Module: SV for Verification
Section: Coverage **Task:** Coverage

Task - EDA Project

Coverage

➢ Coverage Class Code:

```
// Author: Noman Rafiq
// Dated: Sep 24, 2024


/* Transaction Class */

// class transaction;
// rand bit [31:0] address;
// rand bit [31:0] data;
// bit write_i;
// endclass

class coverage;

  // Covergroup Starts here
covergroup cg;
  /* Coverpoint for address Signal */
  cp_addr: coverpoint address[7:0] {
    bins SRR = {8'h40}; // Last 8-bits to specifiy offsets for the Register
    bins SPICR = {8'h60};
    bins SPISR = {8'h64};
    bins SPI_DTR = {8'h68};
    bins SPI_DRR = {8'h6C};
    bins SPISSR = {8'h70};
  }

  // Transition Bin from SPISSR to SPICR
  cp_transition: coverpoint address[7:0] {
    bins SPISSR_to_SPICR = { 8'h70 => 8'h60 };
  }

  // Coverpoints for SPICR
  cp_clock: coverpoint data[4:3] {
      bins comb_00 = {2'b00};
      bins comb_01 = {2'b01};
      bins comb_10 = {2'b10};
      bins comb_11 = {2'b11}; // Explicit bins for all combinantions
    }

    cp_lsb: coverpoint data[9] {
      bins lsb_0 = {1'b0};
      bins lsb_1 = {1'b1};
    }
```

```
    cp_loopback: coverpoint data[0] {
      bins loopback_0 = {1'b0};       // Normal Operation
      bins loopback_1 = {1'b1};       // Loopback mode
    }

  /* SPICR coverpoints End Here*/


  /* Coverpoint for SPIDTR Register*/
  cp_spidtr: coverpoint data[7:0] {   // coverpoint for first 8 bits only
      bins comb_all = {[0:255]};      // bins for all 256 combinations
    }

  /* Cross Coverpoints for SPIDTR and SPICR Registers */

  // Cross Coverpoint for SPIDTR
    cp_spidtr_x_addr: cross cp_spidtr, cp_addr {
    ignore_bins other_addresses = !binsof(cp_addr.SPI_DTR);  // Ignore all
addresses except SPI_DTR
  }

  // Cross Coverpoint for SPICR and address
    cp_spicr_x_addr:  cross cp_clock, cp_lsb, cp_loopback, cp_addr {
      option.auto_bin_max = 0;// Disable automatic bins

      bins spicr_x_clock = binsof(cp_addr.SPICR) && binsof(cp_clock); // 4
Cross-Products, <cp_addr.SPICR,comb_00>, <cp_addr.SPICR,comb_01>,
<cp_addr.SPICR,comb_10>, and <cp_addr.SPICR,comb_11>.

      bins spicr_x_lsb = binsof(cp_addr.SPICR) && binsof(cp_lsb); // 2
Cross-Products, <cp_addr.SPICR,lsb_0>, and <cp_addr.SPICR,lsb_1>

      bins spicr_x_loopback = binsof(cp_addr.SPICR) && binsof(cp_loopback);
// 2 Cross-Products, <cp_addr.SPICR,loopback_0>, and
<cp_addr.SPICR,loopback_1>

    }

endgroup

endclass
```