

We are generating a 32-bit Barrel Shifter with Rotation Control hierarchically in Verilog. We have generated 8-bit Barrel Shifter defined as follows:

```
module barrel_shift_8 (  
  input [7:0] inputData,  
  input [2:0] shiftVal,  
  output [7:0] outputData  
);  
wire [7:0] stage1, stage2;
```

```
// Stage 1: Shift by 1 bit if shiftVal[0] is set  
assign stage1 = shiftVal[0] ? {inputData[6:0], inputData[7]} : inputData;  
  
// Stage 2: Shift by 2 bits if shiftVal[1] is set  
assign stage2 = shiftVal[1] ? {stage1[5:0], stage1[7:6]} : stage1;  
  
// Final Stage: Shift by 4 bits if shiftVal[2] is set  
assign outputData = shiftVal[2] ? {stage2[3:0], stage2[7:4]} : stage2;
```

```
endmodule
```

Please include the previous module(s) in your response and use them to hierarchically generate a 32-bit Barrel Shifter defined as:

```
module barrel_shift_32(inputData[31:0], shiftVal[4:0], outputData[31:0]);  
//Insert code here  
endmodule
```

```
module barrel_shift_8 (  
  input [7:0] inputData,  
  input [2:0] shiftVal,  
  output [7:0] outputData  
);  
  wire [7:0] stage1, stage2;  
  
  // Stage 1: Shift by 1 bit if shiftVal[0] is set  
  assign stage1 = shiftVal[0] ? {inputData[6:0], inputData[7]} : inputData;  
  
  // Stage 2: Shift by 2 bits if shiftVal[1] is set  
  assign stage2 = shiftVal[1] ? {stage1[5:0], stage1[7:6]} : stage1;  
  
  // Final Stage: Shift by 4 bits if shiftVal[2] is set  
  assign outputData = shiftVal[2] ? {stage2[3:0], stage2[7:4]} : stage2;
```

```

endmodule

module barrel_shift_32 (
    input [31:0] inputData,
    input [4:0] shiftVal,
    output [31:0] outputData
);
    wire [31:0] stage1, stage2, stage3;

    // Stage 1: Divide into 8-bit chunks and use barrel_shift_8 for lower 3 bits of shift
    barrel_shift_8 u0 (.inputData(inputData[7:0]), .shiftVal(shiftVal[2:0]), .outputData(stage1[7:0]));
    barrel_shift_8 u1 (.inputData(inputData[15:8]), .shiftVal(shiftVal[2:0]), .outputData(stage1[15:8]));
    barrel_shift_8 u2 (.inputData(inputData[23:16]), .shiftVal(shiftVal[2:0]), .outputData(stage1[23:16]));
    barrel_shift_8 u3 (.inputData(inputData[31:24]), .shiftVal(shiftVal[2:0]), .outputData(stage1[31:24]));

    // Stage 2: Handle shifts by multiples of 8 bits (controlled by shiftVal[3])
    assign stage2 = shiftVal[3] ? {stage1[23:0], stage1[31:24]} : stage1;

    // Stage 3: Handle shifts by multiples of 16 bits (controlled by shiftVal[4])
    assign stage3 = shiftVal[4] ? {stage2[15:0], stage2[31:16]} : stage2;

    // Final output
    assign outputData = stage3;
endmodule

```