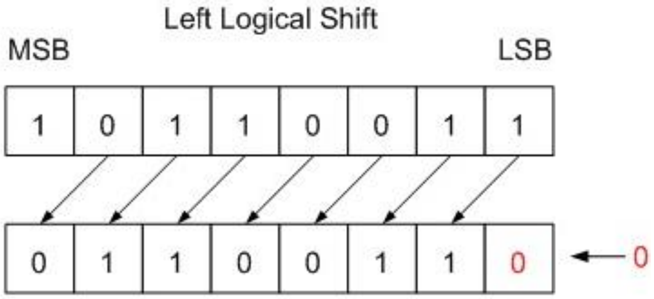
sll

DESCRIPTION

Apply logical shift left given times on value in 2nd register and place the result in 1st register.

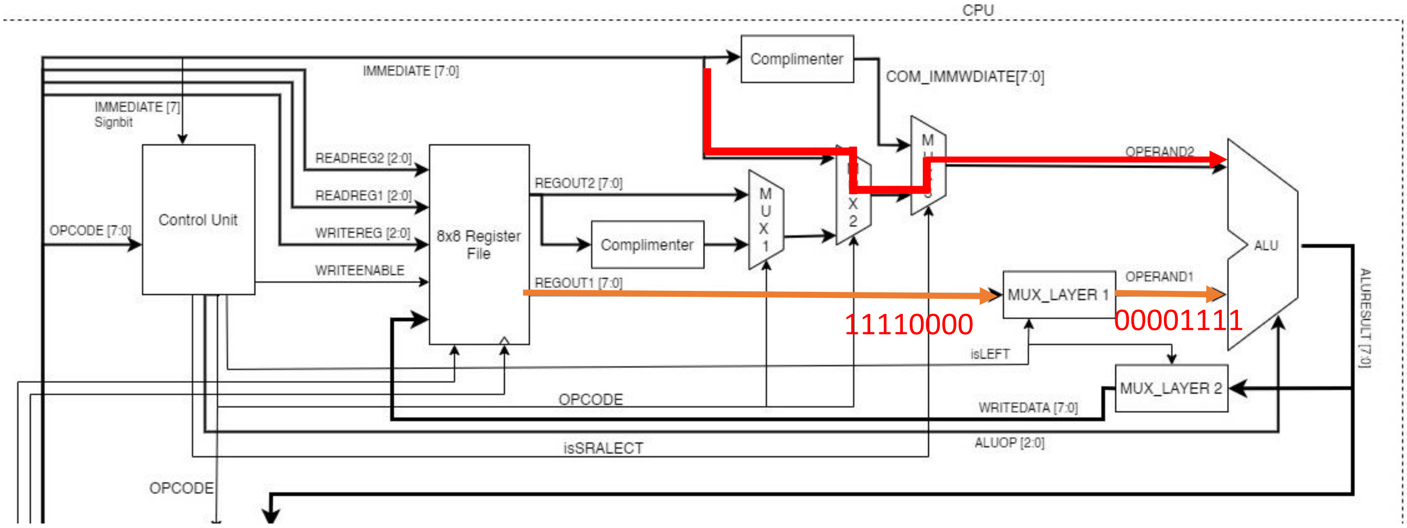
ASSEMBLY EXAMPLE

```
sll 3 2 0x01
```



sll MODULE IMPLEMENTATION

Uses the same module that used for instruction srl. For differentiate left and right shifts for the left shift, the ALU input value and ALU result are backwarded.



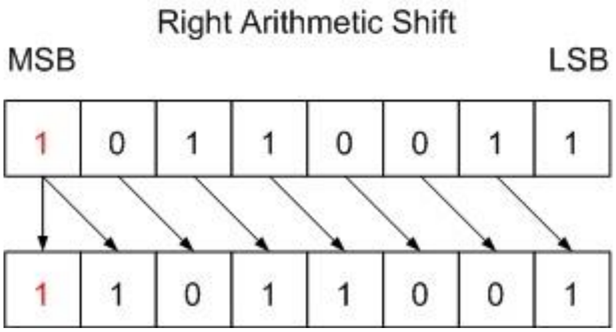
sra

DESCRIPTION

Apply arithmetic shift right given times on value in 2nd register and place the result in 1st register.

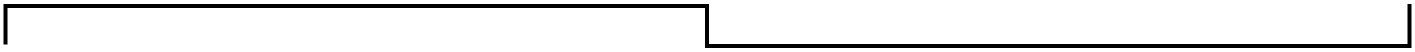
ASSEMBLY EXAMPLE

```
sra 3 5 0x01
```



TIMING DIAGRAM

sra:



Pc Update	Instruction Memory Read		Register Read		ALU		
#1	#2		#2		#1		
	Pc+4 Adder		Decode				
	#1		#1				
Register Write							
#1							

sra MODULE IMPLEMENTATION

Uses the same module that used for instruction srl.

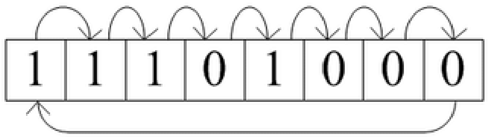
ror

DESCRIPTION

Apply rotate right given times on value in 2nd register and place the result in 1st register.

ASSEMBLY EXAMPLE

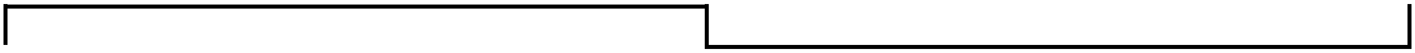
```
ror 3 5 0x01
```



Rotate Right

TIMING DIAGRAM

ror:



Pc Update	Instruction Memory Read		Register Read		ALU	
#1	#2		#2		#1	
	Pc+4 Adder		Decode			
	#1		#1			
Register Write						
#1						

ror MODULE IMPLEMENTATION

Uses the same module that used for instruction srl.

mul

DESCRIPTION

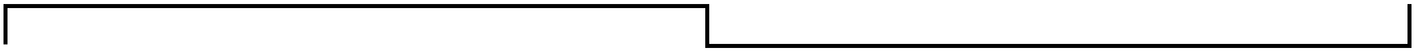
If values in given registers are not equal, branch given no of instructions forward or backward.

ASSEMBLY EXAMPLE

```
mul 4 1 5
```

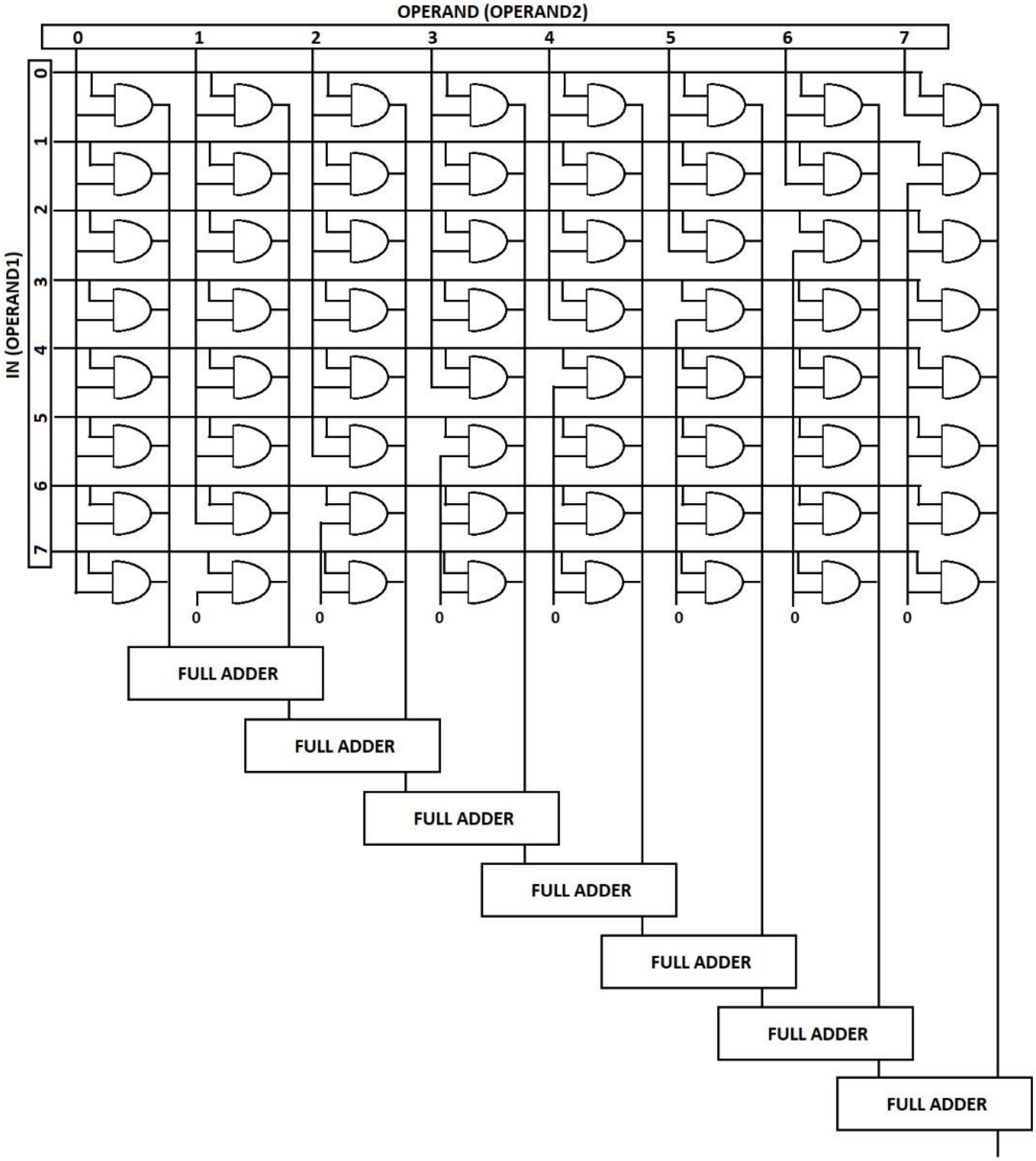
TIMING DIAGRAM

mul:



Pc Update	Instruction Memory Read		Register Read		ALU		
#1	#2		#2		#1		
	Pc+4 Adder		Decode				
	#1		#1				
Register Write							
#1							

mul MODULE IMPLEMENTATION



bne

DESCRIPTION

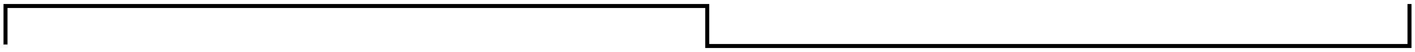
Multiply value in 2nd register by value in 3rd register and place in 1st register.

ASSEMBLY EXAMPLE

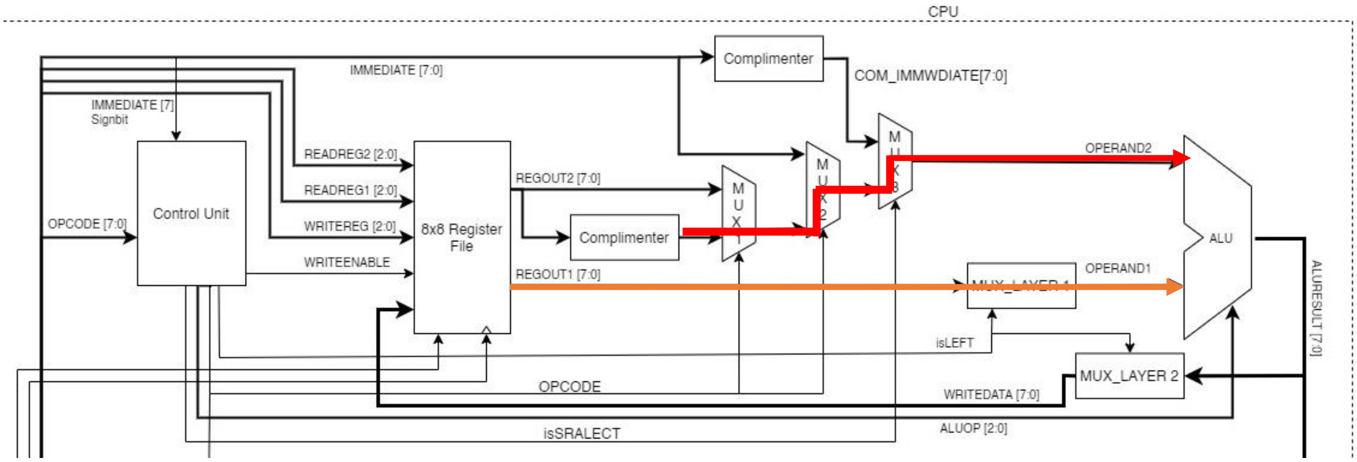
```
    mul 5 2 3
```

TIMING DIAGRAM

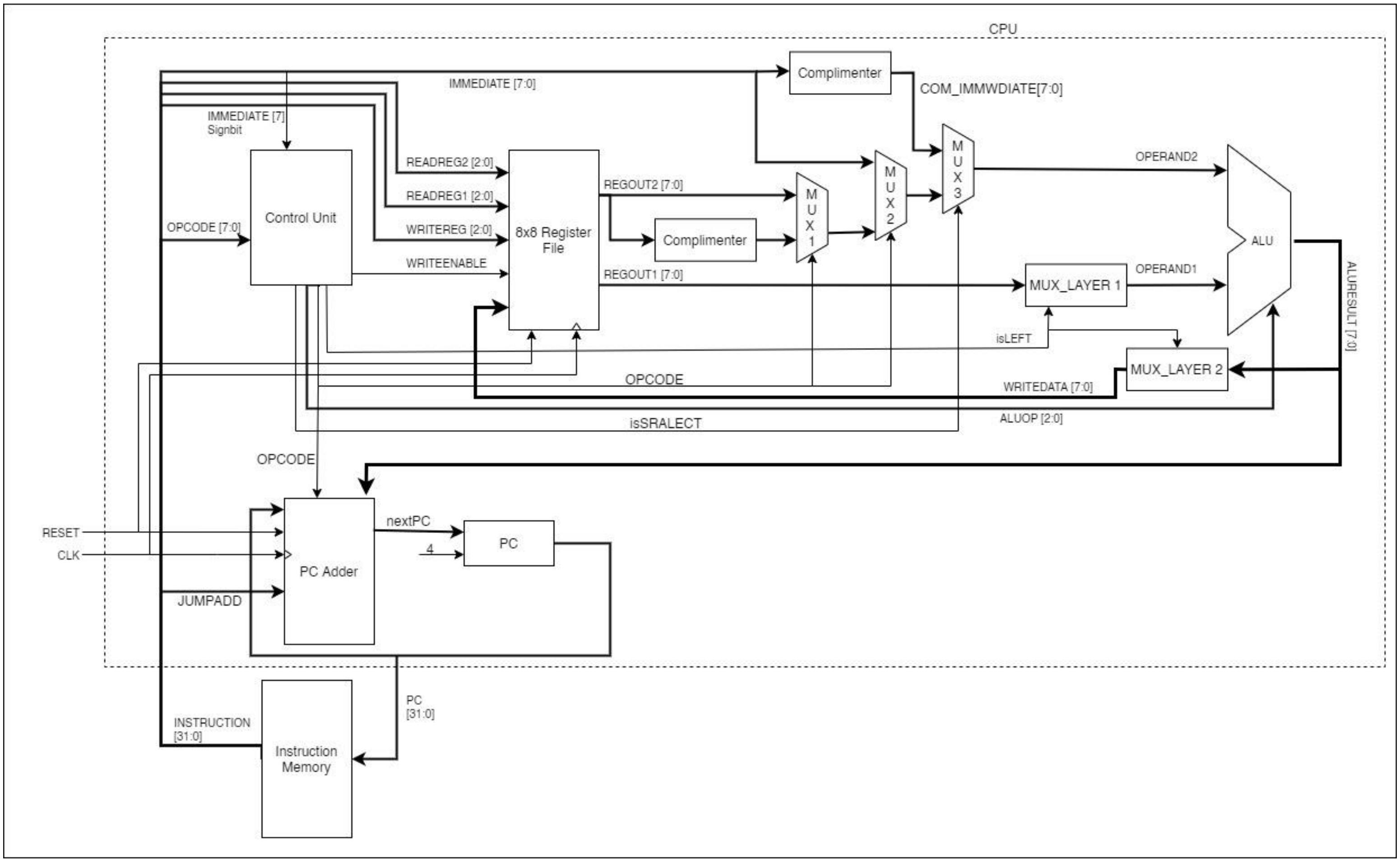
bne:



Pc Update	Instruction Memory Read		Register Read		2’s Comp	ALU	
#1	#2		#2		#1	#2	
	Pc+4 Adder		Branch/Jump Target Adder				
	#1		#2				
			Decode				
			#1				



Bock Diagram


CPU
Complimenter
IMMEDIATE [7:0]
COM_IMMWDIATE[7:0]
IMMEDIATE [7]
Signbit
READREG2 [2:0]
READREG1 [2:0]
WRITEREG [2:0]
WRITEENABLE
OPCODE [7:0]
Control Unit
8x8 Register File
REGOUT2 [7:0]
REGOUT1 [7:0]
Complimenter
MUX 1
MUX 2
MUX 3
OPERAND2
OPERAND1
ALU
MUX_LAYER 1
MUX_LAYER 2
ALURESULT [7:0]
isLEFT
OPCODE
WRITEDATA [7:0]
ALUOP [2:0]
isSRALECT
OPCODE
RESET
CLK
PC Adder
nextPC
4
PC
JUMPADD
INSTRUCTION [31:0]
Instruction Memory
PC [31:0]