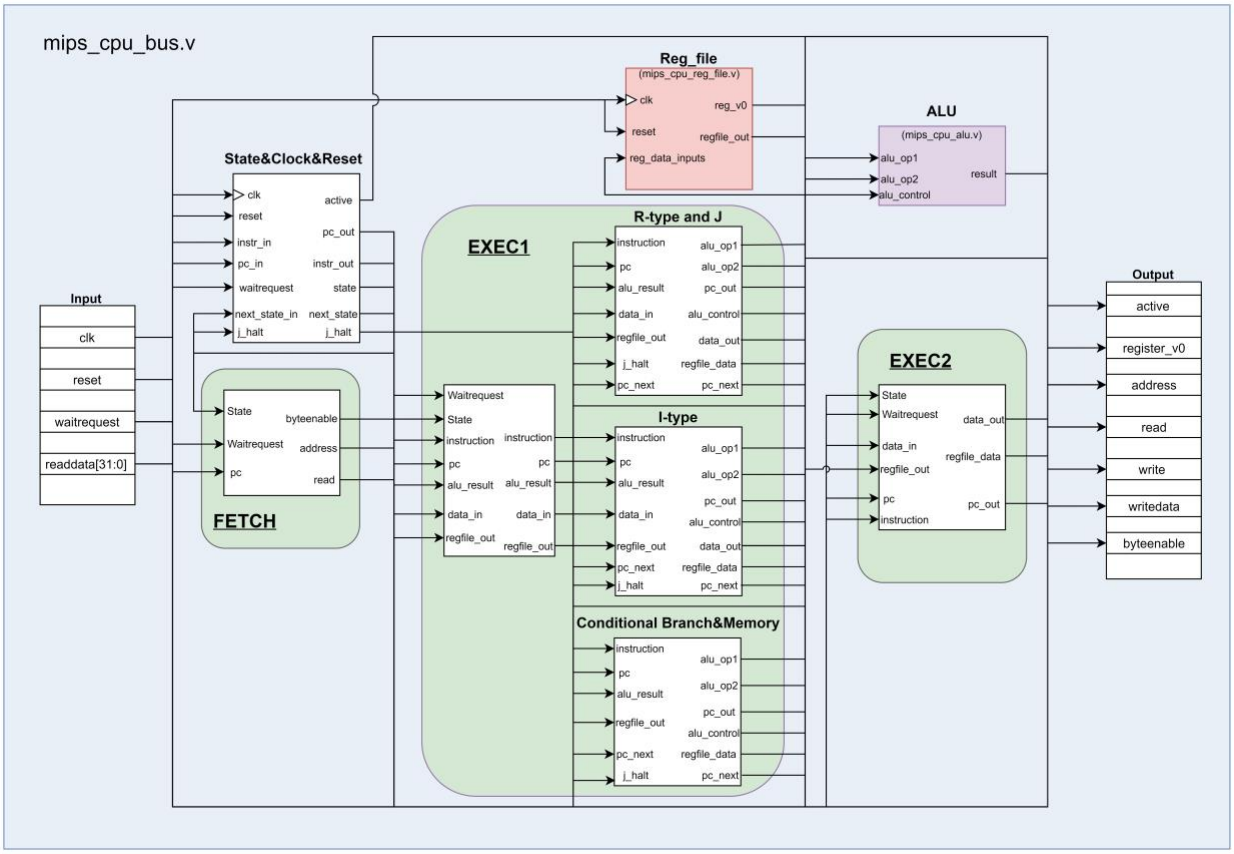
MIPS CPU Design Requirement Specification

In the process of implementing the CPU, all design decisions made are under thorough consideration to achieve the following competitive advantages:

- Human-readable and comprehensible code-based design that allows fast redistribution and easy understanding.
- Sectioned structure with separated functional sections that allow accurate and efficient identification of possible errors and bugs.
- Flexibility to be modified to meet customised demand.
- Compatibility to work on any type of FPGA or other programmable devices.
- Embedded Intel Avalon Protocol.
- Standard I/O ports that can be connected to any external hardware RAM.
- Detailed comment scripts to aid in locating sections and future modifications.

MIPS CPU Architecture



Arithmetic Logic Unit[*mips_cpu_alu.v*]

The Arithmetic Logic Unit is an independent block that computes the arithmetic calculations for the CPU. The sectioned design allows the client to examine the manipulation of numbers through coding. Furtherly, the separation from the main block prevents overcrowding of the Control Unit contents.

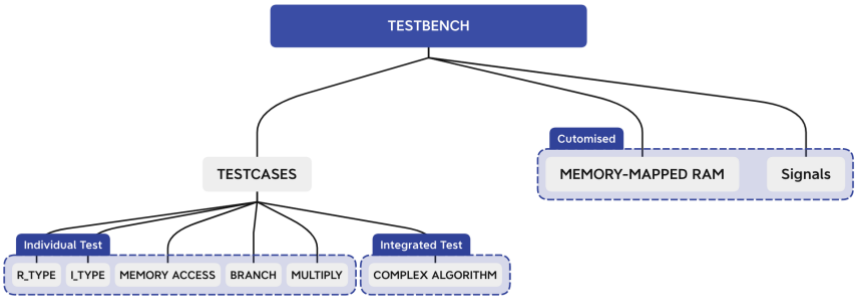
Register File[*mips_cpu_reg_file.v*]

The Register File creates 32-bit registers that temporarily hold data obtained from memory or arithmetic results calculated from the ALU. The size of the Register File can be customised. This block inherits the idea of sectioned design.

Control Unit with Avalon Bus Interface[*mips_cpu_bus.v*]

This main building block contains most of the CPU components with detailed comment scripts to aid client understanding of the MIPS architecture at the block level. The block is separated into functional units and layered into different stages, *Fetch*, *EXEC 1*, and *EXEC 2*. The functional units are *R_TYPE* & *J_TYPE*, *I_TYPE*, and *Conditional Branch & Memory Access*, where each unit is given its control signals and detailed comment scripts to guide other engineers when locating sections, debugging, and modifying instructions. *R_TYPE* and *J_TYPE* are placed together because they share the same operand format. The Avalon Memory-Mapped Interface is integrated with the Control Unit to demonstrate the interaction between instructions (especially *Memory Access*) and external memory. The standardized design of the input and the output ports allows the CPU to be connected to any standardized external memory source.

Testing Approach: Testbench Design



The testing strategy inherits the idea from the sectioned structure and is designed to examine the full functionality of each instruction type individually and collaboratively:

- The individual tests are designed to check the validity of each instruction within the defined type and the integrated tests are designed to examine the overall performance of a given CPU under extreme circumstances. The expected outputs are calculated manually and cross-checked with the output of a MARS MIPS simulator and stored in *4-reference* folder under *test* to ensure the accuracy and the precision of the testcases.
- The customised RAM maps to a 512 of 8 bits memory and is implemented independently outside the CPU for the purpose of testing and can be replaced with the client's memory source.
- Three testing signals (signal 1~3) are implemented inside the Control Unit and the Register File that tells the engineer the exact location of logic errors and bugs when debugging. Despite the fact that the testing signals are not independent, but implemented inside the CPU, they are designed with comment scripts to guide the client to implement the same logic to their customised-designed CPU.

Individual Test

All instructions are tested individually with an exhaustive approach that each instruction is tested, and all possibilities are being considered including positive and negative addition, signed and unsigned comparison, and extreme cases, etc. For instructions, *DIV*, *DIVU*, *MTHI*, *MTLO*, *MULT*, and *MULTU* that interacts with the two special registers *HI* and *LO*, two instructions, *MFHI* and *MFLO* are added to obtain the values from the two special registers to cross-check with the expected values.

Integrated Test

The complex algorithm "merge sort" consists of a series of instructions from all types to test the interactions between each instruction and hence the overall performance of a given CPU in solving complicated tasks. It also demonstrates the interaction between a given CPU and the memory source.

Cyclone IV E 'Auto' Variant

Area Summary

Flow Status	Successful - Fri Dec 17 17:33:59 2021
Quartus Prime Version	20.1.0 Build 711 06/05/2020 SJ Lite Edition
Revision Name	mips_cpu_bus
Top-level Entity Name	mips_cpu_bus
Family	Cyclone IV E
Total logic elements	8,917 / 15,408 (58%)
Total registers	1030
Total pins	234 / 344 (68%)
Total virtual pins	0
Total memory bits	0 / 516,096 (0%)
Embedded Multiplier 9-bit elements	16 / 122 (14%)
Total PLLs	0 / 4 (0%)
Device	EP4CE15F23C6
Timing Models	Final

Timing Summary

Fmax	Restricted Fmax	Clock Name
8.03 MHz	8.03 MHz	waitrequest
49.3 MHz	49.3 MHz	readdate[25]
122.1 MHz	122.1 MHz	alu_control[0]
181.98 MHz	181.98 MHz	clk