source code can be organized into several functional blocks that work together to implement the motion control functionality. Here's an overview of how these modules fit together:

1. CPUConfig.vhd:

• This module is likely responsible for configuring the CPU parameters, such as clock frequency, memory settings, and other CPU-related configurations.

2. Clock\_Gen.vhd:

• The Clock\_Gen module generates various clock signals required by different components of the motion controller.

3. Clock\_Gen\_Clock\_Gen\_0\_FCCC.vhd:

• This module seems to be a specific instantiation of the Clock\_Gen module, possibly generating a particular clock signal named "FCCC."

4. DIO8.vhd:

• This module is likely related to digital input/output (DIO) functionality, providing 8 channels of DIO capabilities.

5. DiscoverControlID.vhd:

• The DiscoverControlID module is involved in discovering and identifying control-related information.

6. DiscoverExpansionID.vhd:

• The DiscoverExpansionID module is responsible for discovering and identifying expansion modules attached to the controller.

7. ExpModuleLED.vhd:

• This module seems to be related to controlling LEDs on expansion modules.

8. ExpansionSigRoute.vhd:

• The ExpansionSigRoute module is likely involved in routing signals between different expansion modules and the main controller.

9. LatencyCounter.vhd:

• This module might be responsible for measuring latency or time delays in the system.

10. MDTTopSimp.vhd:

• The MDTTopSimp module appears to be the top-level module related to the magnetostrictive displacement transducer (MDT) functionality.

11. Quad.vhd:

• This module is likely related to quadrature encoder functionality.

12. QuadXface.vhd:

• The QuadXface module might be an interface for quadrature encoder functionality.

13. SSITop.vhd:

• The SSITop module seems to be the top-level module related to synchronous serial interface (SSI) functionality.

14. WatchDogTimer.vhd:

• This module likely implements a watchdog timer functionality to monitor and reset the system if necessary.

15. analog.vhd:

• The analog module might be related to handling analog signals or analog-to-digital conversion.

16. clockcontrol.vhd:

• This module is likely involved in controlling the clock signals.

17. controlio.vhd:

• The controlio module seems to be related to control input/output operations.

18. controloutput.vhd:

• This module might be responsible for control output functionalities.

19. cpuled.vhd:

• The cpuled module might control LEDs on the main CPU board.

20. databuffer.vhd:

• This module could be related to buffering data.

21. decode.vhd:

• The decode module might be responsible for decoding specific signals or data.

22. discovercontrol.vhd:

• This module could be involved in the discovery process for control-related information.

23. mdssiroute.vhd:

• The mdssiroute module might be related to routing signals in the MDT system.

24. ram128x16bits.vhd:

• This module appears to be a 128x16-bit RAM block.

25. rtdexpidled.vhd:

• The rtdexpidled module seems to control LEDs on the RTD expansion module.

26. serial2parallel.vhd:

• This module might be related to serial-to-parallel data conversion.

27. serial\_mem.vhd:

• The serial\_mem module might be related to serial memory operations.

28. statemachine.vhd:

• This module could implement a state machine for controlling the system behavior.

29. ticksync.vhd:

• The ticksync module might be related to synchronization of timing signals.

30. top.vhd:

• The top module is the top-level module that likely instantiates all other modules and connects them to create the complete motion control system.

From this high-level perspective, we can see that the motion control system is composed of multiple functional blocks, each implemented by one or more modules. These modules interact with each other through their input and output ports, and the top-level module orchestrates the communication and data flow between them to achieve the desired motion control functionalities. The specific details of how these modules work together and the overall system behavior would require further examination of the individual modules and their connections within the top-level module.

Title: RMC75E Modular Motion Controller - Module Interactions

Introduction:

The RMC75E modular motion controller contains several source modules that interact to provide various functionalities for motion control and communication. This document provides an analysis of the interactions among the source modules at both high and low-level perspectives.

High-Level Overview:

The modular motion controller comprises several source modules, each responsible for specific functionalities. The main modules include CPUConfig, Clock\_Gen, DIO8, DiscoverControlID, DiscoverExpansionID, ExpModuleLED, LatencyCounter, MDTTopSimp, Quad, QuadXface, SSITop, WatchDogTimer, analog, clockcontrol, controlio, controloutput, cpuled, databuffer, decode, discovercontrol, mdssiroute, ram128x16bits, rtdexpidled, serial2parallel, serial\_mem, statemachine, ticksync, and top. Let's examine how these modules interact and contribute to the overall functionality of the RMC75E motion controller.

1. DiscoverControlID Module:

The DiscoverControlID module is responsible for identifying the control module connected to the RMC75E motion controller. It communicates with the control module through its specific interface signals, such as M\_Card\_ID\_CLK, M\_Card\_ID\_DATA, M\_Card\_ID\_LATCH, and M\_Card\_ID\_LOAD. During the discovery process, this module reads the control module's identification data and stores it in the ControlID signal. The discovery process involves a state machine that controls the timing of read operations and shifts the data into the ControlID register.

2. DiscoverExpansionID Module:

The DiscoverExpansionID module handles the discovery of expansion modules connected to the RMC75E controller. It uses similar interface signals as the DiscoverControlID module, such as Exp\_ID\_CLK, Exp\_ID\_DATA, Exp\_ID\_LATCH, and Exp\_ID\_LOAD, to communicate with the expansion modules. The module performs a similar discovery process as the control module, reading identification data from the connected expansion modules and storing it in separate ExpansionID signals (ExpansionID0, ExpansionID1, ExpansionID2, and ExpansionID3) for each expansion slot.

3. DiscoverID Module (Wrapper):

The DiscoverID module acts as a wrapper for the DiscoverControlID and DiscoverExpansionID modules. It provides a higher-level interface to the external components and integrates the results of both discovery processes. The module uses several input signals, such as RESET, SysClk, and SlowEnable, to control and synchronize the discovery process. Additionally, it has output signals, such as MDTPresent, ANLGPresent, QUADPresent, and DiscoveryComplete, that indicate the presence of specific modules and the completion of the discovery process.

Low-Level Granular Analysis:

1. DiscoverControlID Module:

- The DiscoverControlID module includes a state machine with various states (s0\_LatchState, s1\_DelayState1, s2\_LoadState, s3\_DelayState2, s4\_ClockState, and s5\_StopState) to control the timing of operations during the discovery process.

- It uses a 6-bit synchronous counter (Count) to generate a terminal count signal (ShiftComplete) when a specific number of clock cycles (TerminalCount) have passed.

- The module shifts the identification data received through the Exp\_ID\_DATA signal into the ExpansionID registers (ExpansionID0 to ExpansionID3) using a 64-bit shift register.

- The Exp\_ID\_CLK signal controls the clock for shifting data into the ExpansionID registers.

2. DiscoverExpansionID Module:

- The DiscoverExpansionID module also includes a state machine (StateMachine) to control the timing of operations during the discovery process for expansion modules.

- It uses a 6-bit synchronous counter (Count) and an output clock signal (OutputClock) to manage the timing of data shifts during the discovery process.

- The module uses a ShiftEnable signal to enable or disable data shifting into the ExpansionID registers based on the state of the state machine.

- The ShiftComplete signal indicates the completion of the discovery process for a specific expansion module slot.

3. DiscoverID Module (Wrapper):

- The DiscoverID module instantiates the DiscoverControlID and DiscoverExpansionID modules and provides an interface to communicate with these modules.

- It handles the overall control and synchronization of the discovery process using input signals such as RESET, SysClk, and SlowEnable.

- The module uses output signals (MDTPresent, ANLGPresent, QUADPresent, and DiscoveryComplete) to convey the results of the discovery process and indicate the presence of specific modules.

Conclusion:

The RMC75E modular motion controller consists of several source modules that interact to perform discovery processes for both control and expansion modules. The DiscoverControlID and DiscoverExpansionID modules handle the identification of control and expansion modules, respectively. The DiscoverID module acts as a wrapper and orchestrates the overall discovery process, providing a higher-level interface to external components. Through these interactions, the motion controller identifies connected modules and determines their presence and types, enabling effective motion control and communication.