**A 16-bit MIPS CPU Simulation on Logisim**

**Baran B. Hökelek, Tümay Özdemir**

We simulated a fully functioning 16-bit, single-cycle MIPS processor that supports 16 instructions in Logisim, a logic circuit simulation software. Our processor is simpler than a complete MIPS processor – it lacks 32-bit data support, pipeline stages, hazard detection unit and exception handling. Nevertheless, it can perform basic logic & arithmetic operations, and has separate functioning instruction & data memories. It has 8 registers, all of which are open for use for all parties. It has 16GB instruction memory, and 128kB of data memory.

Note: Memory blocks in Logisim are already word-addressable, therefore changing the PC value by 1 made it point to the next word in the instruction memory. That’s why our PC increments by 1 instead of 4, which is the traditional offset.

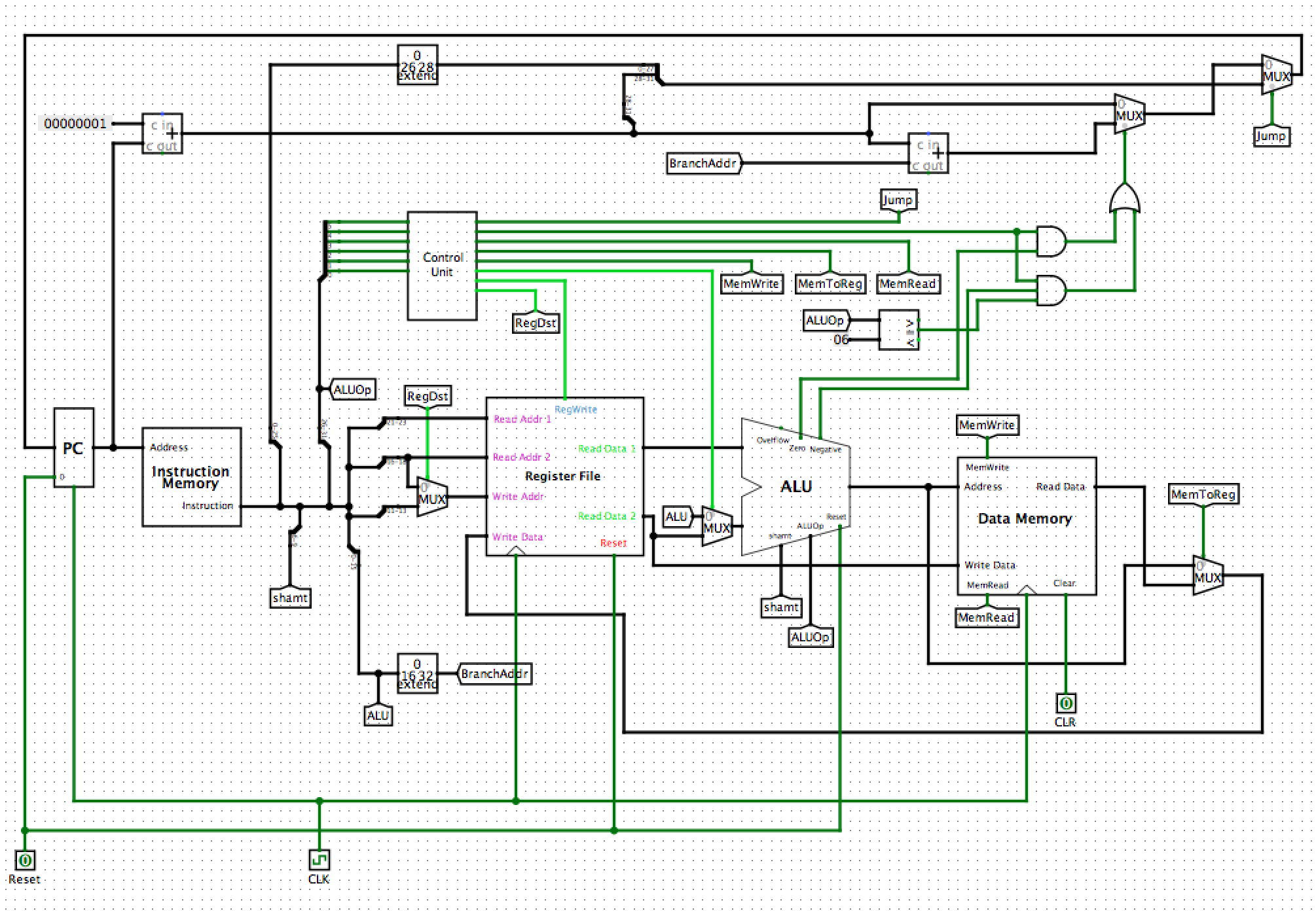


Figure : The CPU, with PC, instruction & data memories, register file and ALU.

**Components:**

* **PC (**Program Counter) takes the address of the next instruction as an input, and passes that value to the instruction memory in each clock pulse.
* **Instruction Memory** takes this address, finds the corresponding instruction, and passes it onto the required components. As seen in the figure, the instruction splits in different parts for different purposes. For example, jump instruction uses the last 26 bits of the instruction as a whole, while R-Type instructions require the division of this part into 4 5-bit & one 6-bit parts.
* **Register File** has 8 registers, labeled $0 to $7.
* **ALU** has special circuits for 16-bit addition, multiplication, and subtraction. It also contains circuits for AND, OR, shift left logical and set less than functions. It has three flags: zero, overflow and negative. Except for the overflow flag, each flag has significance in certain instructions.
* **Data Memory** contains the data that will be used or generated during the execution.
* **Control Unit** takes the opcode from the instruction memory and generates appropriate control signals for the rest of the components to turn on or off, or for certain values to be selected, depending on the instruction.

**Instructions for Pre-Execution**

1. Open “26” from instruction memory. This contains the basic circuit layout of the instruction memory, which contains ROMs (ROMs could take an address of 24-bit max., so we had to extend the number of extra bits two by two to reach 32 bits in 4 iterations. “26” contains the core of that memory.
2. Load the text file named “code” into the first ROM (the one on the top) by using Logisim’s “Load Image” option.
3. Load the text file named “data” into the first RAM inside the data memory by using Logisim’s “Load Image” option.
4. Reset the PC, register file and the multiplication registers in ALU by clicking the “Reset” button at the bottom left.
5. Set $5 to “4”. This is the base address of B [].
6. From the “Simulate” menu at the top, choose “Ticks Enabled” to initiate the simulation.