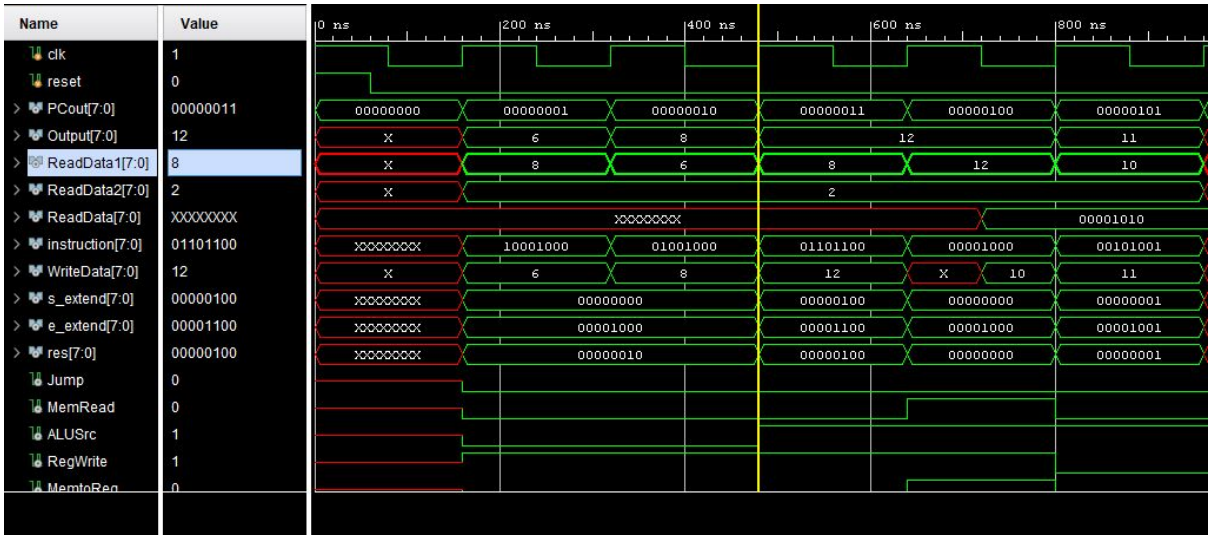
Overview of instructions being executed:

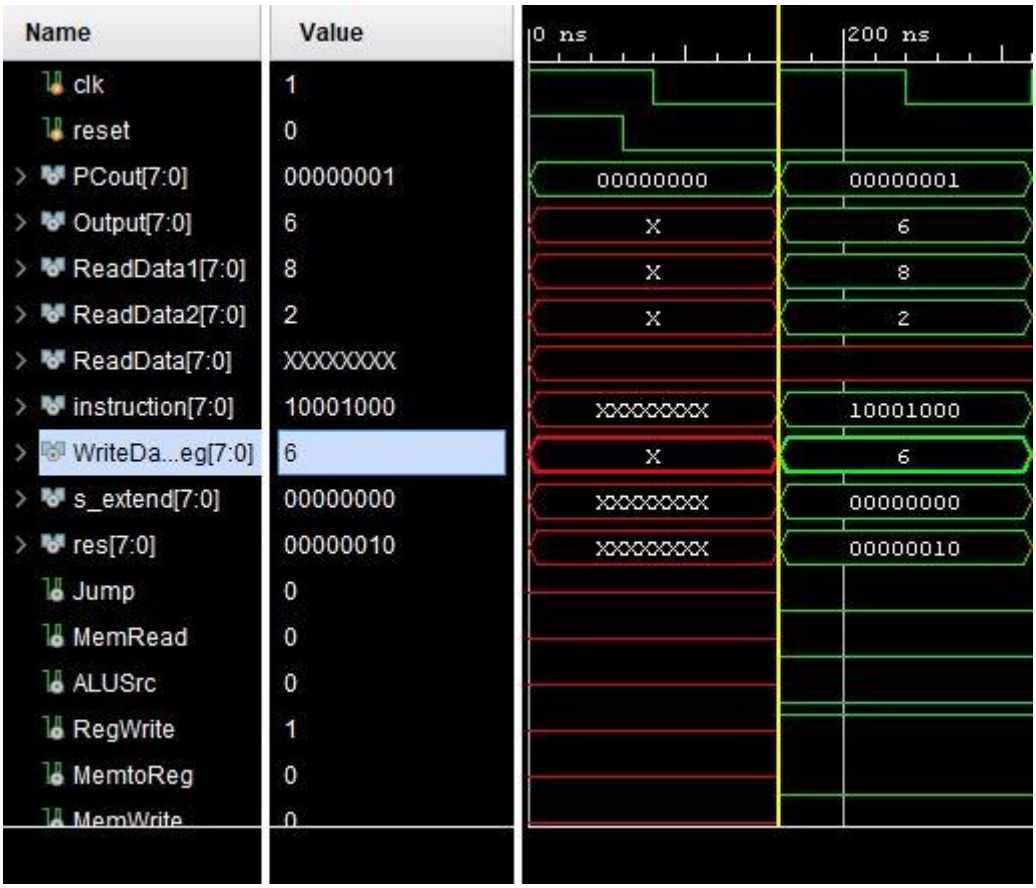
1. Executing add, sub, addi, lw & sw together in a loop

- The executing starts when PCin is 8'b1. The first instruction is "sub r1, r2"
- Registers r1 and r2, initially have values 8 and 2 respectively.
- As the instruction (10001000) executes, 6 is stored back as result in r1 since we are considering r1 as the destination. This concept is followed for every other instruction.
- The WriteData_Reg value (6) indicates that we need to store the value into r1 as a result.
- The next instruction (01001000) is executed with updated value of r1 as 6 and r2 remaining 2. Since it is an add instruction, the Output (ALU Output) is 8. Again, the WriteData_Reg value (8) indicates that we need to store the value 8 into r1 as a result.
- Similarly, the addition immediate instruction (01101100) is executed. The lower 3 bits are sign extended to 8 bits (00000100). Then the value of r1 and sign extended bits is added. (8 + 4 = 12). Then the value 12 is stored as a result in r1.
- Similarly, load instruction (00001000) executes. The final 3-bits 000 are extended to 8-bit value and then the ALU performs the addition of r1 and the sign extended bits [(12 + 0) = 12]. This value 12 will now be used as an address and DataMemory [12] has a value of 10 which is declared in the DataMemory module. Thus, value 10 will be stored in the RegisterFile (r1).
- When the next instruction store (00101001) is executed, r1 has the value 10 and r2 remains 2. The final 3 (001) bits are sign extended to 8 bits and added to ReadData1 register and the ALU will add them. The Output of the ALU will be considered as an address i.e (10 + 1 = 11) and DataMemory [11] will store the value of the register.



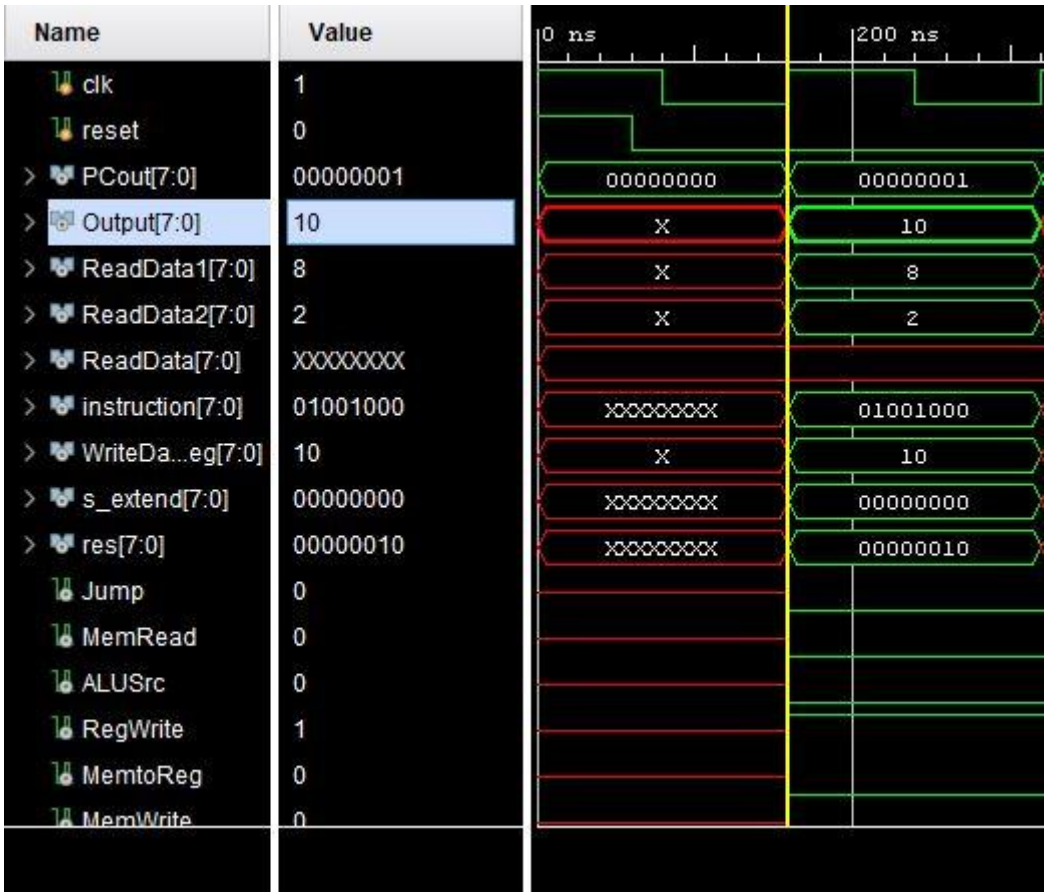
2. Executing sub instruction

- Instruction is (10001000). The two registers have the value 8 and 2 respectively. The result (ALU Output) is 6.



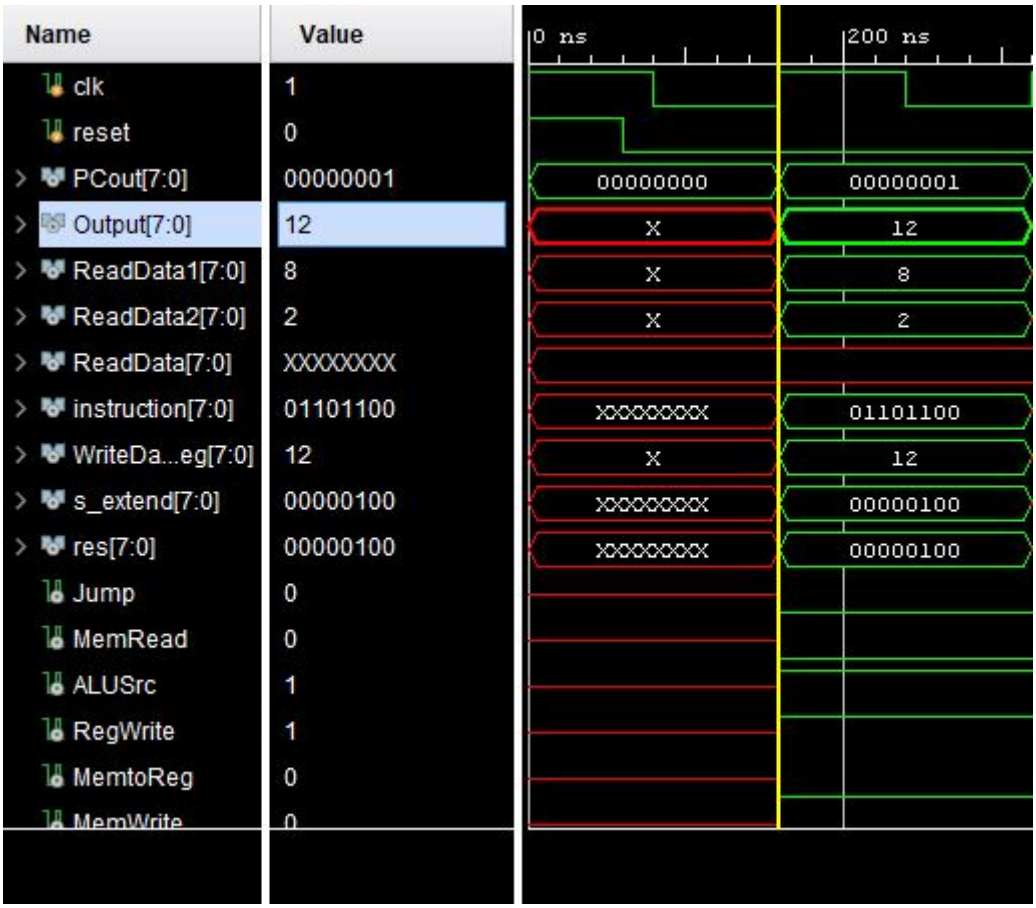
3. Executing add instruction

- Instruction is (01001000). The two registers have the value 8 and 2 respectively. The result (ALU Output) is 10.



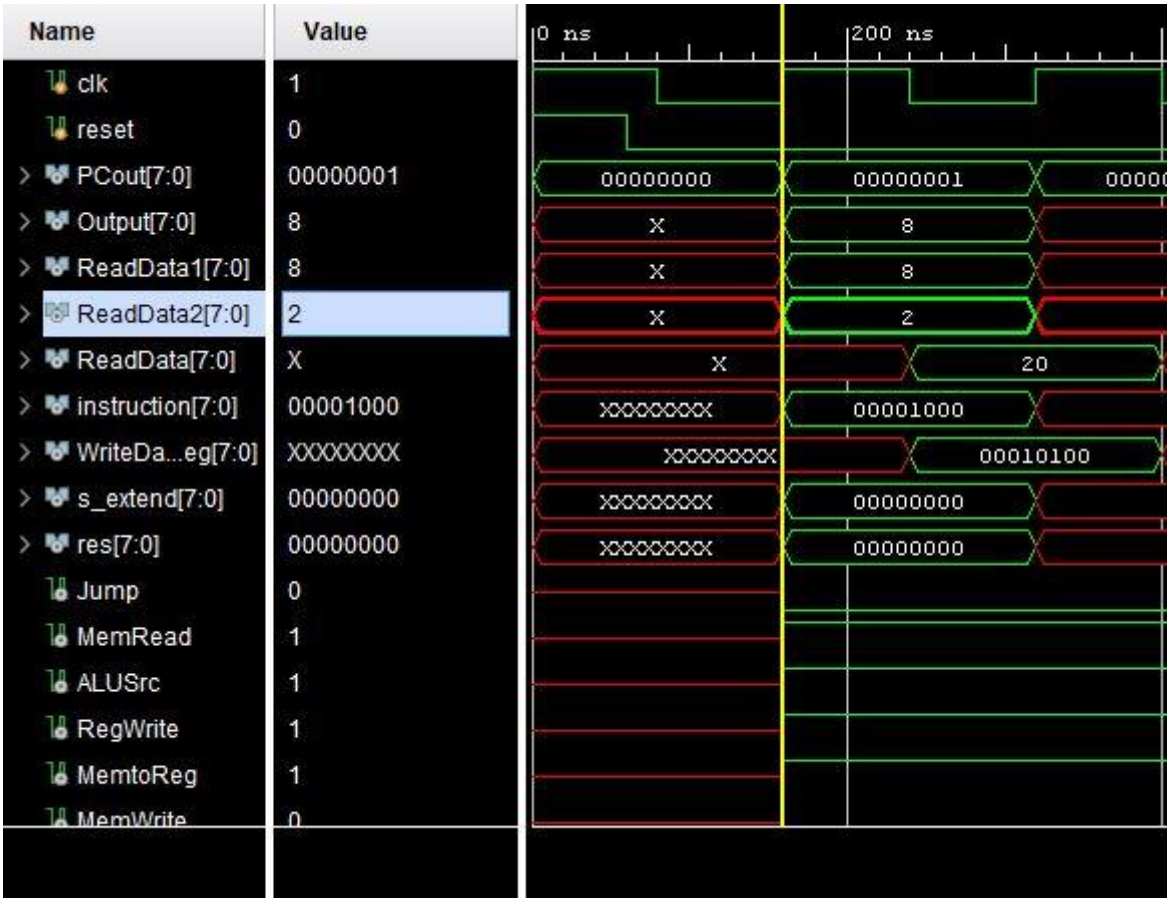
4. Executing addi instruction

- Instruction is (01101100). The register r1 has the value 8. The sign extened lower 3 bits correspond to a value 4. Thus the result is 12.



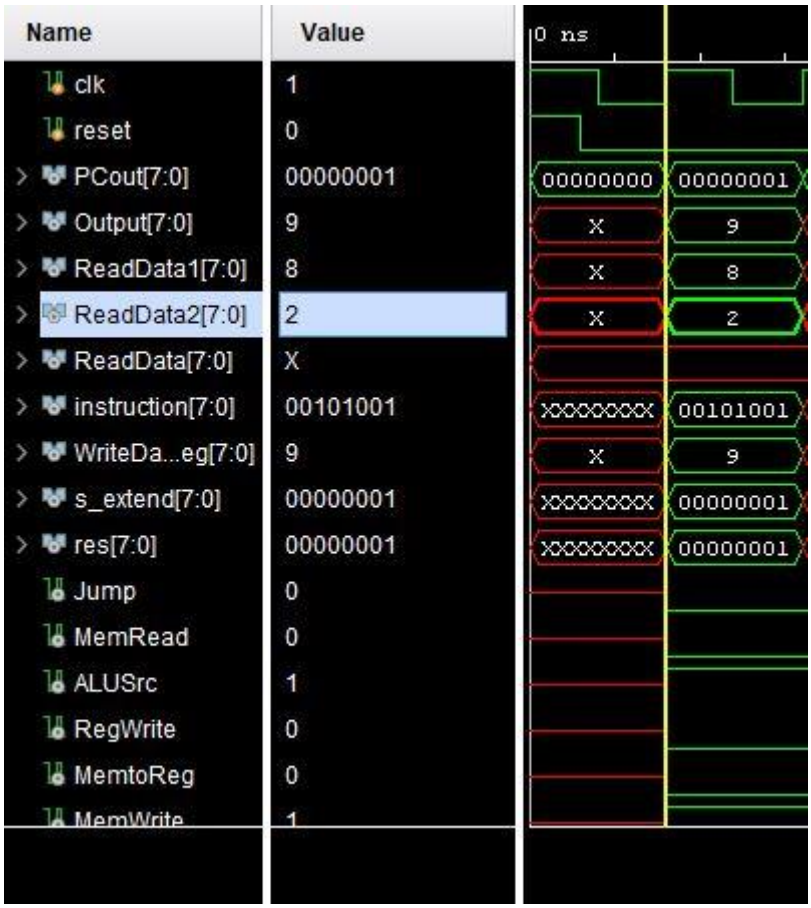
5. Executing lw instruction

- Instruction is (00001000). The final 3-bits 000 are extended to 8-bit value and then the ALU performs the addition of r1 and the sign extended bits [(8 + 0) = 8]. This value 8 will now be used as an address and DataMemory [8] has a value of 20 which is declared in the DataMemory module. Thus, value 20 will be stored in the RegisterFile (r1) which is indicated by WriteData_Reg. The value that is read from the DataMemory is 20 indicated by ReadData.



6. Executing sw instruction

- The instruction is (00101001). r1 has the value 8 and r2 is 2. The final 3 (001) bits are sign extended to 8 bits and added to ReadData1 register and the ALU will add them. The Output of the ALU will be considered as an address i.e (8 + 1 = 9) and DataMemory [9] will store the value of the register.



7. Executing jmp instruction

- The instruction is (1010000). The lower 5 bits are sign extended to 8 bits and the address generated is considered as the next instruction that needs to be executed.