**Module: SV for Verification**

**Section:** Coverage **Task:** Coverage

**Task -** [**EDA Project**](https://www.edaplayground.com/x/kRhW)

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