WIJERATHNA I.M.K.D.I. : E/19/446

DISSANAYAKE D.M.I.G. : E/19/090

GROUP 13

SEMESTER 4

01/06/2023

CO 224 – LAB 5

BUILDING A SIMPLE PROCESSOR

**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 :**

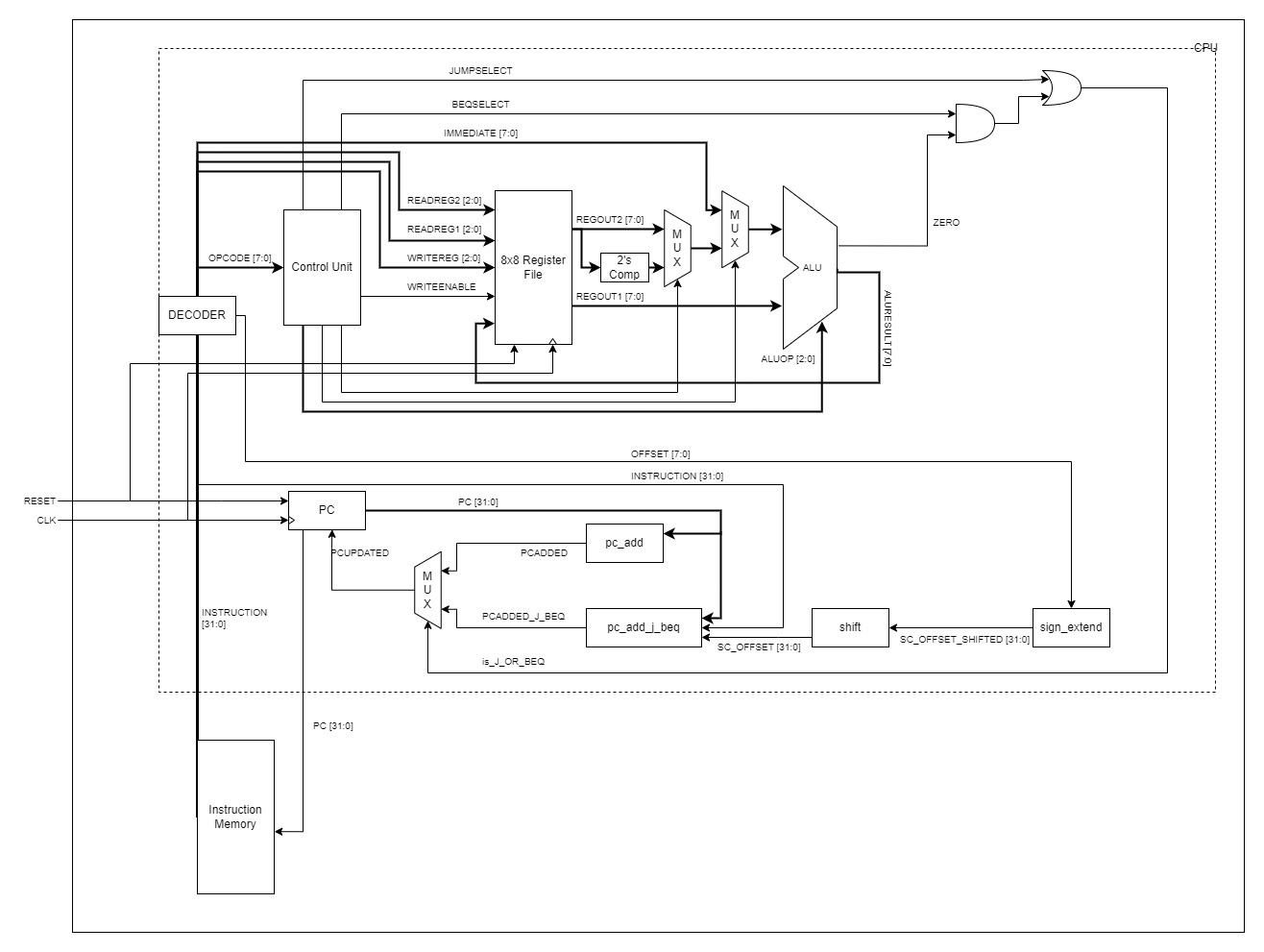
****

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.

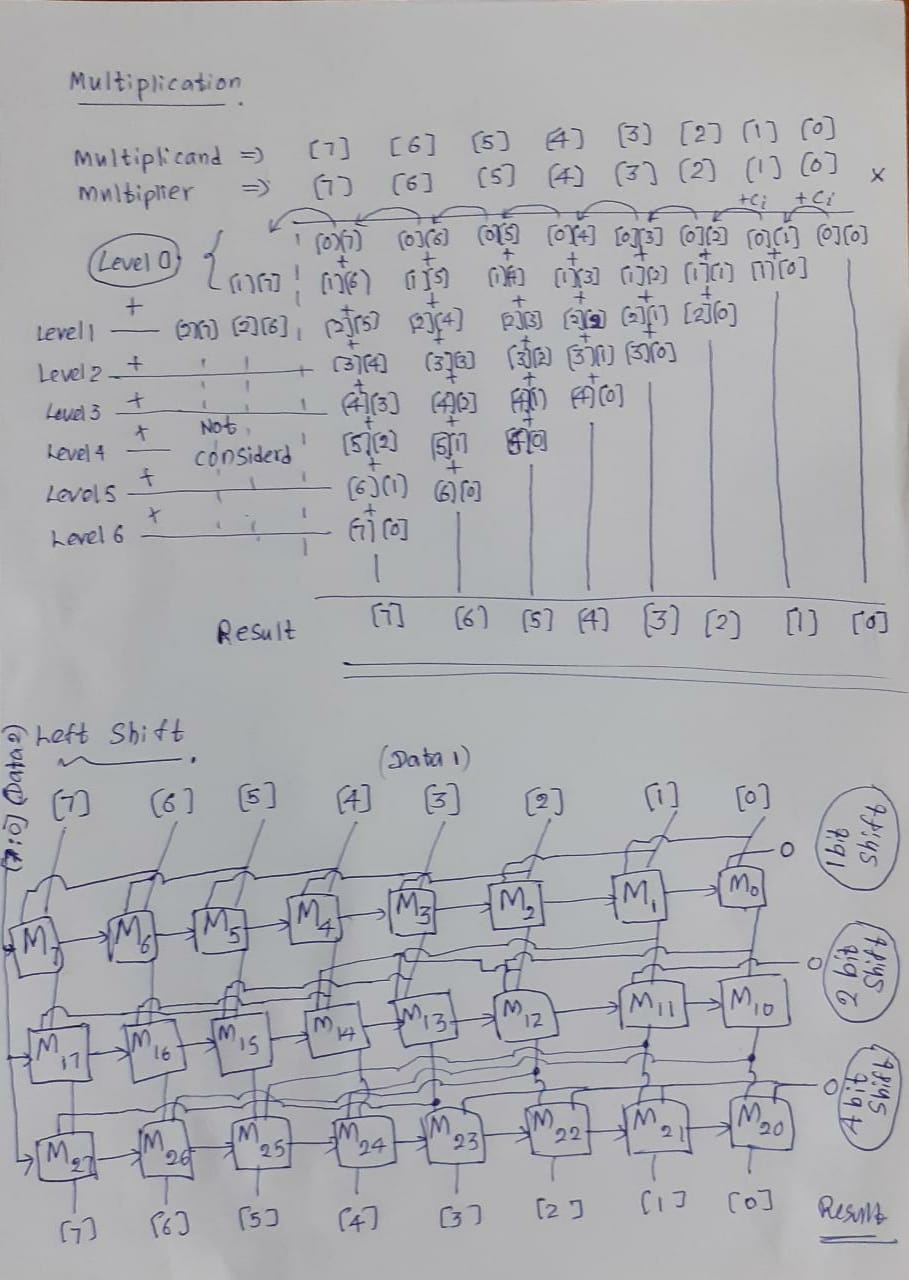
**Additional Instruction 1: mult**

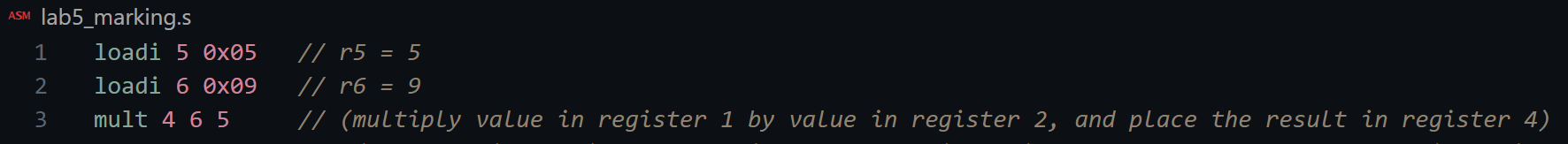
Operation : Multiplication (Result= DATA1 \* DATA2)

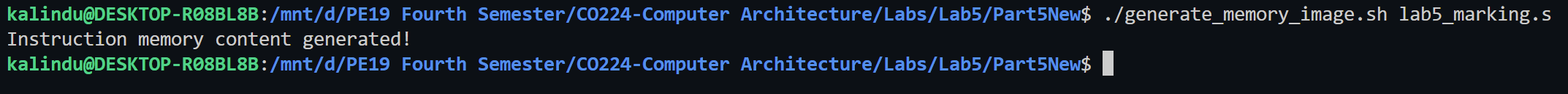
ALUOP : 110

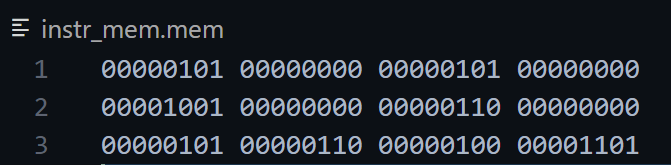
Time delay : 3 time units (because of 7 levels of Full adders used)

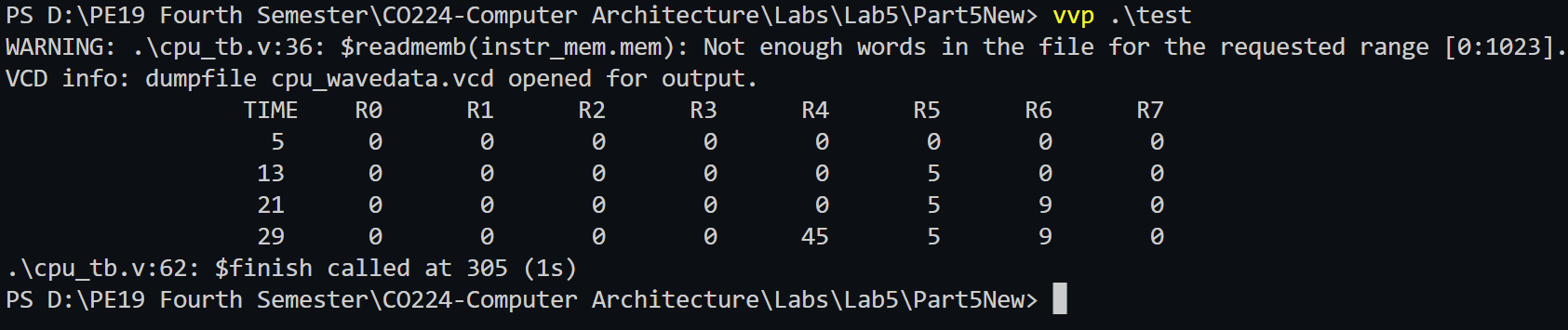
Design of the mult unit:

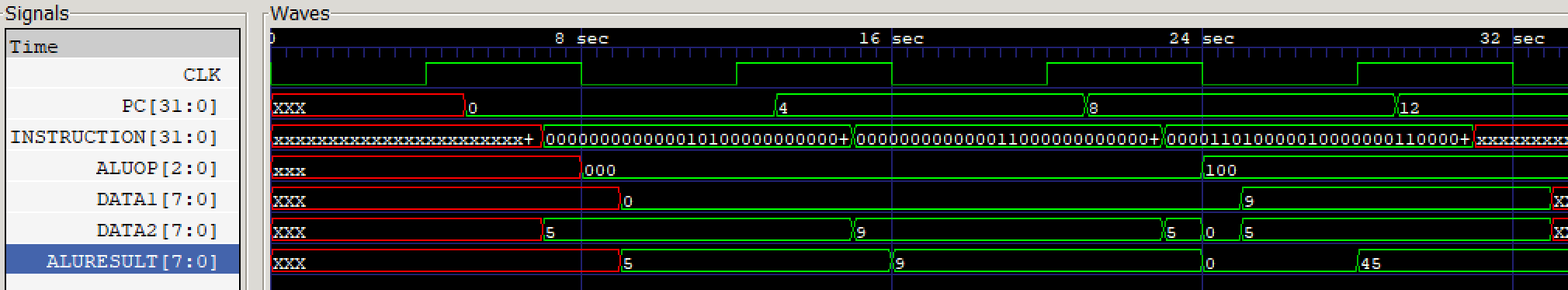


Implementing the code with gtkwave :

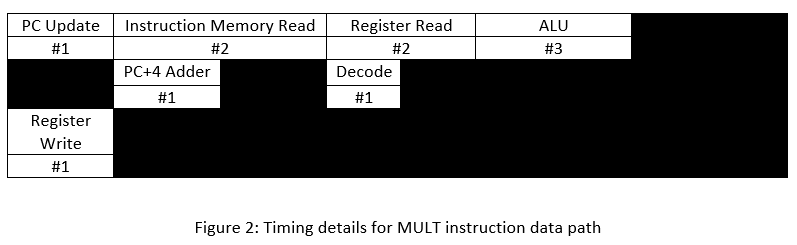






****

Timing diagram details :



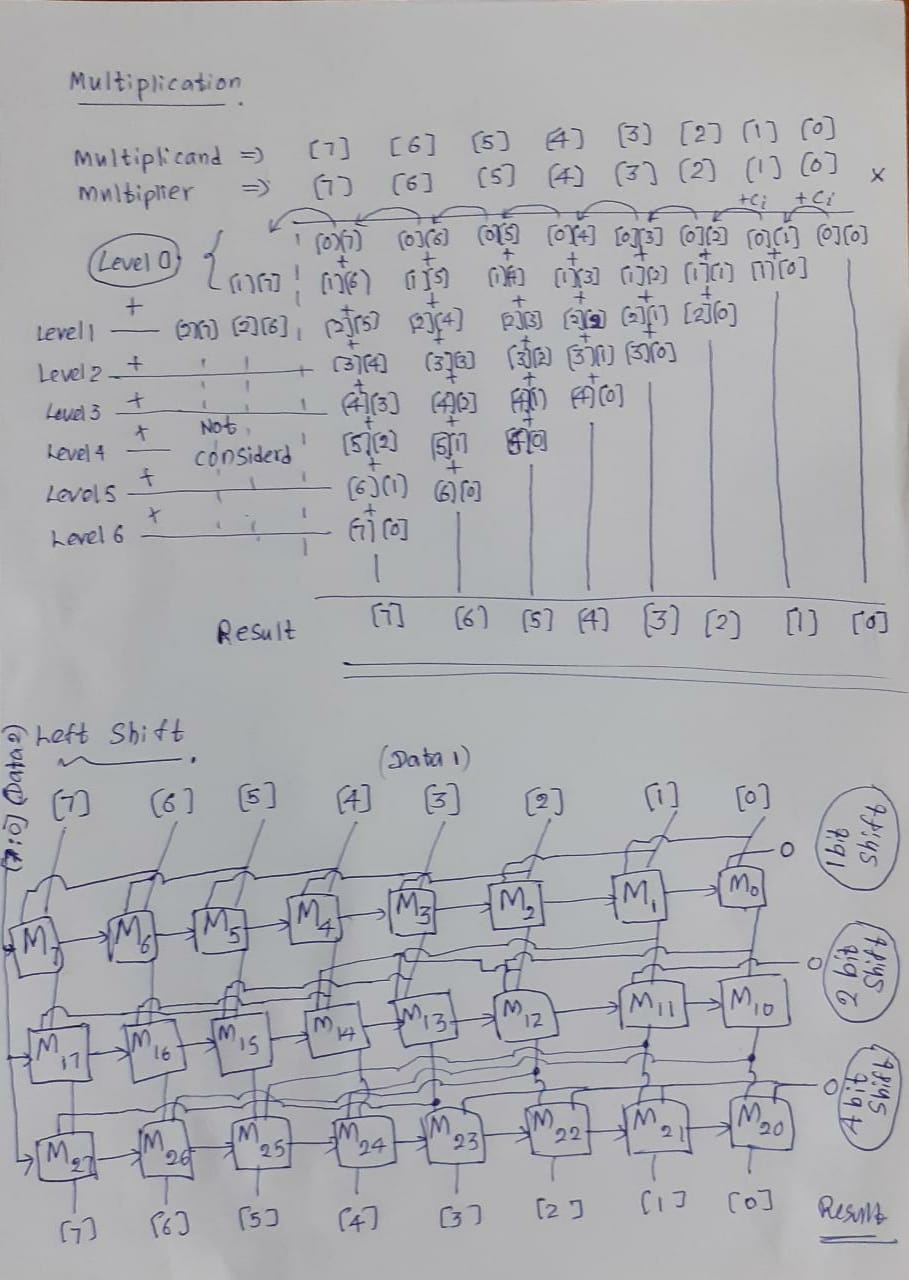
**Additional Instruction 2: sll**

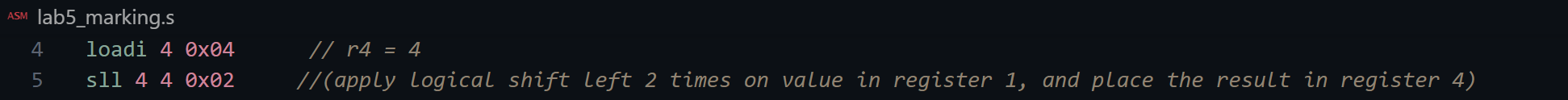
Operation : Logical Shift Left (Result = DATA1 << DATA2 )

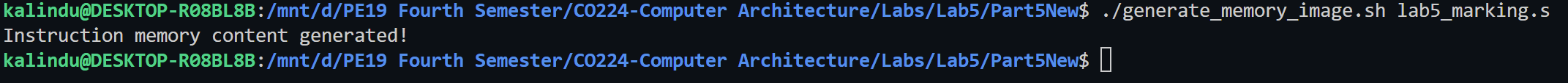
ALUOP : 101

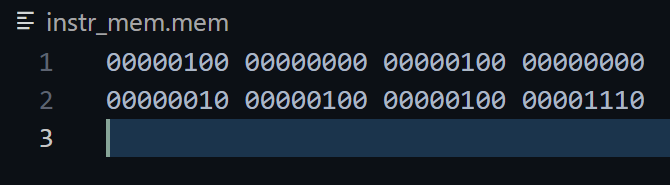
Time delay : 2 time units (because of 2-3 levels of Muxes used)

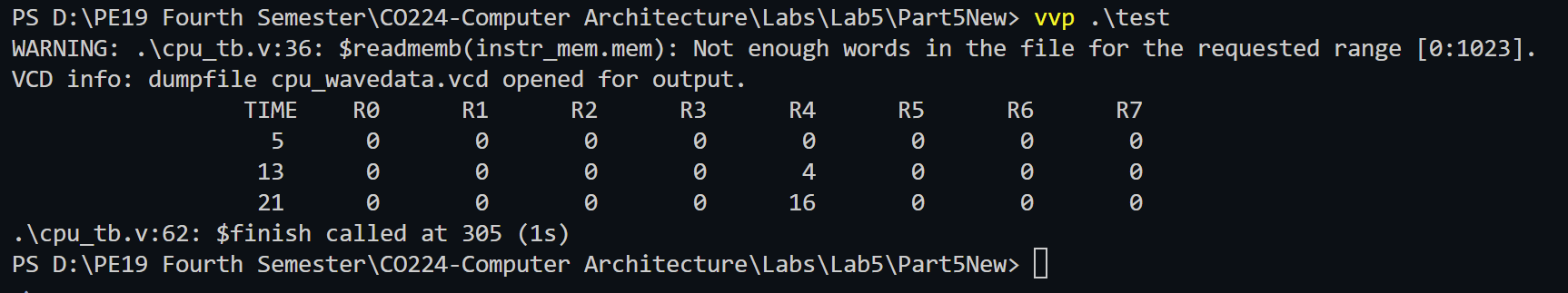
Design of the sll unit:

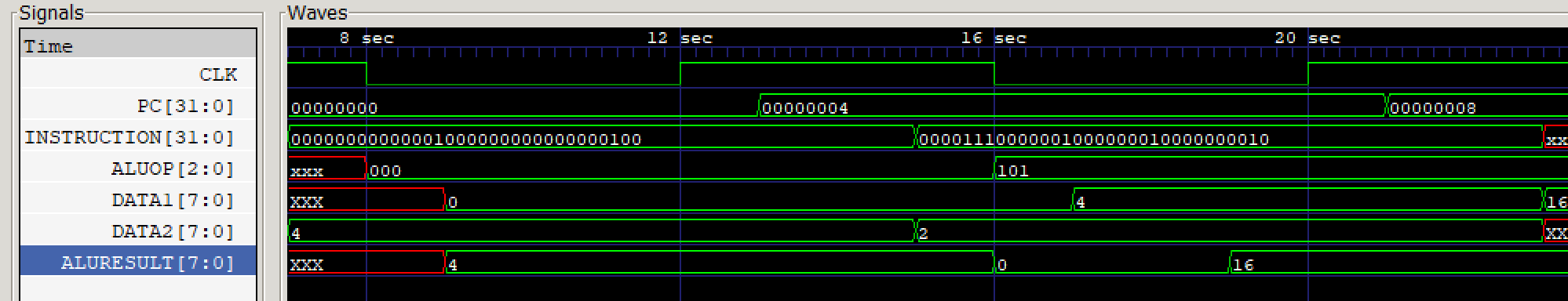


Implementing the code with gtkwave :

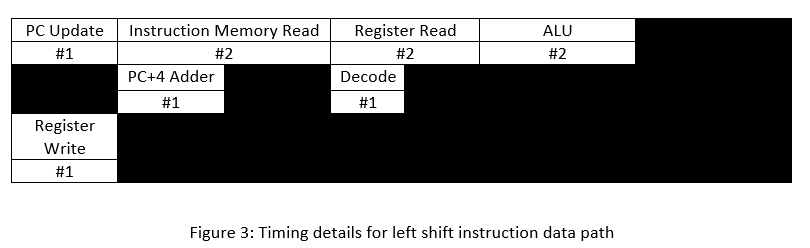








Timing diagram details :



**Additional Instruction 3: slr**

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

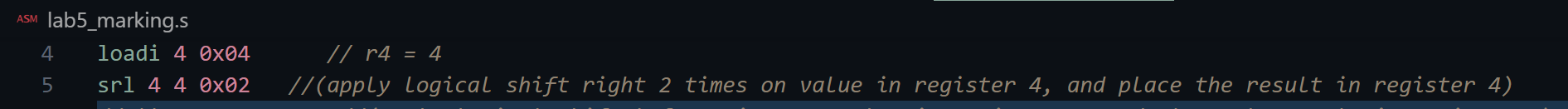
ALUOP : 110

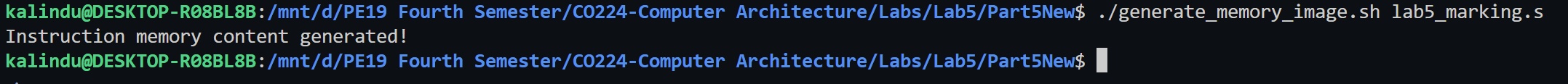
Time delay : 2 time units (because of 2-3 levels of Muxes used)

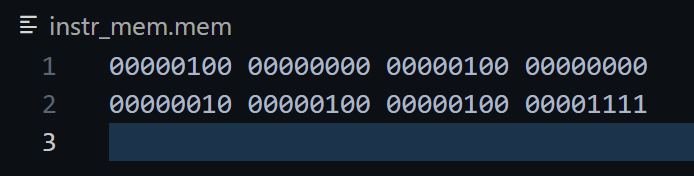
Design of the slr unit:

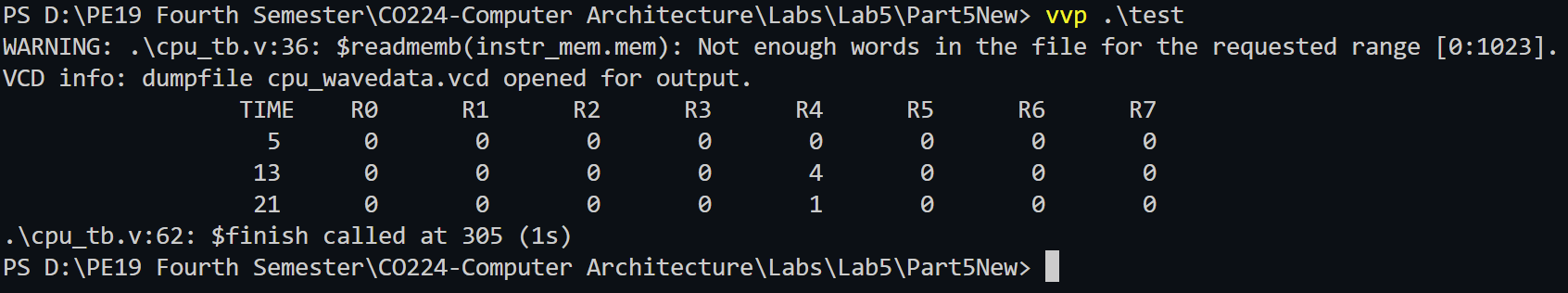
*Similar to the design of sll unit.*

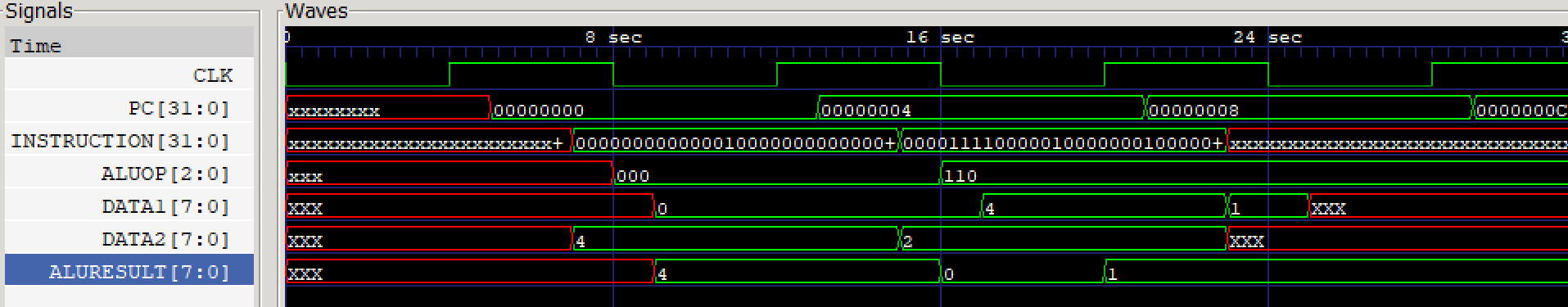
Implementing the code with gtkwave :



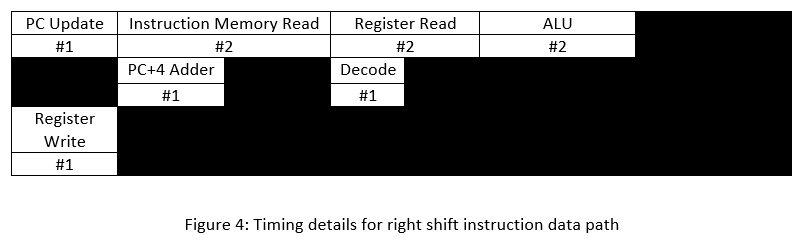








Timing diagram details :

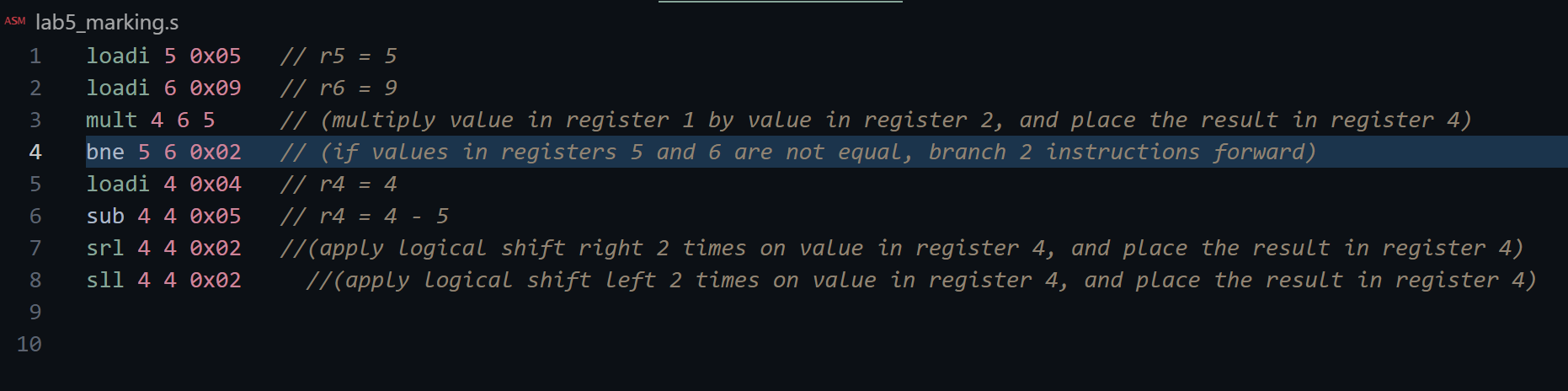


**Additional Instruction 4: bne**

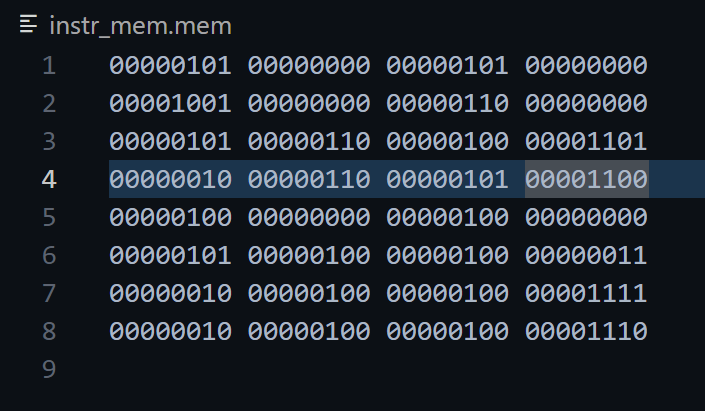
Operation : Branch Not Equal

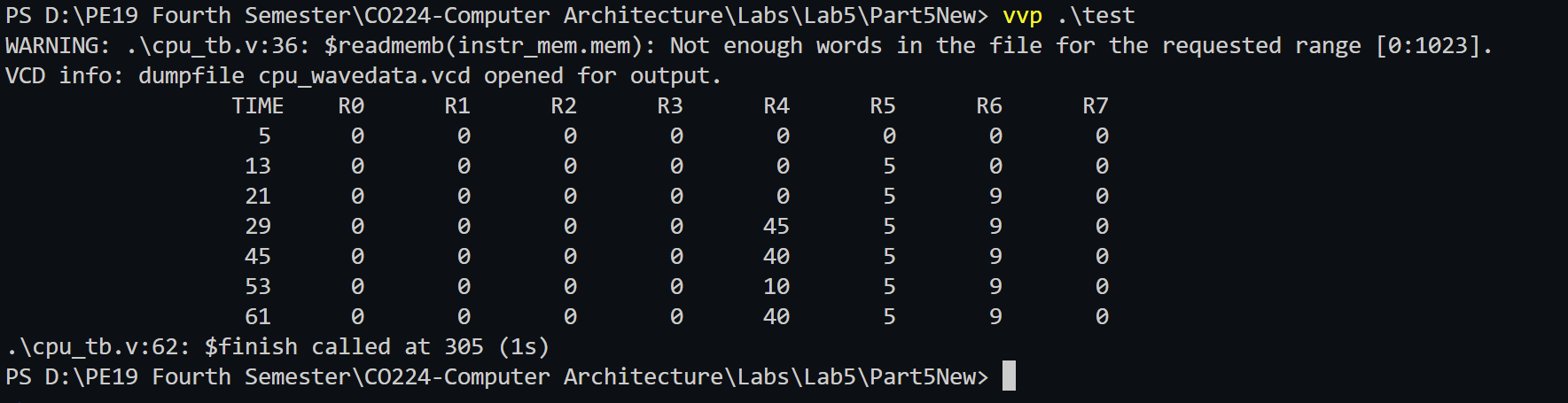
ALUOP : 001

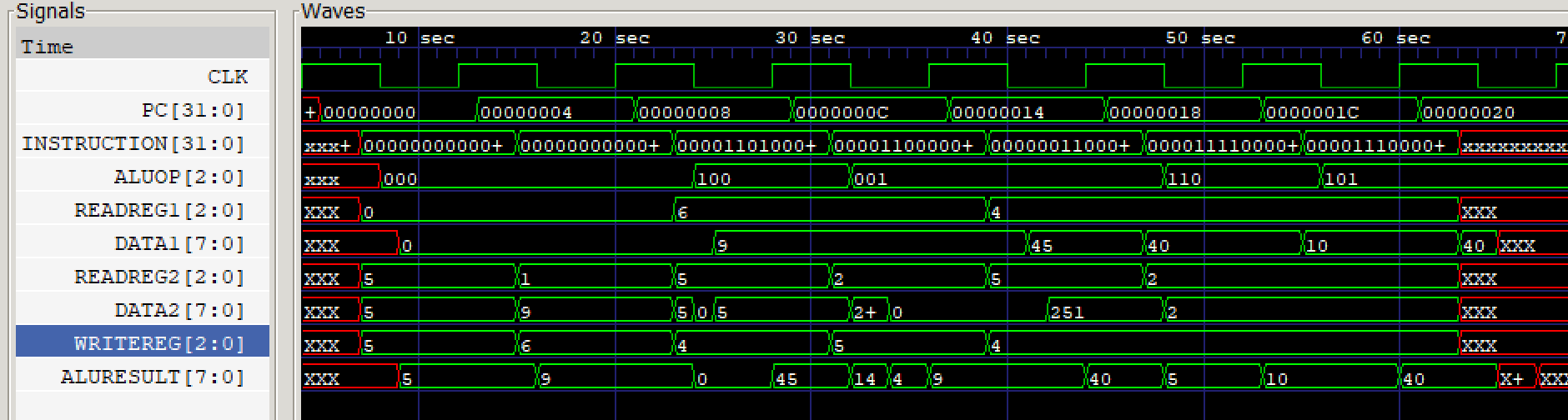
OPCODE : 00001100

Implementing the code with gtkwave :









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 :

