```
Insert SHIM below AFTER clear_mem and BEFORE find_smallest
* Backfill state machine info with the names of the new states, new signals needed, and add any new constants to DataTypes_pkg.vhd
* Extend clear_mem to also clear new DIFFS BRAM area.


<BEGIN SHIM>


-- ================================================================================================
-- SCURFMAN
-- ================================================================================
-- Main lab#0 to lab#1 changes here.  2nd attempt.  1st attemt was too many changes to too many things.
-- This is a cleaned up attempt.  Done in 4 additional states; 0) to preset MUX Select
-- and clear any old data from value registers, 1) read the value from the lower pairwise address,
-- 2)read the value from the upper pairwise address, 3) get their difference and load into lower BRAM memory.
-- Read calls require an address and something to map the output to.  Read calls are looking at data this clock cycle.
-- Write calls require an address, a "write enable", some value to be recorded "_din".
-- Read / Write addresses are handled through the MUX "do_PN_histo_addr", which now must have more than 2 options.
-- MUX'ing on greater than two options requires more bits.  Decimal options such as 1, 2, 3, will be displayed as 01, 10, 11 for MUX.
-- Each read or write state must be first directed to the memory region it's going to need.  Initially before entering get_next_PN_lower,
-- the first read state, for the first time, do_PN_histo_addr should already be directed at PN_BRAM space.  This should be set in the
-- prior state just before 'state_next <= get_next_PN_lower'.  Likewise, the first read state should end with pre setting 'do_PN_histo_addr'
-- to the region the next read state will be needing to pull from.  Conversely though, when calling the write in the last state to
-- store the differenced pairwise value; the 'do_PN_histo_addr' called there is a "pre-loading" instruction.  In that case
-- the MUX value assigned in the last state needs to reflect where the _din data will be written to on the next clock cycle, the DIFFS_BRAM.
-- After storing diffs_val to memory, but before looping directly back to 'get_next_PN_lower', the MUX-SEL needs to be re-pointed
-- to the memory that 'get_next_PN_lower' will be expecting to read from.  To do this and keep things separated, the state 'init_diffs'
-- has been added.  Wile it's main purpose is to re-initialize the MUX-SEL, it's a good place to flush out data from the arithmatic
-- value holders PN_lower_val, PN_upper_val, and PN_diffs_val.

-- The first steps are to make the case structure and shim direction between 'clear_mem' and 'find_smallest'.  Once shimmed
-- other edits necessary will include;
-- == adding the 4 new state names to the declarations
-- == adding declarations for any new signals / variables
-- == adding necessary CONSTANTS such as PN_BRAM_SPLIT & DIFFS_UPPER_LIMIT
-- == adjusting size of diffs vals to hold intergers up to 2x value of PN_BRAM integers but maintain 16bit width
-- == adjust MUX-SEL do_PN_histo_addr to go from binary to 4 option (2 bit) choices.
-- == adding any signals used in CONDITIONAL statement of the 4 new states, as additions to the SENSITIVITY LIST of the PROCESS
-- == to the 'clear_mem' state prior to the SHIM, add instruction for also pre-clearing the area to be used for DIFFS results
-- == redirecting the histogram building process away from PN_BRAM and over to DIFFS_BRAM.  These two regions will not be
-- ==== the same size.  Any impace on difference in size should be noted.
-- == The histogram build from the DIFFS values will not be the same size as the what would have been built for PN_BRAM.
```

```
-- ==== Histo results will take up less space than it did before and not occupy all of the space alloted to it.  Make note
-- ==== of this for purposes of the C program presenting the results on the screen and also comparing their performances.

-- ======================
-- SCURFMAN =============
-- Get value of lower pairwise element of PN_BRAM_BASE
          when init_diffs =>

-- Preset MUX to point to lower pairwise memory needed for next state.
             do_PN_histo_addr = "00";

-- Clear the values that will store and difference the pairwise elements
             PN_lower_val <= (others => '0');
             PN_upper_val <= (others => '0');
             diffs_val <= (others => '0');

-- ======================
-- SCURFMAN =============
-- Get value of lower pairwise element of PN_BRAM_BASE
          when get_next_PN_lower =>
             if (PN_lower_addr_reg = PN_BRAM_SPLIT - 1) then
                 PN_lower_addr_next <= PN_BRAM_BASE;
                 state_next <= find_smallest;
             else
-- Pre set MUX for the next Read call
                 do_PN_histo_addr = "10";
                 PN_lower_addr_next <= PN_lower_addr_reg + 1;
                 PN_lower_val <= resize(unsigned(PNL_BRAM_dout));
                 state_next <= get_next_PN_upper;
             end if;

-- ======================
-- SCURFMAN =============
-- Get value of upper pairwise element of PN_BRAM_BASE
          when get_next_PN_upper =>
             if (PN_upper_addr_reg = PN_UPPER_LIMIT - 1) then
                 PN_upper_addr_next <= PN_BRAM_SPLIT;
                 state_next <= find_smallest;
             else
-- MUX was pre set for this read in the last state.  The next state is a write call, which will process after the state,
                 do_PN_histo_addr = "10";
                 PN_upper_addr_next <= PN_upper_addr_reg + 1;
                 PN_upper_val <= resize(unsigned(PNL_BRAM_dout));
                 state_next <= load_pn_diffs
             end if;

-- ======================
-- SCURFMAN =============
-- Get difference of pairwise elements and store in DIFFS_BRAM
          when load_pn_diffs =>
             if (diffs_addr_reg = DIFFS_UPPER_LIMIT -1) then
                 diffs_addr_next <= 0;
                 state_next <= find_smallest;
             else
                 do_PN_histo_addr = "11";
                 diffs_addr_next <= diffs_addr_reg + 1;
                 PNL_BRAM_we = "1";
                 PNL_BRAM_din <= std_logic_vector(unsigned(PN_upper_val - PN_lower_val));
                 state_next <= get_next_PN_lower;


<END SHIM>
```