```
1 //`include "source/spi_interface.svh"
2 // master
3 //   Ctrl = SPI module view of control interface. see spi_interface.svh
4 //   Spim = master view of SPIbus interface
5 //
6 // Components:
7 //    SPI clock generator: a 0 to 8 x CLKDIV counter, generates
8 //      one SCK cycle per CLKDIV clock cycles. Ctrl.strobe triggers
9 //      it to start counting
10 //    parallel to serial shift register with shift enable
11
12 module master #(CLKDIV=8'd4)(SPIctrl.Master Ctrl, SPIbus.Master Spim,
13                               input Clk_i, Rst_ni);
14
15   logic [7:0] buf_r,buf_nxt, rcv_buf_r;
16   // flag to indicate whether input buffer is full
17   logic inFull_r, inFull_nxt;
18   logic [3:0] bitcnt_r,bitcnt_nxt;
19   logic [7:0] clkcnt_r,clkcnt_nxt;
20   logic [1:0] ss_r,ss_nxt;
21   logic sck_r,sck_nxt;
22   logic [7:0] clkdiv2;
23
24   assign clkdiv2 = CLKDIV >> 1;
25
26   //Transmitter
27
28   always_ff @(posedge Clk_i, negedge Rst_ni) begin
29     if (Rst_ni == 1'b0) begin
30       clkcnt_r <= 8'd0;
31       bitcnt_r <= 8'd0;
32       sck_r <= 1'b0;
33       buf_r <=  8'd0;
34       ss_r <= 2'd0;
35       inFull_r <= 1'b0;
36       end
37     else begin
38       clkcnt_r <= clkcnt_nxt;
39       bitcnt_r <= bitcnt_nxt;
40       sck_r <= sck_nxt;
41       buf_r <= buf_nxt;
42       ss_r <= ss_nxt;
43       inFull_r <= inFull_nxt;
44     end
45   end
46
47   // load shift reg ond ss n strobe
48   always_comb begin
49     buf_nxt = buf_r;
50     ss_nxt = ss_r;
51     if (Ctrl.strobe) begin
52       ss_nxt = Ctrl.ss;
53       buf_nxt = Ctrl.toXmit;
54     end else if (clkcnt_r == CLKDIV) begin
55       buf_nxt = {buf_r[6:0],1'b0};
56     end
57   end
58
59   // update input full flag
60   always_comb begin
61     Ctrl.XmitFull = inFull_r;
62     inFull_nxt = inFull_r;
63     if (Ctrl.strobe)
64       inFull_nxt = 1'b1;
```

```
65      else if (bitcnt_r == 4'd9)
66        inFull_nxt = 1'b0;
67    end
68
69    // pulse sck while output bit is stable
70    always_comb begin
71        sck_nxt = sck_r;
72      if (clkcnt_r == CLKDIV) begin
73        sck_nxt = 1'b0;
74        end
75      else if (clkcnt_r == clkdiv2) begin
76        sck_nxt = 1'b1;
77      end
78    end
79
80    assign Spim.mosi = buf_r[7];
81    assign Spim.sck = sck_r;
82    assign Spim.ss = ss_r;
83
84    // bitcnt holds at 0 waiting for strobe, starts counting after strobe,
85    // at 9 wrap to 0. Only increment after CLKDIV clock cycles
86    always_comb begin
87      bitcnt_nxt = bitcnt_r;
88      // hold at 0 until strobe, then count can proceed at 1
89      if (bitcnt_r == 0 && Ctrl.strobe == 1'b1) begin
90        bitcnt_nxt = 1;
91      // once per bit period, increment bit count, halt at 9
92      end else if (clkcnt_r == CLKDIV - 1) begin
93        if (bitcnt_r > 0 && bitcnt_r < 9) begin
94          bitcnt_nxt = bitcnt_r + 1;
95        end
96      end else if (bitcnt_r == 9) begin
97        bitcnt_nxt = 0;
98        end
99      end
100
101   always_ff @(posedge Clk_i, negedge Rst_ni) begin
102     if (Rst_ni == 1'b0) begin
103       Ctrl.busy = 1'b0;
104       end
105     else begin
106       if (Ctrl.strobe) begin
107         Ctrl.busy = 1'b1;
108         end
109       else if (bitcnt_r == 9) begin
110         Ctrl.busy = 1'b0;
111         end
112       end
113     end
114
115   // clkcnt continuously loops from 1 to CLKDIV after strobe
116   always_comb begin
117     // hold at 0 until Strobe
118     if ((clkcnt_r == 8'd0) && (~Ctrl.strobe) || bitcnt_r == 9) begin
119       clkcnt_nxt = 0;
120     // start at count of 1 upon Strobe
121     end else if ((clkcnt_r == 8'd0) && Ctrl.strobe) begin
122       clkcnt_nxt = 1;
123     // wrap-around to 1 at max clkcnt, or start at 1 on Strobe
124     end else if ((clkcnt_r == CLKDIV) || Ctrl.strobe) begin
125       clkcnt_nxt = 1;
126     end else begin
127       clkcnt_nxt = clkcnt_r + 1;
128       end
```

```
129     end
130
131   //Receiver
132
133   always_ff @(posedge Clk_i, negedge Rst_ni) begin
134     if (Rst_ni == 1'b0) begin
135       Ctrl.Ready <= 1'b0;
136       rcv_buf_r <= 8'd0;
137       Ctrl.Rcvd <= 8'd0;
138       end
139     else begin
140       if (clkcnt_r == clkdiv2) rcv_buf_r <= {rcv_buf_r[6:0],Spim.miso};
141       if (bitcnt_r == 8'd9) begin
142         Ctrl.Rcvd <= rcv_buf_r;
143         Ctrl.Ready <= 1'b1;
144         end
145       //else if (clkcnt_r == clkdiv2) Ctrl.Ready <= 1'b0;
146       else Ctrl.Ready <= 1'b0;
147       end
148     end
149
150 endmodule
```