```vhdl
-- VHDL Fibonacci Sequencer Design

-- VHDL Entity control

LIBRARY ieee ;
USE ieee.std_logic_1164.all;
USE ieee.std_logic_arith.all;

ENTITY control IS
   PORT(
      clock  : IN     std_logic  ;
      reset  : IN     std_logic  ;
      clr    : OUT    std_logic  ;
      inc    : OUT    std_logic  ;
      ld_A_B : OUT    std_logic  ;
      ld_sum : OUT    std_logic
   );

END control ;

-- VHDL Architecture control

LIBRARY ieee ;
USE ieee.std_logic_1164.all;
USE ieee.std_logic_arith.all;

ARCHITECTURE fsm OF control IS

   -- Architecture Declarations
   TYPE state_type IS (
      clr_regs,
      inc_accb,
      load_acc_sum,
      load_acc_A_B
   );

   -- State vector declaration
   ATTRIBUTE state_vector : string;
   ATTRIBUTE state_vector OF fsm : architecture IS "current_state" ;

   -- Declare current and next state signals
   SIGNAL current_state, next_state : state_type ;

BEGIN

   clocked : PROCESS (clock, reset)
   BEGIN
      IF (reset = '1') THEN
         current_state <= clr_regs;
         -- Reset Values
      ELSIF (clock'EVENT AND clock = '1') THEN
         current_state <= next_state;
         -- Default Assignment To Internals

      END IF;

   END PROCESS clocked;

   nextstate : PROCESS (current_state)
   BEGIN
      CASE current_state IS
      WHEN clr_regs =>
            next_state <= inc_accb;
      WHEN inc_accb =>
            next_state <= load_acc_sum;
      WHEN load_acc_sum =>
            next_state <= load_acc_A_B;
      WHEN load_acc_A_B =>
            next_state <= load_acc_sum;
      WHEN OTHERS =>
         next_state <= clr_regs;

      END CASE;

   END PROCESS nextstate;
```

```
76
77        output : PROCESS (current_state)
78        BEGIN
79           -- Default Assignment
80           clr <= '0';
81           inc <= '0';
82           ld_A_B <= '0';
83           ld_sum <= '0';
84
85           -- State Actions
86           CASE current_state IS
87           WHEN clr_regs =>
88              clr <= '1' ; -- Corrected error on 10/29/Monday
89              inc <= '0' ;
90              ld_A_B <= '0' ;
91              ld_sum <= '0' ;
92           WHEN inc_accb =>
93              clr <= '0' ; -- Corrected error on 10/29/Monday
94              inc <= '1' ;
95           WHEN load_acc_sum =>
96              inc <= '0' ;
97              ld_A_B <= '0' ;
98              ld_sum <= '1' ;
99           WHEN load_acc_A_B =>
100             ld_A_B <= '1' ;
101             ld_sum <= '0' ;
102          WHEN OTHERS =>
103             NULL;
104          END CASE;
105
106       END PROCESS output;
107
108    END fsm;
109
110    -- VHDL Entity accumulator
111
112    LIBRARY ieee ;
113    USE ieee.std_logic_1164.all;
114    USE ieee.std_logic_arith.all;
115
116    ENTITY accumulator IS
117       PORT(
118          clock : IN     std_logic  ;
119          clr   : IN     std_logic  ;
120          inc   : IN     std_logic  ;
121          ip    : IN     std_logic_vector (7 DOWNTO 0) ;
122          ld    : IN     std_logic  ;
123          op    : BUFFER std_logic_vector (7 DOWNTO 0)
124       );
125
126    END accumulator ;
127
```

```
128    -- VHDL Architecture accumulator
129
130    ARCHITECTURE spec OF accumulator IS
131
132    BEGIN
133
134       truth_process: PROCESS(clock)
135
136       BEGIN
137          IF (clock'EVENT AND clock = '1') THEN
138             IF (clr = '1') THEN
139                -- Reset Actions
140                op <= "00000000";
141             ELSE
142                IF (ld = '1') THEN
143                   op <= ip;
144                ELSIF (inc = '1') THEN
145                   op <= unsigned(op)+1;
146                ELSE
147                   op <= op;
148                END IF;
149
150             END IF;
151          END IF;
152       END PROCESS truth_process;
153
154    END spec;
155
156
157    -- VHDL Entity Seq_Generator
158
159    LIBRARY ieee ;
160    USE ieee.std_logic_1164.all;
161    USE ieee.std_logic_arith.all;
162
163    ENTITY Seq_Generator IS
164       PORT(
165          clk  : IN     std_logic  ;
166          reset  : IN     std_logic  ;
167          fibout : OUT    std_logic_vector (7 DOWNTO 0)
168       );
169
170    END Seq_Generator ;
171
172    LIBRARY ieee ;
173    USE ieee.std_logic_1164.ALL;
174    USE ieee.std_logic_arith.ALL;
175
176    ARCHITECTURE struct OF Seq_Generator IS
177
178    -- Internal signal declarations
179    SIGNAL A       : std_logic_vector(7 DOWNTO 0);
180    SIGNAL B       : std_logic_vector(7 DOWNTO 0);
181    SIGNAL clr     : std_logic;
182    SIGNAL gnd     : std_logic;
183    SIGNAL inc     : std_logic;
184    SIGNAL ld_A_B  : std_logic;
185    SIGNAL ld_sum  : std_logic;
186    SIGNAL sum     : std_logic_vector(7 DOWNTO 0);
187
188    -- Implicit buffer signal declarations
189    SIGNAL fibout_internal : std_logic_vector (7 DOWNTO 0);
190
```

```
191     -- Component Declarations
192     COMPONENT accumulator
193        PORT (
194           clock : IN     std_logic ;
195           clr   : IN     std_logic ;
196           inc   : IN     std_logic ;
197           ip    : IN     std_logic_vector (7 DOWNTO 0);
198           ld    : IN     std_logic ;
199           op    : BUFFER std_logic_vector (7 DOWNTO 0)
200        );
201     END COMPONENT;
202     COMPONENT control
203        PORT (
204           clock  : IN     std_logic ;
205           reset  : IN     std_logic ;
206           clr    : OUT    std_logic ;
207           inc    : OUT    std_logic ;
208           ld_A_B : OUT    std_logic ;
209           ld_sum : OUT    std_logic
210        );
211     END COMPONENT;
212
213     BEGIN
214        -- Architecture concurrent statements
215
216        -- Add signals A and B together
217        sum <= unsigned(A) + unsigned(B) ;
218
219        -- Tie signal gnd to ground
220        gnd <= '0' ;
221
222        acc_A : accumulator
223           PORT MAP (
224              clock => clk,
225              clr   => clr,
226              inc   => gnd,
227              ip    => fibout_internal,
228              ld    => ld_A_B,
229              op    => A
230           );
231        acc_B : accumulator
232           PORT MAP (
233              clock => clk,
234              clr   => clr,
235              inc   => inc,
236              ip    => A,
237              ld    => ld_A_B,
238              op    => B
239           );
240        acc_sum : accumulator
241           PORT MAP (
242              clock => clk,
243              clr   => clr,
244              inc   => gnd,
245              ip    => sum,
246              ld    => ld_sum,
247              op    => fibout_internal
248           );
249        FSM: control
250           PORT MAP (
251              clock  => clk,
252              reset  => reset,
253              clr    => clr,
254              inc    => inc,
255              ld_A_B => ld_A_B,
256              ld_sum => ld_sum
257           );
258
259        -- Implicit buffered output assignments
260        fibout <= fibout_internal;
261
262     END struct;
```