UNIVERSITAT POLITÈCNICA DE CATALUNYA
BARCELONATECH
Facultat d'Informàtica de Barcelona

PROCESSOR ARCHITECTURE

# Final Project: Verilog design for a 32-bit RISC-V Processor

Sergio García Esteban

sergio.garcia@bsc.es

22/01/2023

## Index

## The Processor

In this project a processor has been designed and implemented from scratch. As initial decisions, Verilog as RTL language, ModelSim as RTL simulator, RISC-V as ISA and VisualStudioCode as programming environment are chosen.

The designed processor consists of 5 segmented stages (F, D, A, C, W), execution in order and 1 instruction launched per cycle.

The instruction set supported by the processor are the following RISC-V instructions: ADD, ADDI, SUB, SUBI, SLL, SLLI, SLR, SLRI, MUL, LDW, STW, BEQ, BNE.

The MUL instruction takes 5 cycles to execute, so the pipeline is lengthened in this case to F, D, M1, M2, M3, M4, M5, C, W.

The calculation of the jump instructions BEQ, BNE and JUMP is advanced to the D stage to reduce the performance loss at each jump.

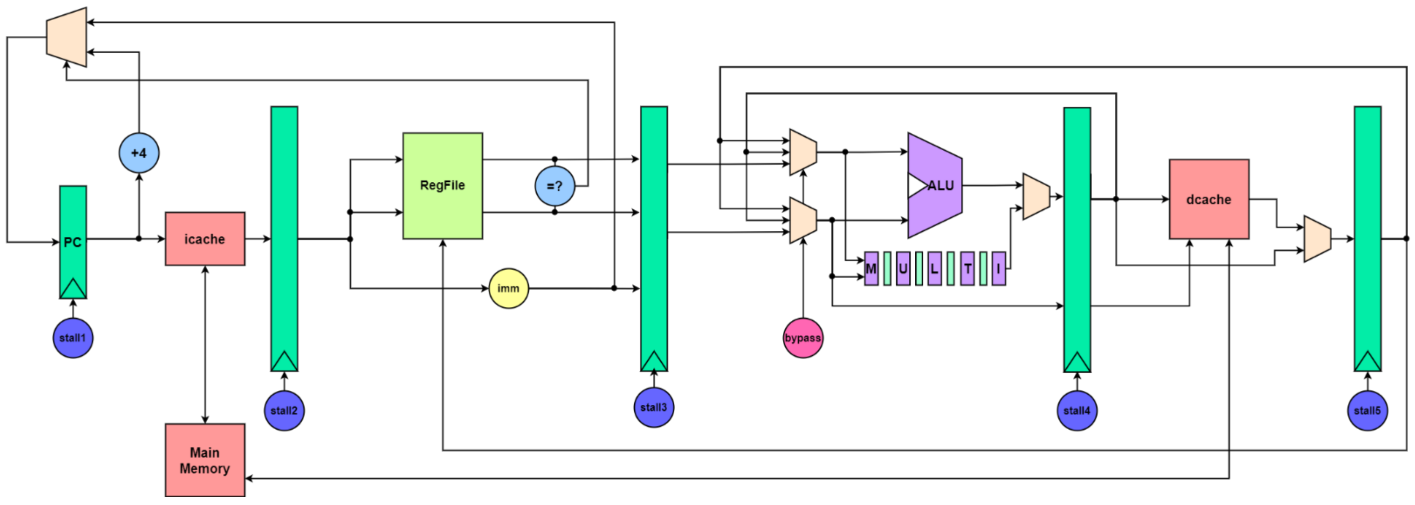
The processor includes the complete set of bypasses.

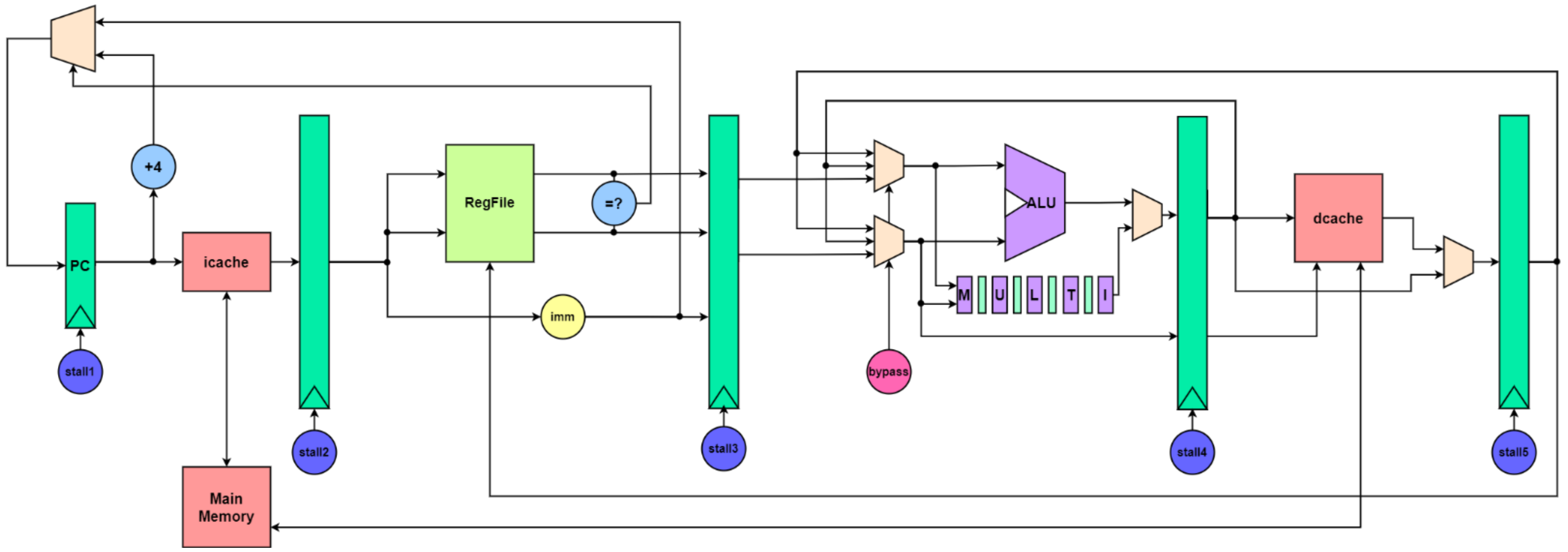
The processor includes Instruction Cache and Data Cache, both 4 lines of 128b each, Direct Mapped. Each time one of the caches has to go to main memory it will take 10 cycles to go and return.

Only the raw processor pipeline has been developed, it does not include virtual memory or exceptions. Mechanisms such as Store Buffer or Reorder Buffer have neither been implemented.

The processor is capable of running the 3 proposed benchmarks, BufferSum, MemCopy and MatrixMultiply.

## Diagram





PC
stall1
+4
icache
Main Memory
stall2
RegFile
=?
imm
stall3
bypass
ALU
M
U
L
T
I
stall4
dcache
stall5

# Instructions

## Load Instruction

Only the LW instruction, 4 bytes aligned, is supported.

In F stage, instruction is read from the icache (10 stop cycles in case of miss). In D stage, the register that stores the address of the load is read. In stage A, the address and the offset (imm) are added. In stage C, the data stored in the calculated address is read (10 stop cycles in case of miss). In stage W the value is written in the destination register (rising edge).

## Store instruction

Only SW instruction is supported, 4 bytes aligned.

In F stage, instruction is read from the icache (10 stop cycles in case of miss). In D stage, the register that stores the address of the load is read. In stage A, the address and the offset (imm) are added. In stage C the data is stored in the calculated address (10 stop cycles in case of miss).

## Branch instruction

BEQ and BNE instructions are supported.

In F stage, instruction is read from the icache (10 stop cycles in case of miss). In D stage, registers (or bypass) are read and compared. If the jump is taken, the jump address (imm) is entered in PC.

## Arithmetic instruction

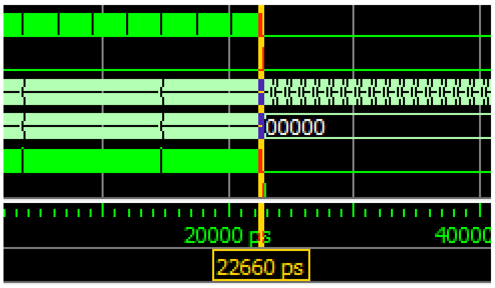
The ADD, ADDI, SUB, SUBI, SLL, SLLI, SLR, SLRI and MUL instructions are supported.

In the F stage, the instruction is read from the cache (10 stall cycles in case of a miss). In the D stage, the registers are read. In stage A, it performs the corresponding operation in the ALU, in the case of MUL, it will take 5 cycles to execute, delaying the rest of the instructions. In stage C nothing happens. In the W stage, the result of the ALU is written (rising edge).

## Performance

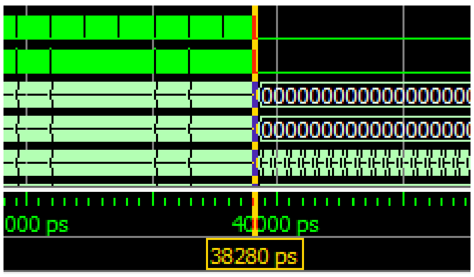
### Buffer Sum

The executed benchmark performs the sum of the 128 elements of a vector in memory by executing 128 loads and 1 store. The results show that it took 1133 cycles to execute the 645 instructions, resulting in an IPC of 0.57.



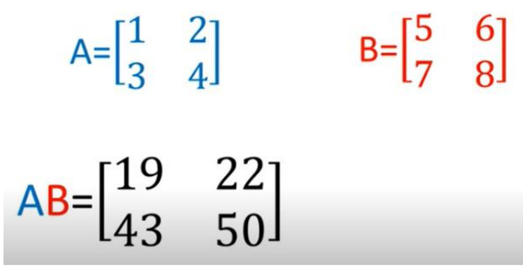
### Mem Copy

The executed benchmark performs the writing of the value 5 in the 128 elements of vector A to subsequently copy each element of vector A to vector B, executing 128 loads and 256 stores. The results show that it took 1914 cycles to execute the 1285 instructions, resulting in an IPC of 0.67.



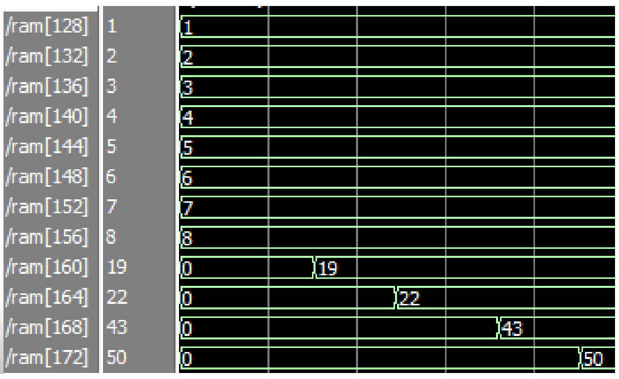
### Matrix Multiply

First of all, a test with 2x2 matrix has been run to check that the result is correct. Below is an image with the expected input and output matrices, and another image with the result of the simulation. We note that the results are correct and that it has been completed in 452 cycles.



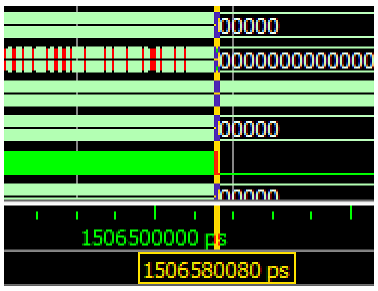
$$A=\begin{bmatrix}1 & 2\\3 & 4\end{bmatrix} \quad B=\begin{bmatrix}5 & 6\\7 & 8\end{bmatrix}$$

$$AB=\begin{bmatrix}19 & 22\\43 & 50\end{bmatrix}$$



```
/ram[128]  1   1
/ram[132]  2   2
/ram[136]  3   3
/ram[140]  4   4
/ram[144]  5   5
/ram[148]  6   6
/ram[152]  7   7
/ram[156]  8   8
/ram[160]  19  0 )19
/ram[164]  22  0 )22
/ram[168]  43  0 )43
/ram[172]  50  0 )50
```

Now, we run the proposed benchmark with 128x128 matrix. The benchmark consists of 29507977 instructions and the developed processor has executed it in 75329004 cycles, which results in an IPC of 0.39.


00000
0000000000000
00000
00000
1506500000 ps
1506580080 ps