```
1  --
2  --  10,000 year calendar clock
3  --
4  --  Context Clauses
5
6      --  Library Clause
7  LIBRARY ieee;
8
9      --  Use Clause
10 USE ieee.std_logic_1164.all;
11 USE ieee.std_logic_unsigned.all;
12
13 --  Library Units
14
15     --  Package Declaration (optional)
16
17     --  Package Body (optional)
18
19     --  Entity Declaration
20 ENTITY calendar IS
21     GENERIC(NCLKSEC : integer :=  999999;
22             NCLKDISP : integer:=  511);
23     PORT(
24         VALUE       : IN    STD_LOGIC_VECTOR(7 downto 0);
25         S       : IN    STD_LOGIC_VECTOR(2 downto 0);
26         CLOCK1M     : IN    STD_LOGIC;
27         nPRESET     : IN    STD_LOGIC;
28         SEG         : OUT   STD_LOGIC_VECTOR(6 downto 0);
29         nDIGIT      : OUT   STD_LOGIC_VECTOR(13 downto 0));
30 END calendar;
31
32     --  Architecture Body
33 ARCHITECTURE behavioral OF calendar IS
34     SIGNAL YEAR : STD_LOGIC_VECTOR(15 downto 0);
35     SIGNAL MONTH : STD_LOGIC_VECTOR(4 downto 0);
36     SIGNAL DAY  : STD_LOGIC_VECTOR(5 downto 0);
37     SIGNAL HOUR : STD_LOGIC_VECTOR(5 downto 0);
38     SIGNAL MINUTE   : STD_LOGIC_VECTOR(6 downto 0);
```

```
39      SIGNAL SECOND   : STD_LOGIC_VECTOR(6 downto 0);
40      SIGNAL counter  : STD_LOGIC_VECTOR(19 downto 0);
41      SIGNAL dispcnt  : STD_LOGIC_VECTOR(15 downto 0);
42      SIGNAL dispsel  : STD_LOGIC_VECTOR(3 downto 0);
43      FUNCTION bcddecode(bcd:STD_LOGIC_VECTOR(3 downto 0))
44          RETURN STD_LOGIC_VECTOR IS
45          VARIABLE ledout:STD_LOGIC_VECTOR(6 downto 0);
46      BEGIN
47          CASE bcd IS
48              WHEN "0000" =>
49                  ledout:="0111111";
50              WHEN "0001" =>
51                  ledout:="0000110";
52              WHEN "0010" =>
53                  ledout:="1011011";
54              WHEN "0011" =>
55                  ledout:="1001111";
56              WHEN "0100" =>
57                  ledout:="1100110";
58              WHEN "0101" =>
59                  ledout:="1101101";
60              WHEN "0110" =>
61                  ledout:="1111101";
62              WHEN "0111" =>
63                  ledout:="0000111";
64              WHEN "1000" =>
65                  ledout:="1111111";
66              WHEN "1001" =>
67                  ledout:="1101111";
68              WHEN OTHERS =>
69                  ledout:="0000000";
70          END CASE;
71          RETURN ledout;
72      END bcddecode;
73      FUNCTION digitdecode(sel:STD_LOGIC_VECTOR(3 downto 0))
74          RETURN STD_LOGIC_VECTOR IS
75          VARIABLE digit:STD_LOGIC_VECTOR(13 downto 0);
76      BEGIN
```

```
77          CASE sel IS
78              WHEN "0000" =>
79                  digit:="11111111111110";
80              WHEN "0001" =>
81                  digit:="11111111111101";
82              WHEN "0010" =>
83                  digit:="11111111111011";
84              WHEN "0011" =>
85                  digit:="11111111110111";
86              WHEN "0100" =>
87                  digit:="11111111101111";
88              WHEN "0101" =>
89                  digit:="11111111011111";
90              WHEN "0110" =>
91                  digit:="11111110111111";
92              WHEN "0111" =>
93                  digit:="11111101111111";
94              WHEN "1000" =>
95                  digit:="11111011111111";
96              WHEN "1001" =>
97                  digit:="11110111111111";
98              WHEN "1010" =>
99                  digit:="11101111111111";
100             WHEN "1011" =>
101                 digit:="11011111111111";
102             WHEN "1100" =>
103                 digit:="10111111111111";
104             WHEN "1101" =>
105                 digit:="01111111111111";
106             WHEN OTHERS =>
107                 digit:="11111111111111";
108         END CASE;
109         IF (MONTH(4)='0') THEN
110             digit(4):='1';
111         END IF;
112         IF (DAY(5 downto 4)="00") THEN
113             digit(6):='1';
114         END IF;
```

```
115         IF (HOUR(5 downto 4)="00") THEN
116             digit(8):='1';
117         END IF;
118         IF (MINUTE(6 downto 4)="000") THEN
119             digit(10):='1';
120         END IF;
121         IF (SECOND(6 downto 4)="000") THEN
122             digit(12):='1';
123         END IF;
124         RETURN digit;
125     END digitdecode;
126     FUNCTION selbcd(sel:STD_LOGIC_VECTOR(3 downto 0))
127         RETURN STD_LOGIC_VECTOR IS
128         VARIABLE segment:STD_LOGIC_VECTOR(3 downto 0);
129     BEGIN
130         CASE sel IS
131             WHEN "0000" =>
132                 segment:=YEAR(15 downto 12);
133             WHEN "0001" =>
134                 segment:=YEAR(11 downto 8);
135             WHEN "0010" =>
136                 segment:=YEAR(7 downto 4);
137             WHEN "0011" =>
138                 segment:=YEAR(3 downto 0);
139             WHEN "0100" =>
140                 segment:="000"&MONTH(4);
141             WHEN "0101" =>
142                 segment:=MONTH(3 downto 0);
143             WHEN "0110" =>
144                 segment:="00" & DAY(5 downto 4);
145             WHEN "0111" =>
146                 segment:=DAY(3 downto 0);
147             WHEN "1000" =>
148                 segment:="00" & HOUR(5 downto 4);
149             WHEN "1001" =>
150                 segment:=HOUR(3 downto 0);
151             WHEN "1010" =>
152                 segment:='0' & MINUTE(6 downto 4);
```

```
153             WHEN "1011" =>
154                 segment:=MINUTE(3 downto 0);
155             WHEN "1100" =>
156                 segment:='0' & SECOND(6 downto 4);
157             WHEN "1101" =>
158                 segment:=SECOND(3 downto 0);
159             WHEN OTHERS =>
160 --              segment:="0000";
161         END CASE;
162         RETURN segment;
163     END selbcd;
164     FUNCTION LeapYear(y:STD_LOGIC_VECTOR(15 downto 0))
165         RETURN STD_LOGIC IS
166         VARIABLE isLeapYear:STD_LOGIC;
167     BEGIN
168         IF (y(7 downto 0)=X"00") THEN
169             CASE y(15 downto 8) IS
170                 WHEN X"00" => -- year 0000
171                     isLeapYear := '1';
172                 WHEN X"04" => -- year 0400
173                     isLeapYear := '1';
174                 WHEN X"08" => -- year 0800
175                     isLeapYear := '1';
176                 WHEN X"12" => -- year 1200
177                     isLeapYear := '1';
178                 WHEN X"16" => -- year 1600
179                     isLeapYear := '1';
180                 WHEN X"20" => -- year 2000
181                     isLeapYear := '1';
182                 WHEN X"24" => -- year 2400
183                     isLeapYear := '1';
184                 WHEN X"28" => -- year 2800
185                     isLeapYear := '1';
186                 WHEN X"32" => -- year 3200
187                     isLeapYear := '1';
188                 WHEN X"36" => -- year 3600
189                     isLeapYear := '1';
190                 WHEN X"40" => -- year 4000
```

```
191                     isLeapYear := '1';
192                 WHEN X"44" => -- year 4400
193                     isLeapYear := '1';
194                 WHEN X"48" => -- year 4800
195                     isLeapYear := '1';
196                 WHEN X"52" => -- year 5200
197                     isLeapYear := '1';
198                 WHEN X"56" => -- year 5600
199                     isLeapYear := '1';
200                 WHEN X"60" => -- year 6000
201                     isLeapYear := '1';
202                 WHEN X"64" => -- year 6400
203                     isLeapYear := '1';
204                 WHEN X"68" => -- year 6800
205                     isLeapYear := '1';
206                 WHEN X"72" => -- year 7200
207                     isLeapYear := '1';
208                 WHEN X"76" => -- year 7600
209                     isLeapYear := '1';
210                 WHEN X"80" => -- year 8000
211                     isLeapYear := '1';
212                 WHEN X"84" => -- year 8400
213                     isLeapYear := '1';
214                 WHEN X"88" => -- year 8800
215                     isLeapYear := '1';
216                 WHEN X"92" => -- year 9200
217                     isLeapYear := '1';
218                 WHEN X"96" => -- year 9600
219                     isLeapYear := '1';
220                 WHEN OTHERS =>
221                     isLeapYear := '0';
222             END CASE;
223         ELSIF (y(1 downto 0)="00") THEN
224             isLeapYear := '1';
225         ELSE
226             isLeapYear := '0';
227         END IF;
228         RETURN isLeapYear;
```

```
229     END LeapYear;
230 BEGIN
231     --  Process Statement
232     PROCESS (CLOCK1M)
233     BEGIN
234         IF CLOCK1M'EVENT AND CLOCK1M = '1' THEN
235             IF (nPRESET='0') THEN
236                 CASE S IS
237                     WHEN "000" =>
238                         YEAR(15 downto 8) <= VALUE;
239                     WHEN "001" =>
240                         YEAR(7 downto 0) <= VALUE;
241                     WHEN "010" =>
242                         MONTH(4 downto 0) <= VALUE(4 downto 0);
243                     WHEN "011" =>
244                         DAY(5 downto 0) <= VALUE(5 downto 0);
245                     WHEN "100" =>
246                         HOUR(5 downto 0) <= VALUE(5 downto 0);
247                     WHEN "101" =>
248                         MINUTE(6 downto 0) <= VALUE(6 downto 0);
249                     WHEN "110" =>
250                         SECOND(6 downto 0) <= VALUE(6 downto 0);
251                     WHEN OTHERS =>
252                 END CASE;
253             ELSIF (counter=NCLKSEC) THEN -- see if 1sec event
254                 FOR i IN 0 to 19 LOOP
255                     counter(i) <= '0';
256                 END LOOP;
257                 IF (SECOND="1011001") THEN -- see if second is 59
258                     SECOND <= "0000000";
259                     IF (MINUTE="1011001") THEN -- see if minute is 59
260                         MINUTE <= "0000000";
261                         IF (HOUR="100011") THEN -- see if hour is 23
262                             HOUR <= "000000";
263                             IF (MONTH="00010") THEN -- see if month is feb
264                                 IF ((LeapYear(YEAR)='1' AND (DAY="101001")) OR
265                                     (LeapYear(YEAR)='0' AND (DAY="101000"))) THEN -- see if day is 29th of le
ap year or 28th
```

```
266                             DAY <= "000001";
267                             MONTH(3 downto 0) <= MONTH(3 downto 0)+'1'; -- should become march 1st
268                         ELSIF (DAY(3 downto 0)="1001") THEN -- see if day is x9
269                             DAY(3 downto 0) <= "0000";
270                             DAY(5 downto 4) <= DAY(5 downto 4)+'1';
271                         ELSE
272                             DAY(3 downto 0) <= DAY(3 downto 0)+'1';
273                         END IF;
274                     ELSIF ((MONTH="00001") OR (MONTH="00011") OR (MONTH="00101") OR
275                             (MONTH="00111") OR (MONTH="01000") OR (MONTH="10000") OR
276                             (MONTH="10010")) THEN -- see if month is jan, march, may, july, aug, oct,
    or dec
277                         IF (DAY="110001") THEN -- see if day is 31th
278                             DAY <= "000001"; -- should become 1st
279                             IF (MONTH="10010") THEN -- see if month is dec
280                                 MONTH <= "00001"; -- should become jan
281                                 IF (YEAR(3 downto 0)="1001") THEN -- see if year is xxx9
282                                     YEAR(3 downto 0) <= "0000"; -- should become xxx0
283                                     IF (YEAR(7 downto 4)="1001") THEN -- see if year is xx99
284                                         YEAR(7 downto 4) <= "0000"; -- should become xx00
285                                         IF (YEAR(11 downto 8)="1001") THEN -- see if year is x999
286                                             YEAR(11 downto 8) <= "0000"; -- should become x000
287                                             IF (YEAR(15 downto 12)="1001") THEN -- see if year is 999
    9
288                                                 YEAR(15 downto 12) <= "0000"; -- should become 0000
289                                             ELSE
290                                                 YEAR(15 downto 12) <= YEAR(15 downto 12)+'1';
291                                             END IF;
292                                         ELSE
293                                             YEAR(11 downto 8) <= YEAR(11 downto 8)+'1';
294                                         END IF;
295                                     ELSE
296                                         YEAR(7 downto 4) <= YEAR(7 downto 4)+'1';
297                                     END IF;
298                                 ELSE
299                                     YEAR(3 downto 0) <= YEAR(3 downto 0)+'1';
300                                 END IF;
301                             ELSE -- month is not dec
```

```
                                    MONTH(3 downto 0) <= MONTH(3 downto 0)+'1';
                                END IF;
                            ELSIF (DAY(3 downto 0)="1001") THEN -- see if day is x9
                                DAY(3 downto 0) <= "0000";
                                DAY(5 downto 4) <= DAY(5 downto 4)+'1';
                            ELSE -- day is not x9
                                DAY(3 downto 0) <= DAY(3 downto 0)+'1';
                            END IF;
                        ELSIF (DAY(5 downto 0)="110000") THEN -- see if day is 30th
                            DAY(5 downto 0) <= "000001"; -- should become 1st
                            IF (MONTH(4 downto 0)="01001") THEN -- see if month is sep
                                MONTH(4 downto 0) <= "10000"; -- should become oct
                            ELSE -- month is not sep
                                MONTH(3 downto 0) <= MONTH(3 downto 0)+'1'; -- of next month
                            END IF;
                        ELSIF (DAY(3 downto 0)="1001") THEN -- see if day is x9th
                            DAY(3 downto 0) <= "0000"; -- should become x0th
                            DAY(5 downto 4) <= DAY(5 downto 4)+'1';
                        ELSE -- day is not x9th
                            DAY(3 downto 0) <= DAY(3 downto 0)+'1';
                        END IF;
                    ELSIF (HOUR(3 downto 0)=X"9") THEN -- see if hour is x9
                        HOUR(3 downto 0) <= "0000"; -- should become x0
                        HOUR(5 downto 4) <= HOUR(5 downto 4)+'1';
                    ELSE -- hour is not x9
                        HOUR(3 downto 0) <= HOUR(3 downto 0)+'1';
                    END IF;
                ELSIF (MINUTE(3 downto 0)="1001") THEN -- see if minute is x9
                    MINUTE(3 downto 0) <= "0000"; -- should become x0
                    MINUTE(6 downto 4) <= MINUTE(6 downto 4)+'1';
                ELSE -- minute is not x9
                    MINUTE(3 downto 0) <= MINUTE(3 downto 0)+'1';
                END IF;
            ELSIF (SECOND(3 downto 0)="1001") THEN -- see if second is x9
                SECOND(3 downto 0) <= "0000"; -- should become x0
                SECOND(6 downto 4) <= SECOND(6 downto 4)+'1';
            ELSE -- second is not x9
                SECOND(3 downto 0) <= SECOND(3 downto 0)+'1';
```

```
340                 END IF;
341             ELSE -- not 1sec event
342                 counter <= counter+'1';
343             END IF;
344             IF (dispcnt=NCLKDISP) THEN -- see if display change event
345                 FOR i IN 0 to 15 LOOP
346                     dispcnt(i) <= '0';
347                 END LOOP;
348                 IF (dispsel="1101") THEN -- see if dispsel is 13
349                     dispsel <= "0000";
350                 ELSE -- dispsel is not 13
351                     dispsel <= dispsel+'1';
352                 END IF;
353             ELSE -- not display change event
354                 dispcnt <= dispcnt+'1';
355             END IF;
356         END IF;
357     END PROCESS;
358
359     --  Concurrent Procedure Call
360
361     --  Concurrent Signal Assignment
362     SEG <= bcddecode(selbcd(dispsel));
363     nDIGIT <= digitdecode(dispsel);
364
365     --  Conditional Signal Assignment
366
367     --  Selected Signal Assignment
368
369     --  Component Instantiation Statement
370
371     --  Generate Statement
372 END behavioral;
373
374
```