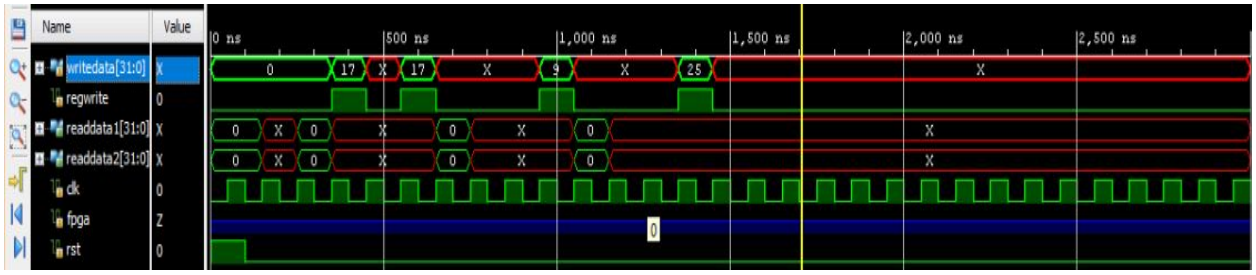
Name : salma soliman
Id : 900182325
Lab7 Monday 3:30pm

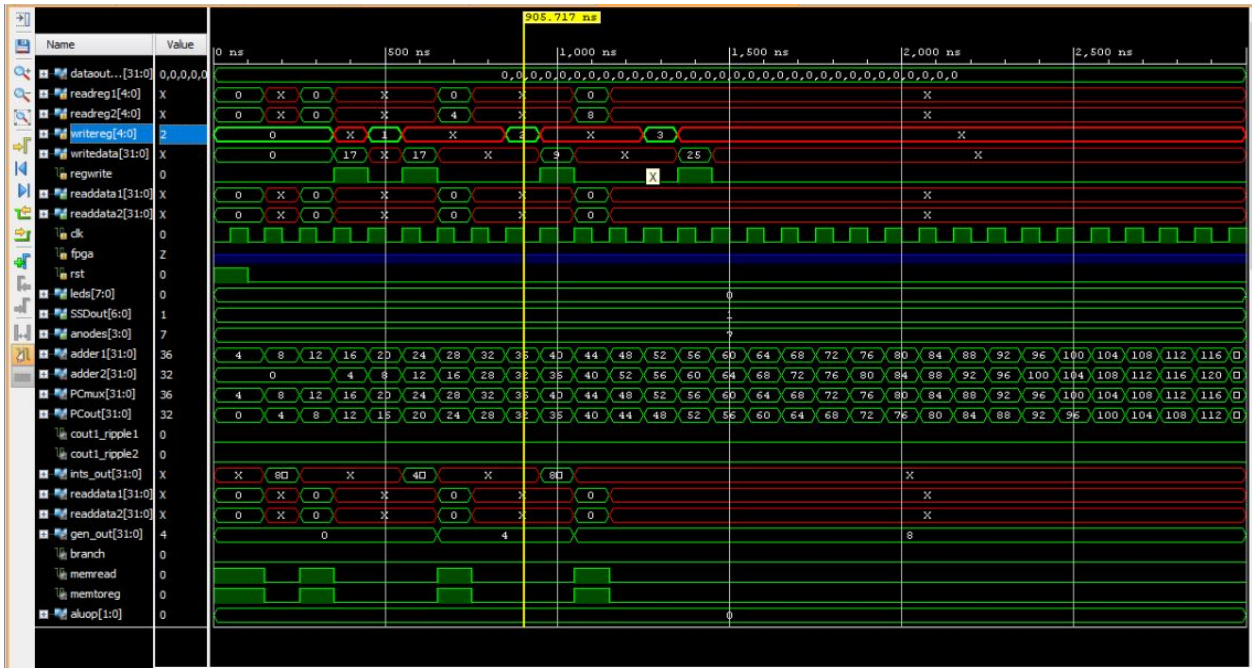
Experiment (1):
All verilog codes are in the file attached.

Experiment (2):
Data memory signals are working correctly and their output is 17,9 and 25 as expected.

The output of the mux (write data) is working correctly. Its output is 17,9 and 25 as expected.



All processor signals :



The table of the expected values of the program :

instructions	PC output	PC input	writedata of the register file	writedata of the data memory
`add x0, x0, x0`	0	4	0	0
`lw x1, 0(x0)`	4	8	17	17
`lw x2, 4(x0)`	20	24	9	9
`lw x3, 8(x0)`	36	40	25	25
`or x4, x1, x2`	52	56	25	0
`beq x4, x3, 16`	16*2+68 = 100	104	0	0
`add x3, x1, x2`	Skip 84	Skip 88	0	0
`add x5, x3, x2`	100	104	25+9=34	0
`sw x5, 12(x0)`	116	120	0	0
`lw x6, 12(x0)`	132	136	34	34
`and x7, x6, x1`	148	152	34&&17= 0	0
`sub x8, x1, x2`	164	168	17-9= 8	0
`add x0, x1, x2`	180	184	17+9= 26// (0) as its x0	0
`add x9, x0, x1`	196	200	0+17= 17	0

The program before modifying :

```
add x0, x0, x0
lw x1, 0(x0)
add x0, x0, x0
add x0, x0, x0
add x0, x0, x0
lw x2, 4(x0)
add x0, x0, x0
add x0, x0, x0
add x0, x0, x0
lw x3, 8(x0)
add x0, x0, x0
add x0, x0, x0
add x0, x0, x0
or x4, x1, x2
add x0, x0, x0
add x0, x0, x0
add x0, x0, x0
beq x4, x3, 16
add x0, x0, x0
add x0, x0, x0
add x0, x0, x0
add x3, x1, x2
add x0, x0, x0
add x0, x0, x0
add x0, x0, x0
```

```
add x5, x3, x2
add x0, x0, x0
add x0, x0, x0
add x0, x0, x0
sw x5, 12(x0)
add x0, x0, x0
add x0, x0, x0
add x0, x0, x0
lw x6, 12(x0)
add x0, x0, x0
add x0, x0, x0
add x0, x0, x0
and x7, x6, x1
add x0, x0, x0
add x0, x0, x0
add x0, x0, x0
sub x8, x1, x2
add x0, x0, x0
add x0, x0, x0
add x0, x0, x0
add x0, x1, x2
add x0, x0, x0
add x0, x0, x0
add x0, x0, x0
add x9, x0, x1
```

1) The NOP instructions that are marked in red can be cancelled as we are already writing on X0 which will be zero always so we don't need to wait for the writing stage to be completed.
2) The NOP instructions that are marked in green can be cancelled as I don't need X8 or X7 for any further operations.
3) The NOP instructions that are marked in yellow can be cancelled as I don't need them. We are only writing in different registers.
4) The NOP instructions that are marked in pink can be cancelled as I don't need them. This add instruction will be skipped due to the beq instruction.
5) Also the AND instruction that depends on the lw can be moved to the end of the program so that the lw doesn't need NOP anymore.

   The type of hazards that caused NOPs to be added:
   - Data hazards in all NOP except after beq.
   - Control hazards after the beq instruction only.

The modified program of lab6:

```
.data

.text
main:


lw t4, 0(zero)
lw t5, 4(zero)
lw t6, 8(zero)
NOP(data hazard)
NOP
NOP
loop:
beq t4,t6,exit
NOP (control hazard)
NOP
NOP
add  t4,t4,t5
beq zero,zero,loop
NOP (control  hazard)
NOP
exit:
sw t4, 12(zero)
```