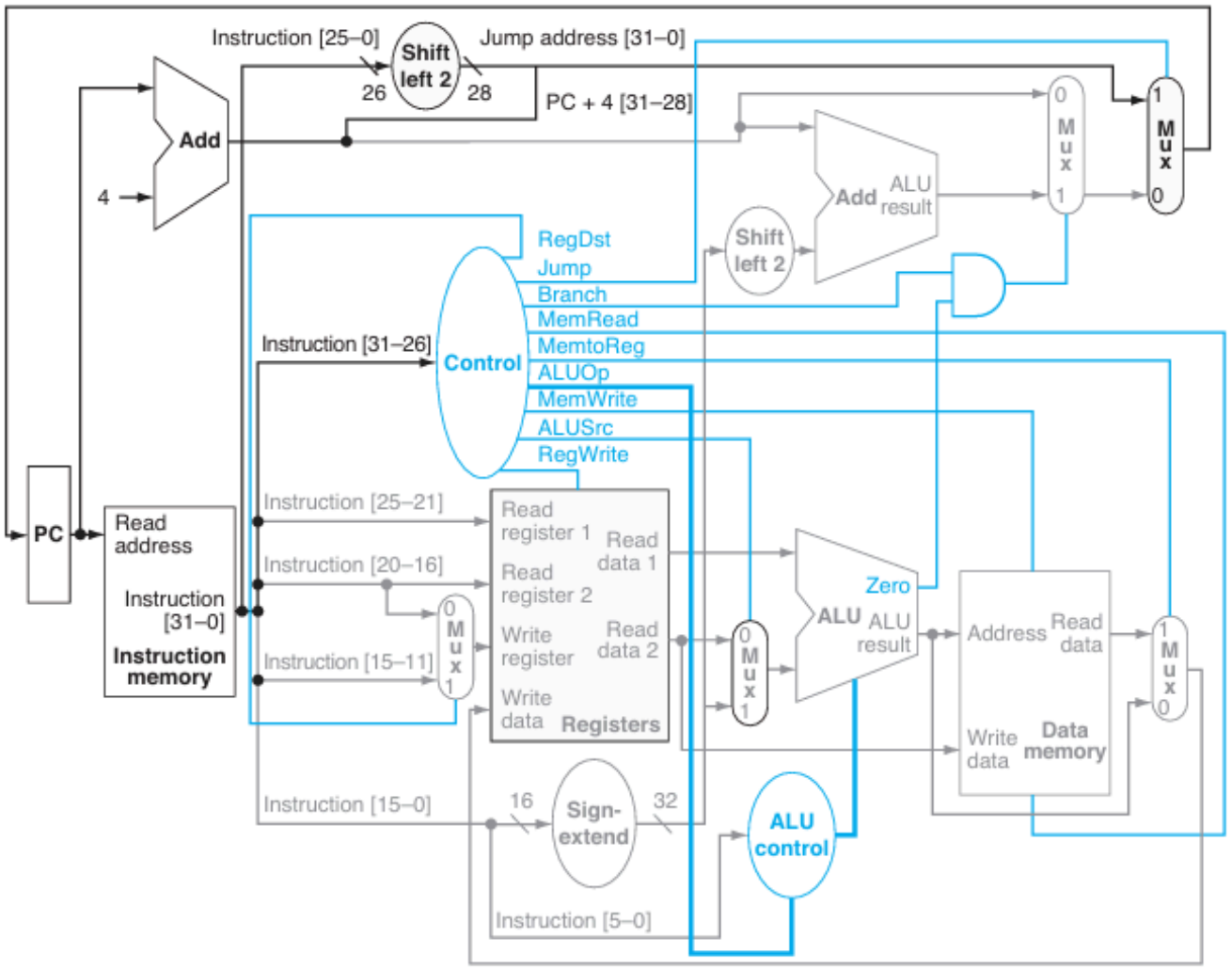
# MIPS Single Cycle Project


Designed by Yaniv Cohen Zagouri

Github




*Schematic 1: MIPS shematic*

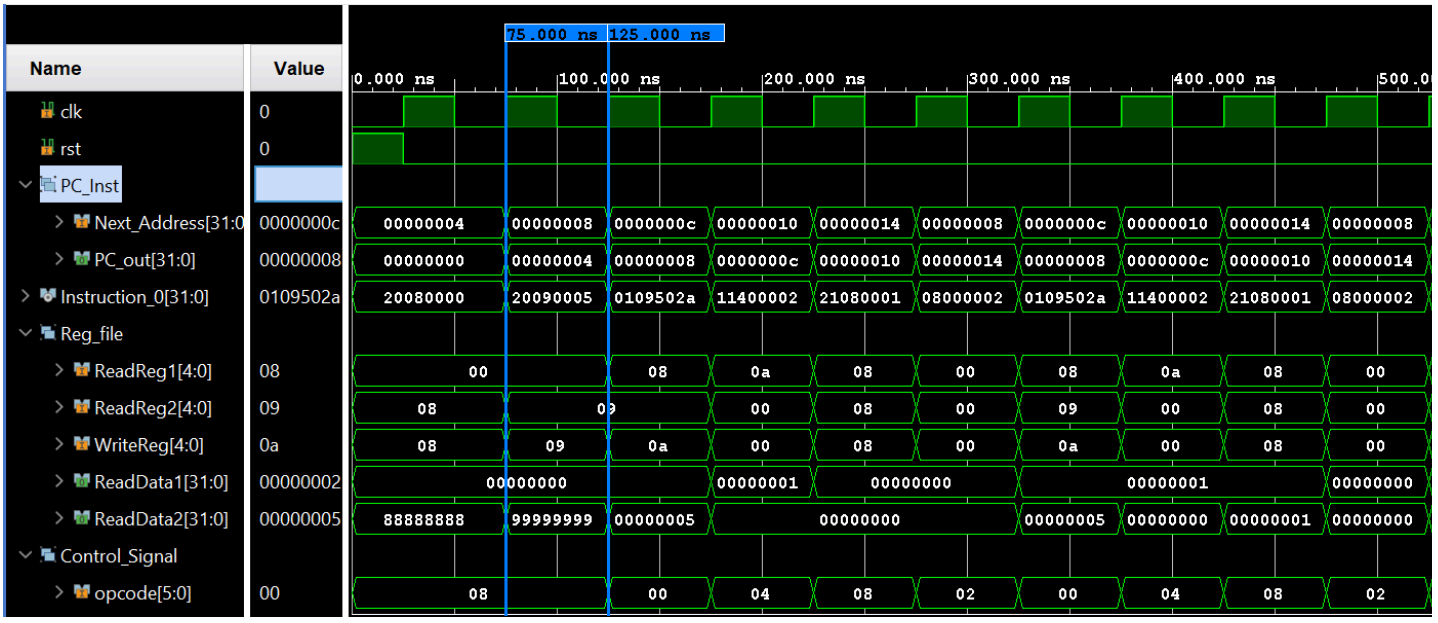
## Example of MIPS single cycle processus

This is the full code that was uploaded to the Instruction Memory:

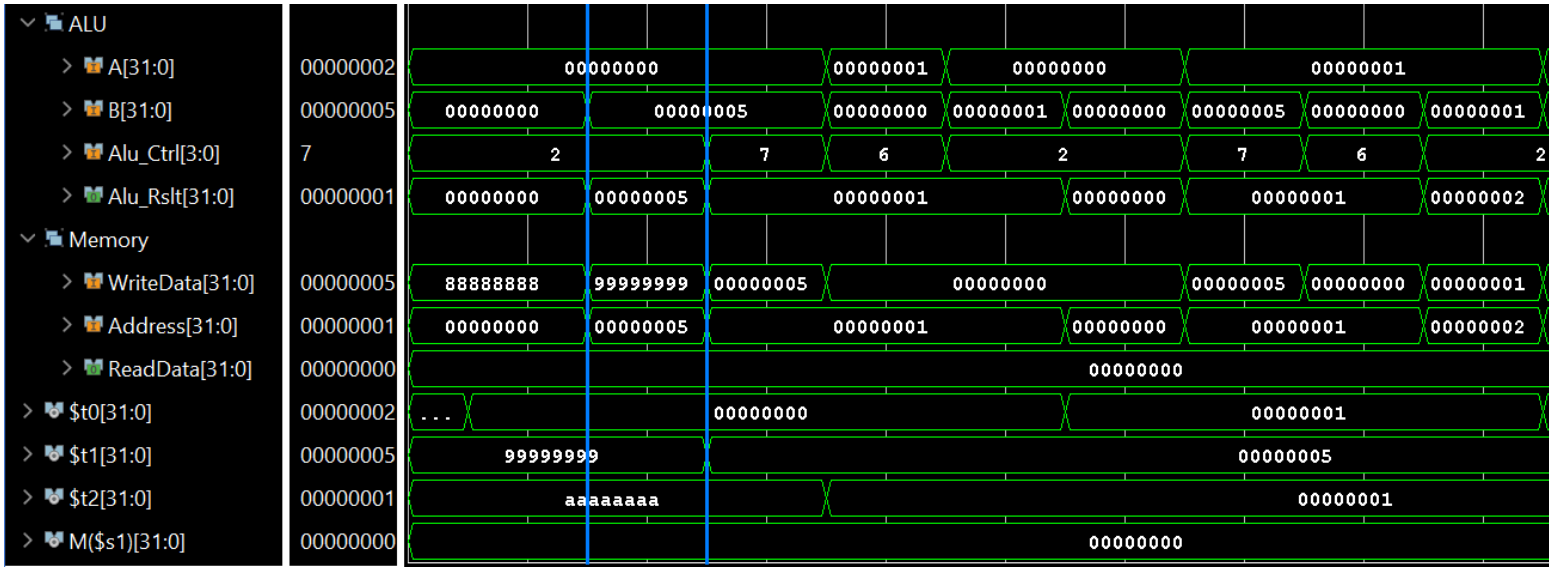
```
    addi $t0, $zero, 0      ; $t0 = 0          ; counter
    addi $t1, $zero, 5      ; $t1 = 5          ; limit
loop:
    slt  $t2, $t0, $t1      ; $t2 = ($t0 < $t1) ? 1 : 0
    beq  $t2, $zero, end    ; if $t2 == 0, exit loop
    addi $t0, $t0, 1        ; $t0 = $t0 + 1
    j    loop               ;repeat loop
end:
    sw   $t0, 0($s1)        ; store final value of $t0 at address
in $s1
```

In the next images, we will decode the five different parts of the cycle using Instruction 2 as an example:

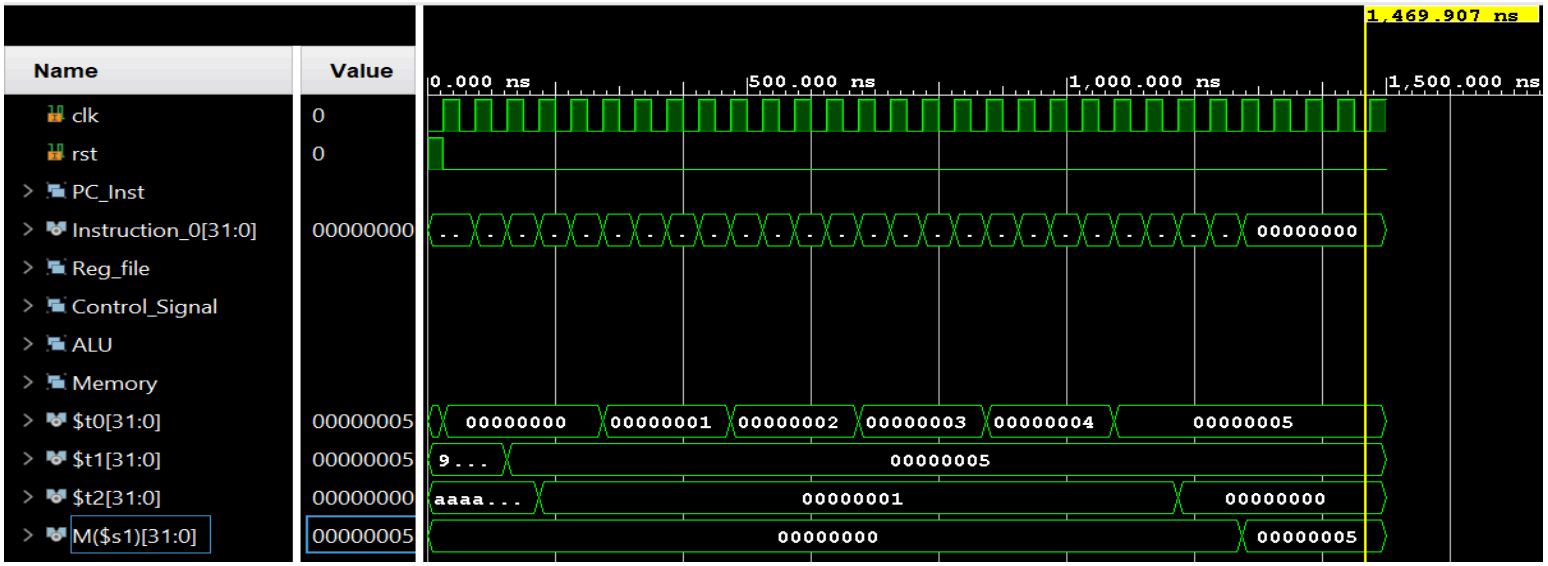
```
addi $t1, $zero, 5
```




*Waveform1: Fetch and Decode part*




*Waveform2: Execute, Memory Access and Write Back*




*Waveform3:Value of important register and M($S1) during tb*

1. **Fetch:**
   - ➔ The **PC** (Program Counter) reads the instruction from address `x"00000004"` in the instruction memory.
   - ➔ It is also incremented by 4 (since each instruction is 4 bytes) to point to the next instruction at address `x"00000008".`

2. **Instruction Decode:**
   - ➔ The instruction (`x"20090005"`) is broken down and sent to different components:
     - **Reg_File** (for reading registers),
     - **Control Unit** (to understand the instruction),
     - The **control unit** takes the **opcode** ( `x"08"`), which tells the processor what operation to perform (in this case, `addi`).

3. **Execute:**
   - ➔ The **ALU** performs the operation given by the **ALU Control**. Here:
     - `A = 0` (the value in register `$zero`) ,`B = 5` (the immediate value), Alu_Ctrl= 2 (which means "add")
     - Alu_Rslt= 5

4. **Memory Access:**
   - ➔ This instruction (`addi`) does not read from or write to memory.Instead, it modifies a **register** value.

5. **Write Back :**
   - ➔ We store the result `5` into the **Register File** at address `x"09"`, which is register `$t1`.Now `$t1 = 5`

In a single-cycle processor, all these steps happen in **one single clock cycle**.That means even simple instructions (like `add`) and more complex ones (like `load`) take the **same amount of time**.