The goal of this project was to design and implement a CPU capable of executing basic instructions. The CPU uses many components, including a ROM for program memory, RAM for data memory, and an ALU for arithmetic and logic operations. This project involved building a state machine to manage the CPU's operation and ensuring that the CPU could execute a set of instructions, including load, store, arithmetic operations, function calls, and stack operations.

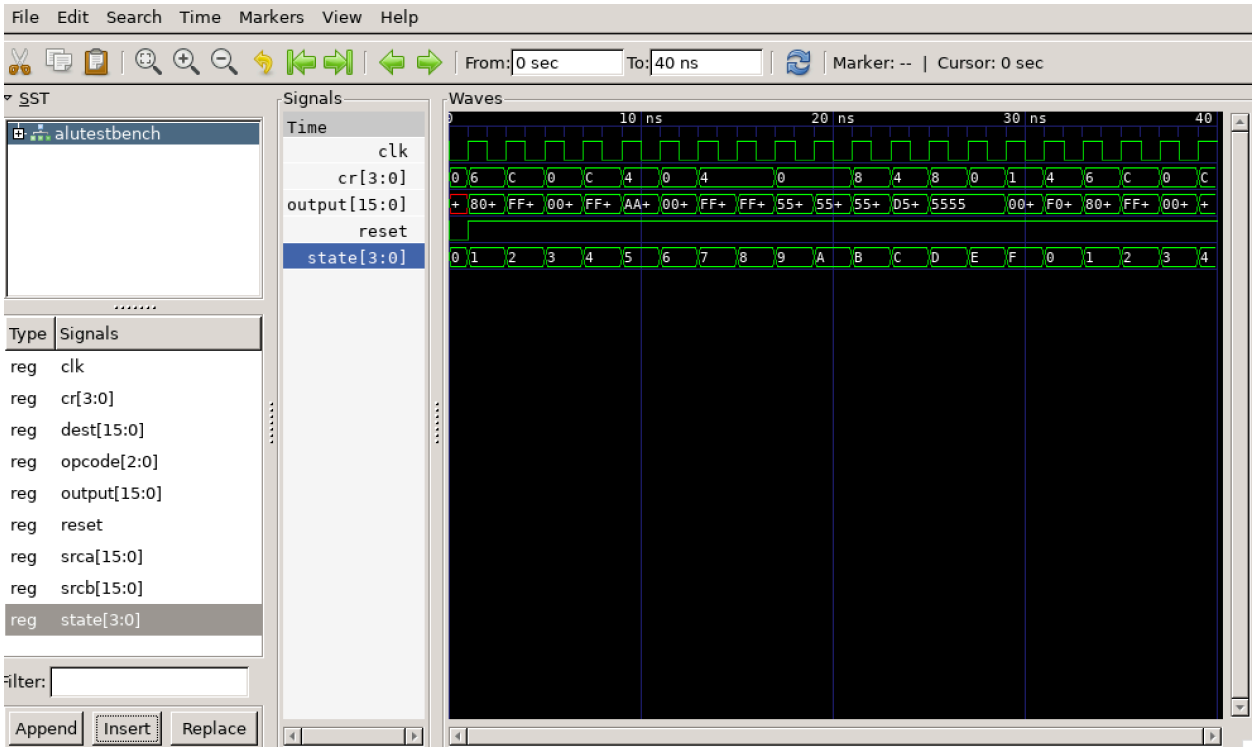

Image 1: Gtkwave for ALU

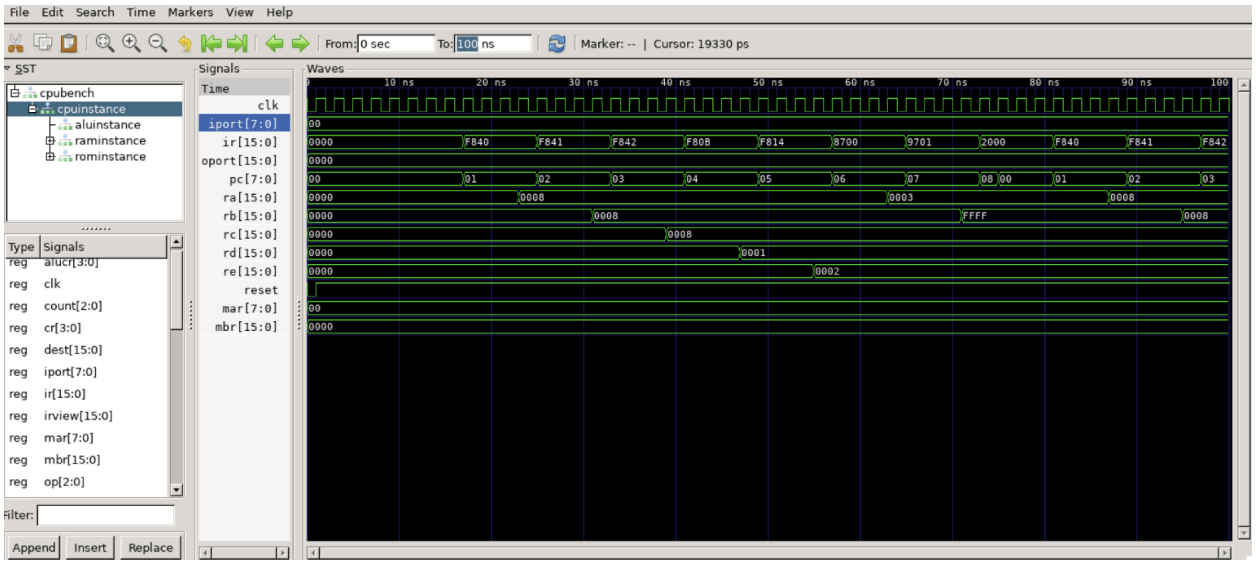

Image 2: Gtkwave for CPU test

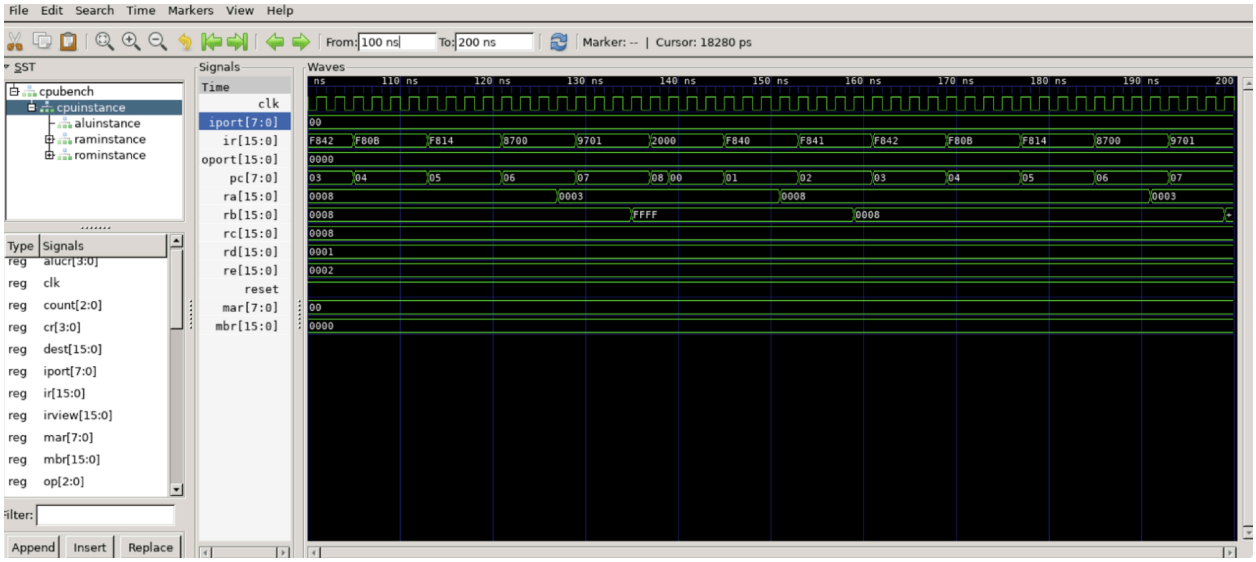

Image 3: Gtkwave for CPU test

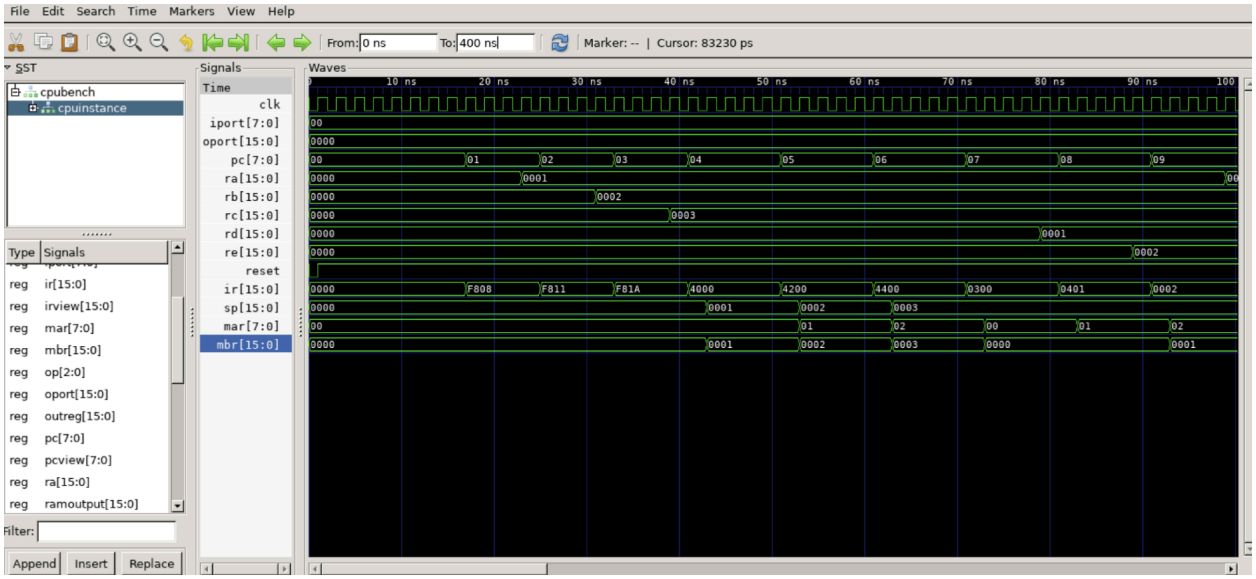

Image 4: Gtkwave for push test

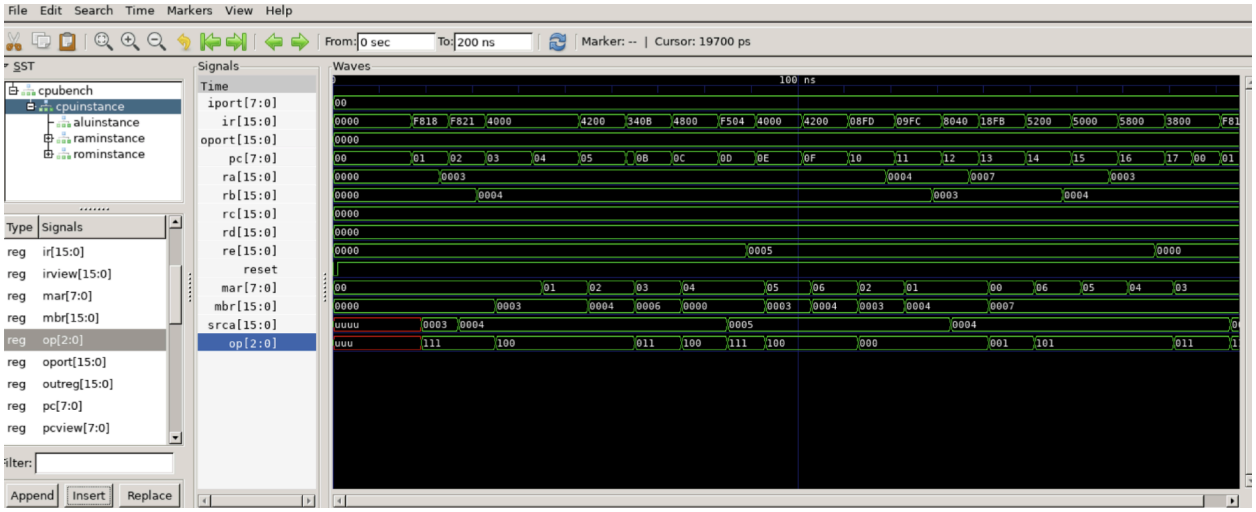

Image 5: Gtkwave for call test

fibonacci.mif

```
DEPTH = 256;
WIDTH = 16;
ADDRESS_RADIX = HEX;
DATA_RADIX = BIN;
CONTENT

BEGIN
-- Everything is moved to the RA, hence why it's the
-- only one selected in the shown GtkWave

00: 1111100000000000; -- move 0 into RA
01: 1111100000001000; -- move 1 into RA
02: 1111100000001000; -- same as above
03: 1111100000010000; -- move 2 into RA
04: 1111100000011000; -- move 3 into RA
05: 1111100000101000; -- move 5 into RA
06: 1111100001000000; -- move 8 into RA
07: 1111100001101000; -- move 13 into RA
08: 1111100010101000; -- move 21 into RA
09: 1111100100010000; -- move 34 into RA
[0A..FF]: 1111111111111111; -- end

END
```

Image 6: Machine code for Fibonacci

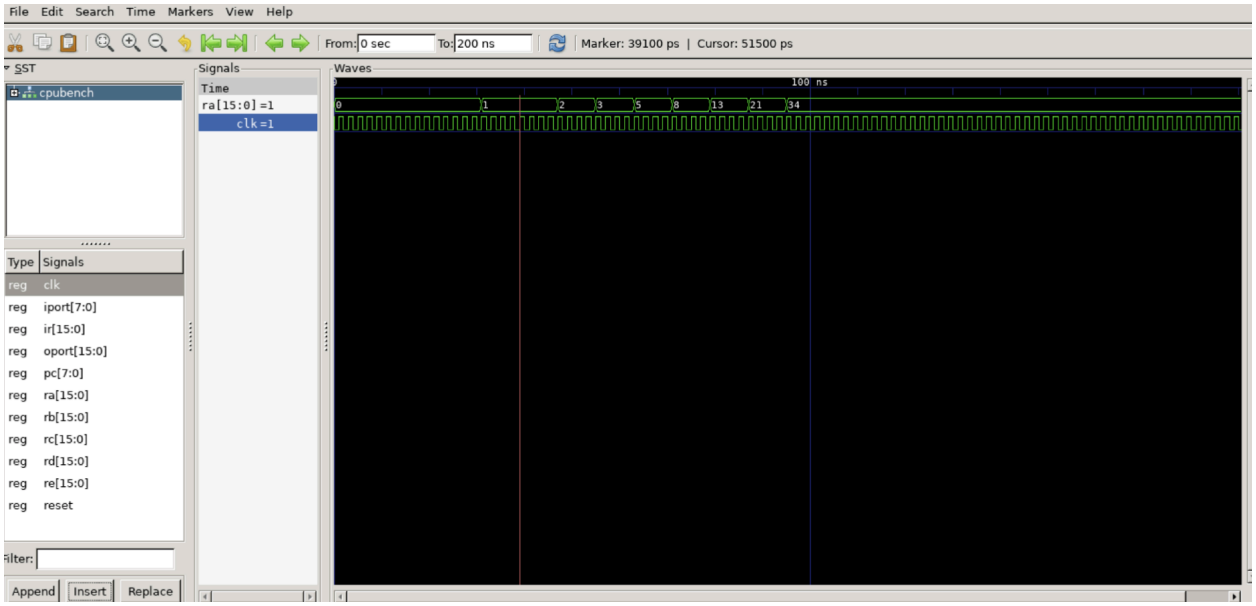

Image 7: Gtkwave for fibonacci sequence

Acknowledgements: I got help from online resources especially stack overflow and youtube videos.