CO 224 – LAB 5

BUILDING A SIMPLE PROCESSOR

WIJERATHNA I.M.K.D.I. : E/19/446
DISSANAYAKE D.M.I.G. : E/19/090
GROUP 13
SEMESTER 4
01/06/2023

Introduction:

In this lab report, we present our findings and documentation on the design and implementation of an Extended Instruction Set Architecture (ISA) for a CPU using Verilog. The goal of this project was to expand the capabilities of the CPU by incorporating additional instructions, specifically bne (Branch Not Equal), mult (Multiply), lsl (Logical Shift Left), and lsr (Logical Shift Right). By introducing these new instructions, we aimed to enhance the CPU's functionality, improve performance, and enable more complex computations.

The addition of these instructions required careful consideration of their functionalities, encoding, and integration within the existing CPU architecture. The Verilog hardware description language was utilized for the design and simulation of the extended CPU, allowing us to model and implement the desired changes effectively.

Part 5–Extended ISA :

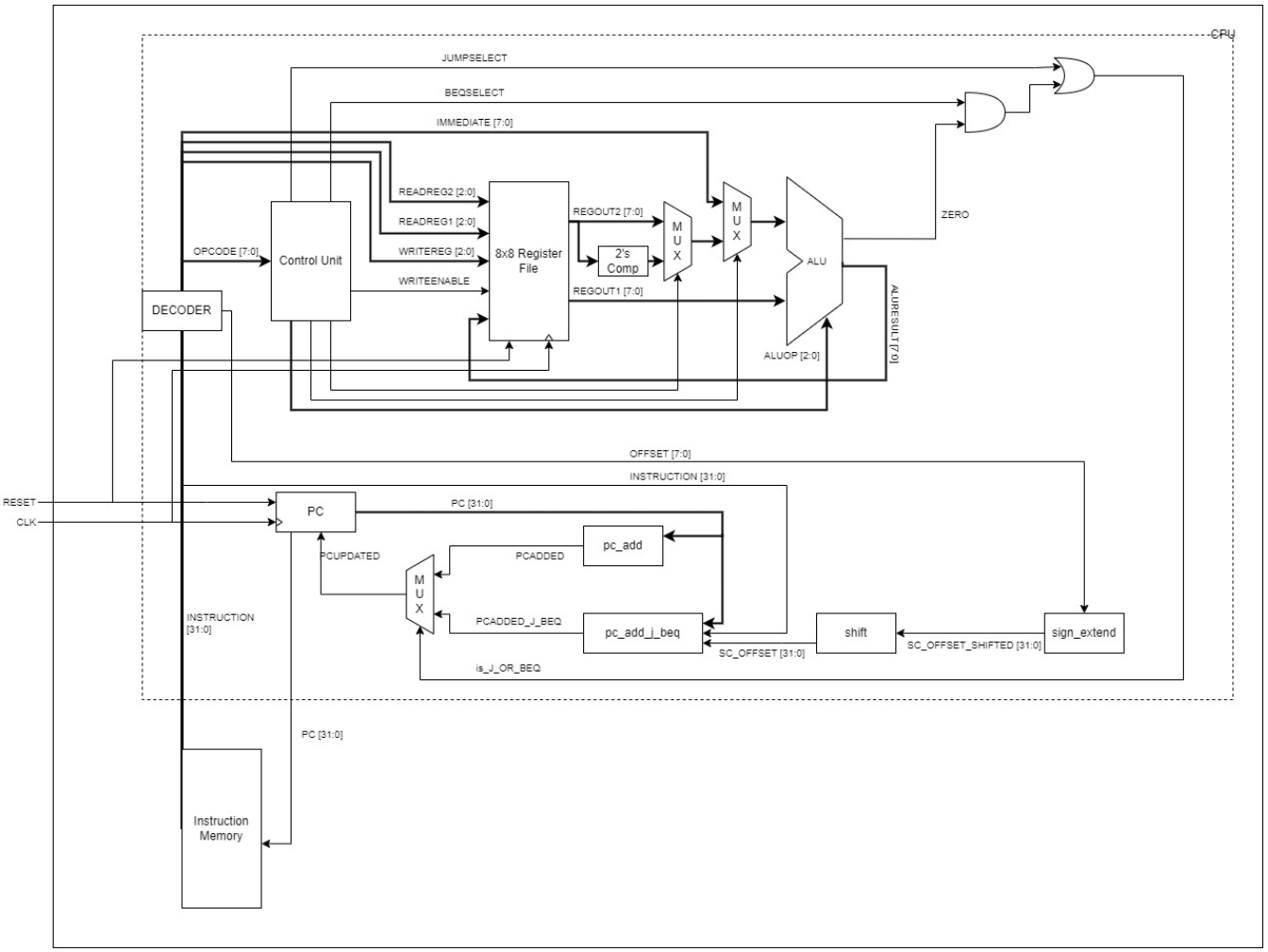


FIGURE 1: COMPLETE CPU BLOCK DIAGRAM

In part 5 of lab 5 it is intended to extend the instruction set architecture with 2 or more instructions.
Hence we have modified our alu.v to support multiplication, logical shift left , logical shift right , rotate right.

Moreover, we added a bne (Branch Not Euqal) operation as well.

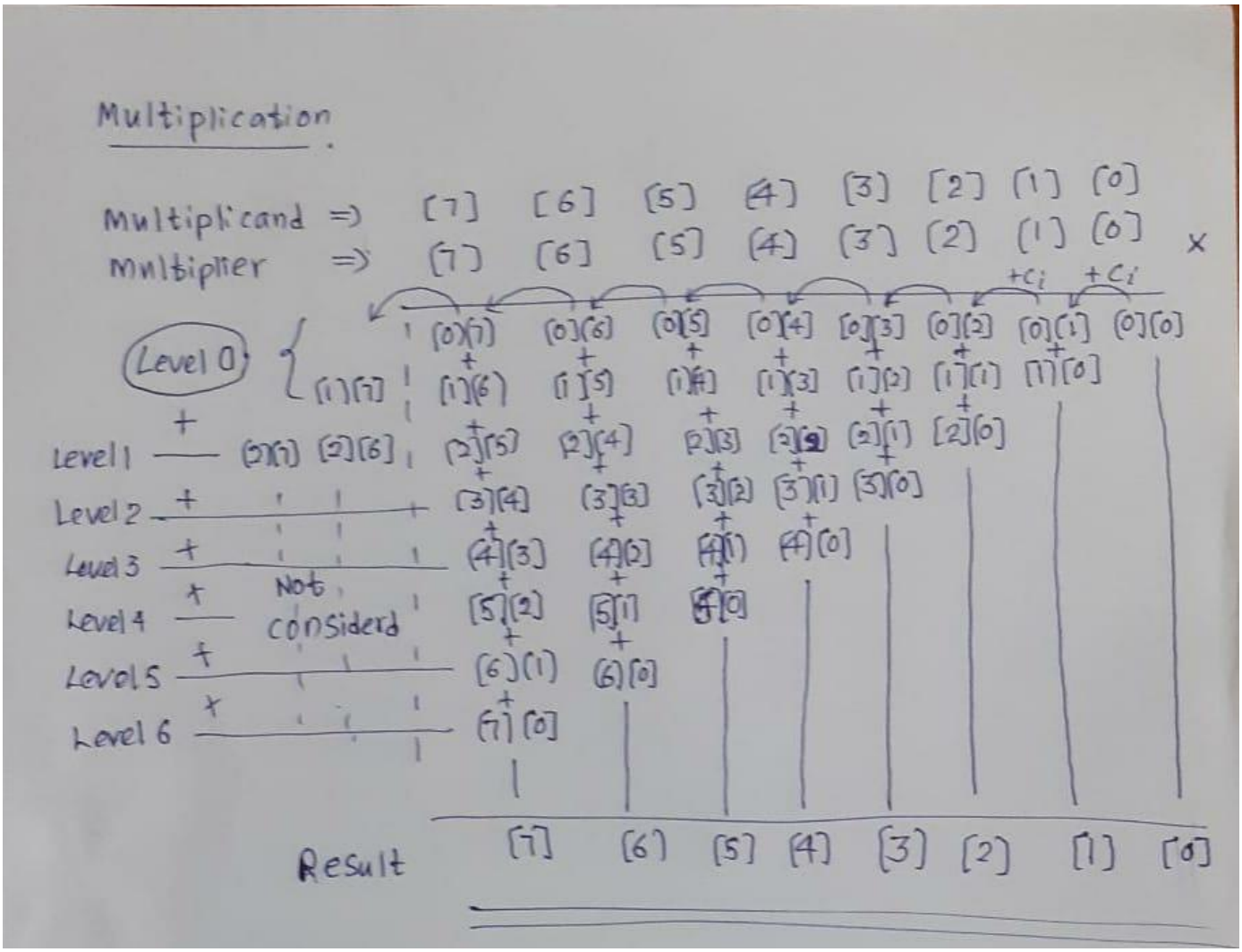
Additional Instruction 1: mult

Operation : Multiplication (Result= DATA1 * DATA2)

ALUOP : 110

Time delay : 3 time units (because of 7 levels of Full adders used)

Design of the mult unit:



Implementing the code with gtkwave :

lab5_marking.s

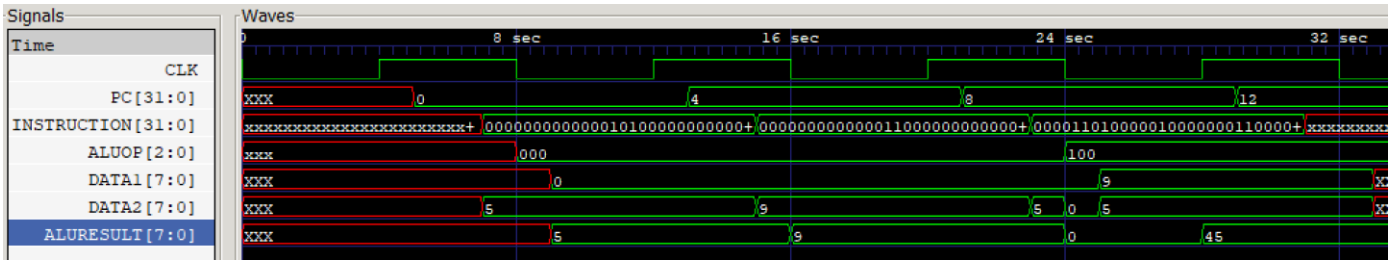
```
1   loadi 5 0x05    // r5 = 5
2   loadi 6 0x09    // r6 = 9
3   mult 4 6 5      // (multiply value in register 1 by value in register 2, and place the result in register 4)
```

```
kalindu@DESKTOP-R08BL8B:/mnt/d/PE19 Fourth Semester/CO224-Computer Architecture/Labs/Lab5/Part5New$ ./generate_memory_image.sh lab5_marking.s
Instruction memory content generated!
kalindu@DESKTOP-R08BL8B:/mnt/d/PE19 Fourth Semester/CO224-Computer Architecture/Labs/Lab5/Part5New$
```

instr_mem.mem

```
1   00000101 00000000 00000101 00000000
2   00001001 00000000 00000110 00000000
3   00000101 00000110 00000100 00001101
```

```
PS D:\PE19 Fourth Semester\CO224-Computer Architecture\Labs\Lab5\Part5New> vvp .\test
WARNING: .\cpu_tb.v:36: $readmemb(instr_mem.mem): Not enough words in the file for the requested range [0:1023].
VCD info: dumpfile cpu_wavedata.vcd opened for output.
                TIME     R0      R1      R2      R3      R4      R5      R6      R7
                   5      0       0       0       0       0       0       0       0
                  13      0       0       0       0       0       5       0       0
                  21      0       0       0       0       0       5       9       0
                  29      0       0       0       0      45       5       9       0
.\cpu_tb.v:62: $finish called at 305 (1s)
PS D:\PE19 Fourth Semester\CO224-Computer Architecture\Labs\Lab5\Part5New>
```



Timing diagram details :

PC Update	Instruction Memory Read	Register Read	ALU
#1	#2	#2	#3
	PC+4 Adder	Decode	
	#1	#1	
Register Write			
#1			

Figure 2: Timing details for MULT instruction data path

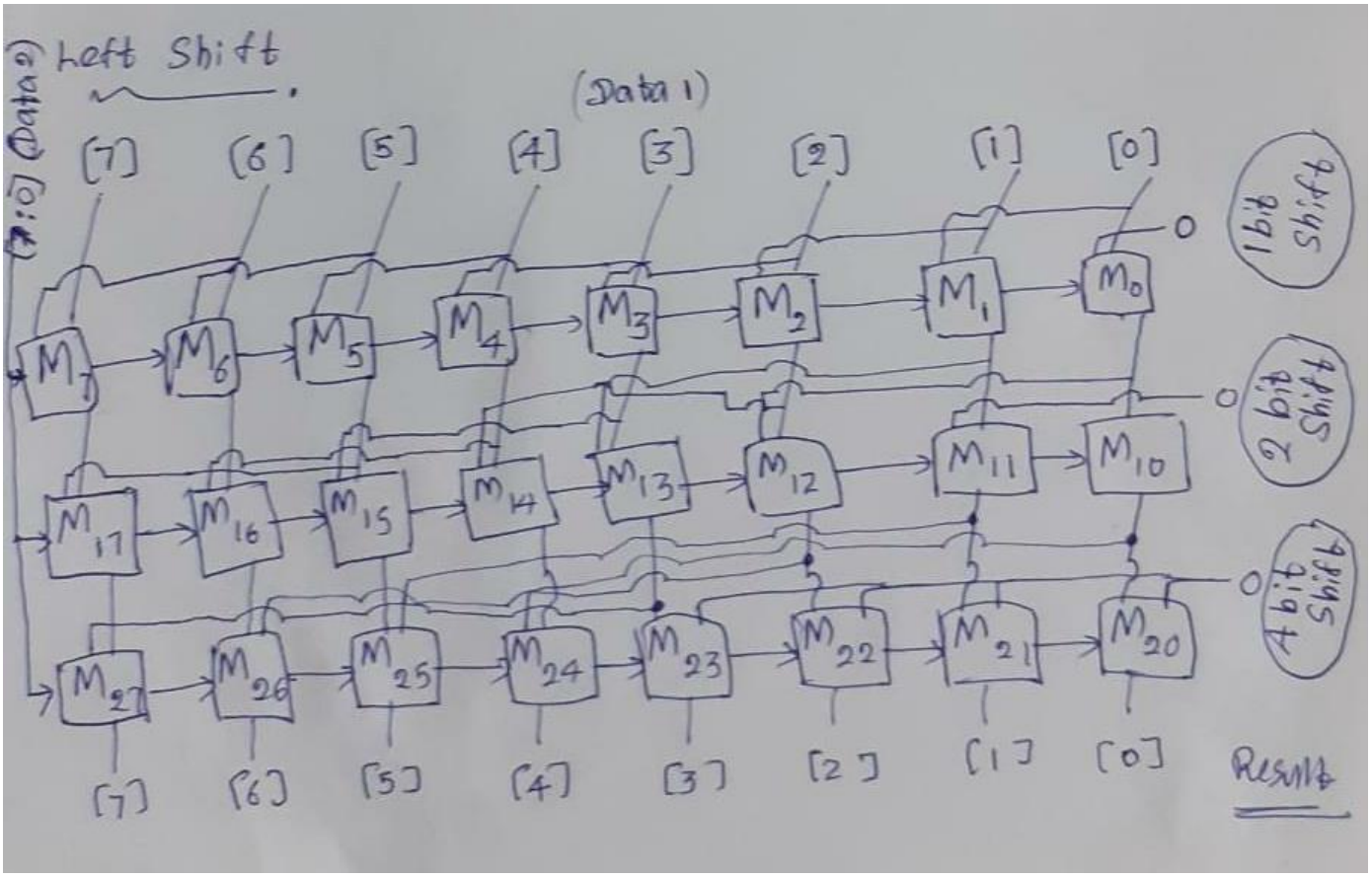
Additional Instruction 2: sll

Operation : Logical Shift Left (Result = DATA1 << DATA2)

ALUOP : 101

Time delay : 2 time units (because of 2-3 levels of Muxes used)

Design of the sll unit:



Implementing the code with gtkwave :

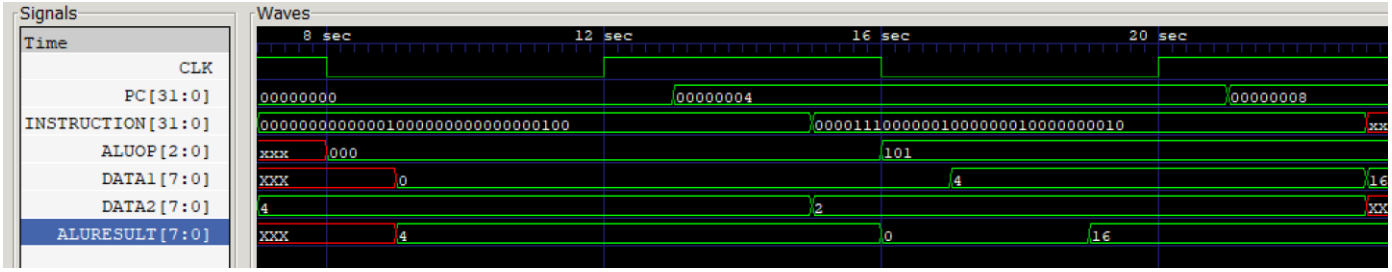
```
lab5_marking.s
4    loadi 4 0x04      // r4 = 4
5    sll 4 4 0x02     //(apply logical shift left 2 times on value in register 1, and place the result in register 4)
```

```
kalindu@DESKTOP-R08BL8B:/mnt/d/PE19 Fourth Semester/CO224-Computer Architecture/Labs/Lab5/Part5New$ ./generate_memory_image.sh lab5_marking.s
Instruction memory content generated!
kalindu@DESKTOP-R08BL8B:/mnt/d/PE19 Fourth Semester/CO224-Computer Architecture/Labs/Lab5/Part5New$
```

instr_mem.mem

```
1   00000100 00000000 00000100 00000000
2   00000010 00000100 00000100 00001110
3
```

```
PS D:\PE19 Fourth Semester\CO224-Computer Architecture\Labs\Lab5\Part5New> vvp .\test
WARNING: .\cpu_tb.v:36: $readmemb(instr_mem.mem): Not enough words in the file for the requested range [0:1023].
VCD info: dumpfile cpu_wavedata.vcd opened for output.
                TIME     R0      R1      R2      R3      R4      R5      R6      R7
                   5      0       0       0       0       0       0       0       0
                  13      0       0       0       0       4       0       0       0
                  21      0       0       0       0      16       0       0       0
.\cpu_tb.v:62: $finish called at 305 (1s)
PS D:\PE19 Fourth Semester\CO224-Computer Architecture\Labs\Lab5\Part5New>
```



Timing diagram details :

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

Figure 3: Timing details for left shift instruction data path

Additional Instruction 3: slr

Operation : Logical Shift Right (Result= DATA1 >> DATA2)

ALUOP : 110

Time delay : 2 time units (because of 2-3 levels of Muxes used)

Design of the slr unit:

Similar to the design of sll unit.

Implementing the code with gtkwave :

lab5_marking.s

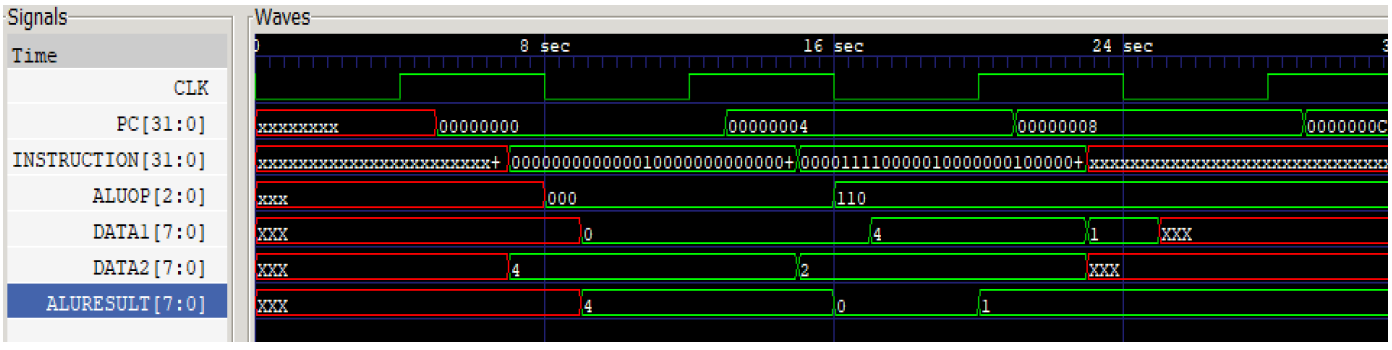
```
4    loadi 4 0x04      // r4 = 4
5    srl 4 4 0x02   ///(apply logical shift right 2 times on value in register 4, and place the result in register 4)
```

```
kalindu@DESKTOP-R08BL8B:/mnt/d/PE19 Fourth Semester/CO224-Computer Architecture/Labs/Lab5/Part5New$ ./generate_memory_image.sh lab5_marking.s
Instruction memory content generated!
kalindu@DESKTOP-R08BL8B:/mnt/d/PE19 Fourth Semester/CO224-Computer Architecture/Labs/Lab5/Part5New$
```

instr_mem.mem

```
1    00000100 00000000 00000100 00000000
2    00000010 00000100 00000100 00001111
3
```

```
PS D:\PE19 Fourth Semester\CO224-Computer Architecture\Labs\Lab5\Part5New> vvp .\test
WARNING: .\cpu_tb.v:36: $readmemb(instr_mem.mem): Not enough words in the file for the requested range [0:1023].
VCD info: dumpfile cpu_wavedata.vcd opened for output.
                TIME    R0      R1      R2      R3      R4      R5      R6      R7
                   5     0       0       0       0       0       0       0       0
                  13     0       0       0       0       4       0       0       0
                  21     0       0       0       0       1       0       0       0
.\cpu_tb.v:62: $finish called at 305 (1s)
PS D:\PE19 Fourth Semester\CO224-Computer Architecture\Labs\Lab5\Part5New>
```



Timing diagram details :

PC Update	Instruction Memory Read	Register Read	ALU
#1	#2	#2	#2

	PC+4 Adder	Decode	
	#1	#1	

Register Write	
#1	

Figure 4: Timing details for right shift instruction data path

Additional Instruction 4: bne

Operation : Branch Not Equal

ALUOP : 001

OPCODE : 00001100

Implementing the code with gtkwave :

lab5_marking.s

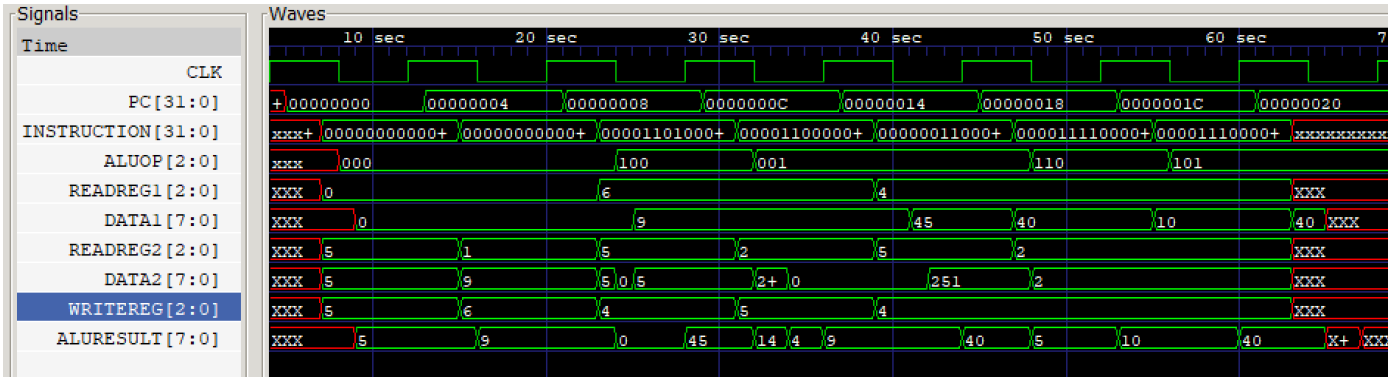
```
1   loadi 5 0x05   // r5 = 5
2   loadi 6 0x09   // r6 = 9
3   mult 4 6 5     // (multiply value in register 1 by value in register 2, and place the result in register 4)
4   bne 5 6 0x02   // (if values in registers 5 and 6 are not equal, branch 2 instructions forward)
5   loadi 4 0x04   // r4 = 4
6   sub 4 4 0x05   // r4 = 4 - 5
7   srl 4 4 0x02   //(apply logical shift right 2 times on value in register 4, and place the result in register 4)
8   sll 4 4 0x02     //(apply logical shift left 2 times on value in register 4, and place the result in register 4)
9
10
```

```
kalindu@DESKTOP-R08BL8B:/mnt/d/PE19 Fourth Semester/CO224-Computer Architecture/Labs/Lab5/Part5New$ ./generate_memory_image.sh lab5_marking.s
Instruction memory content generated!
```

instr_mem.mem

```
1   00000101 00000000 00000101 00000000
2   00001001 00000000 00000110 00000000
3   00000101 00000110 00000100 00001101
4   00000010 00000110 00000101 00001100
5   00000100 00000000 00000100 00000000
6   00000101 00000100 00000100 00000011
7   00000010 00000100 00000100 00001111
8   00000010 00000100 00000100 00001110
9
```

```
PS D:\PE19 Fourth Semester\CO224-Computer Architecture\Labs\Lab5\Part5New> vvp .\test
WARNING: .\cpu_tb.v:36: $readmemb(instr_mem.mem): Not enough words in the file for the requested range [0:1023].
VCD info: dumpfile cpu_wavedata.vcd opened for output.
                TIME     R0      R1      R2      R3      R4      R5      R6      R7
                   5      0       0       0       0       0       0       0       0
                  13      0       0       0       0       0       5       0       0
                  21      0       0       0       0       0       5       9       0
                  29      0       0       0       0      45       5       9       0
                  45      0       0       0       0      40       5       9       0
                  53      0       0       0       0      10       5       9       0
                  61      0       0       0       0      40       5       9       0
.\cpu_tb.v:62: $finish called at 305 (1s)
PS D:\PE19 Fourth Semester\CO224-Computer Architecture\Labs\Lab5\Part5New>
```



It can be clearly observed from the diagram that ;

Since r5 != r6 the bne instruction has executed. Therefore, 2 instructions has branch forwaord after the 4 th instruction(bne) in the diagram.

Timing diagram details :

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		

Figure 5: Timing details for BNE instruction data path