*Project: RMC75E TEST BENCH*

*Module:* ClockControl*.vhd*

*Author: Satchel Hamilton*

*Company: Delta Motion*

*Date: 7/26/2023*

*Last updated: July 26, 2023*

Contents

[High Level 1](#_Toc141280292)

[Low level 1](#_Toc141280293)

[Simulation 2](#_Toc141280294)

# High Level

The **ClockControl** module is responsible for controlling the clock signals in the system. It interfaces with the **Clock\_Gen** component, which generates the clock signals based on the input clock sources. The module takes various input and output ports, including **H1\_PRIMARY**, **H1\_CLKWR**, **H1\_CLK**, **H1\_CLK90**, **SysClk**, **RESET**, **DLL\_RST**, **DLL\_LOCK**, **SysRESET**, **PowerUp**, **Enable**, and **SlowEnable**.

Inside the **ClockControl** module, there are internal signals and processes that handle the synchronization and control of clock-related operations. The module ensures proper synchronization and control of the clock signals based on the DLL lock status and system reset conditions. It coordinates the generation and distribution of clock signals to different components within the system.

# Low level

The **ClockControl** module uses the **Clock\_Gen** component to generate clock signals. It contains a process to synchronize the DLL lock status (**DLL\_LOCK\_Int**) to the system clock (**SysClk**) and a process to synchronize the DLL reset command (**DLL\_RST\_sync**) to the system clock. It also includes a process to generate initialization pulses (**PowerUpOneShot**) on startup or PLL reset. The **EnableCount** signal is used to generate enable pulses for clocking the logic at different frequencies (**Enable** and **SlowEnable**).

The module has commented-out sections related to a previous implementation of a state machine (**StateMachine**) that monitored the status of the DLL and performed certain actions based on its state. However, this state machine is no longer utilized in the current design, which uses the **Clock\_Gen** component.

## Simulation