The Register File module stands as a cornerstone in the MIPS architecture, and in our CPU design, we have meticulously crafted a RegisterFile module to fulfill the essential role of managing registers efficiently. The module declaration encompasses standard signals, aligning with the MIPS convention, including Clock, ReadReg1, ReadReg2, WriteReg, WriteData, Reg\_write\_Control, ReadData1, and ReadData2.

Our design extends beyond the conventional MIPS Register File by incorporating additional signals to tailor the functionality to our specific requirements:

Reset Signal:

The Reset signal serves as a mechanism to reset all registers to a don’t care (rubbish value) when activated. This feature ensures a clean and controlled initialization, contributing to the reliability and predictability of the system.

PC\_Store Signal:

The PC\_Store signal is a custom addition to our Register File module, specifically designed to interact with register number 31. This signal, initiated by the control unit, enables register 31 to store a value from the write bus. The stored value represents the program counter value and plays a crucial role in the execution of instructions that involve branching, such as Jump And Link and Jump And Link Register.

In addition to these custom signals, we have implemented special-purpose registers to enhance the versatility and efficiency of the Register File:

First Register (Register Number 0):

This register is designed to always maintain a value of zero. It serves as a constant reference point and is exclusively readable. No instruction is permitted to write to this register, ensuring its consistency as a zero value register.

Last Register (Register Number 31):

Register 31 has been specially configured to serve a unique purpose in our design. It is enabled by the PC\_Store signal from the control unit, allowing it to store the program counter value. This functionality is essential for instructions like Jump And Link and Jump And Link Register, where the program counter value is stored in register 31, facilitating the management of subroutine calls and returns.

By incorporating these features, our Register File module not only adheres to MIPS standards but also offers enhanced functionality and customization tailored to the specific requirements of our CPU design.