```
1  //
2  //  10,000 year calendar clock in FLEX10K
3  //
4  module calendar(VALUE,SELECT,nPRESET,CLOCK1M,SEG,nDIGIT);
5      parameter NCLKSEC=20'd999999;
6      parameter NCOUNTER=20;
7      parameter NCLKDISP=16'd511;
8      parameter NDISPCNT=16;
9      parameter NDISPSEL=4;
10
11     input [7:0] VALUE;
12     input [2:0] SELECT;
13     input nPRESET;
14     input CLOCK1M;
15
16     output [6:0] SEG;
17     output [13:0] nDIGIT;
18
19     reg [15:0] YEAR;
20     reg [4:0] MONTH;
21     reg [5:0] DAY;
22     reg [5:0] HOUR;
23     reg [6:0] MINUTE;
24     reg [6:0] SECOND;
25     reg [NCOUNTER-1:0] counter;
26     reg [NDISPCNT-1:0] dispcnt;
27     reg [NDISPSEL-1:0] dispsel;
28
29     integer i;
30
31 function [6:0] bcddecode;
32     input [3:0] bcd;
33
34     case (bcd[3:0])
35         4'd0:
36                 bcddecode = 7'b0111111;
37         4'd1:
38                 bcddecode = 7'b0000110;
```

```
39          4'd2:
40                  bcddecode = 7'b1011011;
41          4'd3:
42                  bcddecode = 7'b1001111;
43          4'd4:
44                  bcddecode = 7'b1100110;
45          4'd5:
46                  bcddecode = 7'b1101101;
47          4'd6:
48                  bcddecode = 7'b1111101;
49          4'd7:
50                  bcddecode = 7'b0000111;
51          4'd8:
52                  bcddecode = 7'b1111111;
53          4'd9:
54                  bcddecode = 7'b1101111;
55          default:
56                  bcddecode = 7'b0000000;
57      endcase
58
59  endfunction
60
61  function [13:0] digit;
62      input [3:0] sel;
63  begin
64      case (sel)
65          4'b0000:
66              digit = 14'b11111111111110;
67          4'b0001:
68              digit = 14'b11111111111101;
69          4'b0010:
70              digit = 14'b11111111111011;
71          4'b0011:
72              digit = 14'b11111111110111;
73          4'b0100:
74              digit = 14'b11111111101111;
75          4'b0101:
76              digit = 14'b11111111011111;
```

```
77          4'b0110:
78              digit = 14'b11111110111111;
79          4'b0111:
80              digit = 14'b11111101111111;
81          4'b1000:
82              digit = 14'b11111011111111;
83          4'b1001:
84              digit = 14'b11110111111111;
85          4'b1010:
86              digit = 14'b11101111111111;
87          4'b1011:
88              digit = 14'b11011111111111;
89          4'b1100:
90              digit = 14'b10111111111111;
91          4'b1101:
92              digit = 14'b01111111111111;
93          default:
94              digit = 14'b11111111111111;
95      endcase
96      if (MONTH[4]==1'b0) digit[4] = 1'b1;
97      if (DAY[5:4]==2'b00) digit[6] = 1'b1;
98      if (HOUR[5:4]==2'b00) digit[8] = 1'b1;
99      if (MINUTE[6:4]==3'b000) digit[10] = 1'b1;
100     if (SECOND[6:4]==3'b000) digit[12] = 1'b1;
101 end
102 endfunction
103
104 function [3:0] selbcd;
105     input [3:0] sel;
106
107     case (sel)
108         4'b0000:
109             selbcd = YEAR[15:12];
110         4'b0001:
111             selbcd = YEAR[11:8];
112         4'b0010:
113             selbcd = YEAR[7:4];
114         4'b0011:
```

```
115             selbcd = YEAR[3:0];
116         4'b0100:
117             selbcd = {1'b0,1'b0,1'b0,MONTH[4]};
118         4'b0101:
119             selbcd = MONTH[3:0];
120         4'b0110:
121             selbcd = {1'b0,1'b0,DAY[5:4]};
122         4'b0111:
123             selbcd = DAY[3:0];
124         4'b1000:
125             selbcd = {1'b0,1'b0,HOUR[5:4]};
126         4'b1001:
127             selbcd = HOUR[3:0];
128         4'b1010:
129             selbcd = {1'b0,MINUTE[6:4]};
130         4'b1011:
131             selbcd = MINUTE[3:0];
132         4'b1100:
133             selbcd = {1'b0,SECOND[6:4]};
134         4'b1101:
135             selbcd = SECOND[3:0];
136         default:
137             selbcd = 4'b0000;
138     endcase
139 endfunction
140
141 function isLeapYear;
142     input [15:0] year;
143
144     if (year[7:0] == 8'h00)
145         case (year[15:8])
146         8'h00: // year 0000
147             isLeapYear = 1'b1;
148         8'h04: // year 0400
149             isLeapYear = 1'b1;
150         8'h08: // year 0800
151             isLeapYear = 1'b1;
152         8'h12: // year 1200
```

```
153             isLeapYear = 1'b1;
154           8'h16: // year 1600
155             isLeapYear = 1'b1;
156           8'h20: // year 2000
157             isLeapYear = 1'b1;
158           8'h24: // year 2400
159             isLeapYear = 1'b1;
160           8'h28: // year 2800
161             isLeapYear = 1'b1;
162           8'h32: // year 3200
163             isLeapYear = 1'b1;
164           8'h36: // year 3600
165             isLeapYear = 1'b1;
166           8'h40: // year 4000
167             isLeapYear = 1'b1;
168           8'h44: // year 4400
169             isLeapYear = 1'b1;
170           8'h48: // year 4800
171             isLeapYear = 1'b1;
172           8'h52: // year 5200
173             isLeapYear = 1'b1;
174           8'h56: // year 5600
175             isLeapYear = 1'b1;
176           8'h60: // year 6000
177             isLeapYear = 1'b1;
178           8'h64: // year 6400
179             isLeapYear = 1'b1;
180           8'h68: // year 6800
181             isLeapYear = 1'b1;
182           8'h72: // year 7200
183             isLeapYear = 1'b1;
184           8'h76: // year 7600
185             isLeapYear = 1'b1;
186           8'h80: // year 8000
187             isLeapYear = 1'b1;
188           8'h84: // year 8400
189             isLeapYear = 1'b1;
190           8'h88: // year 8800
```

```
191             isLeapYear = 1'b1;
192         8'h92: // year 9200
193             isLeapYear = 1'b1;
194         8'h96: // year 9600
195             isLeapYear = 1'b1;
196         default:
197             isLeapYear = 1'b0;
198         endcase
199     else if (year[1:0] == 2'b00)
200         isLeapYear = 1'b1;
201     else isLeapYear = 1'b0;
202 endfunction
203 
204 always@(posedge CLOCK1M) begin
205     if (!nPRESET) begin // see if preset strobed
206         case (SELECT)
207         3'b000:
208             begin
209                 YEAR[15:8] <= VALUE[7:0];
210             end
211         3'b001:
212             begin
213                 YEAR[7:0] <= VALUE[7:0];
214             end
215         3'b010:
216             begin
217                 MONTH[4:0] <= VALUE[4:0];
218             end
219         3'b011:
220             begin
221                 DAY[5:0] <= VALUE[5:0];
222             end
223         3'b100:
224             begin
225                 HOUR[5:0] <= VALUE[5:0];
226             end
227         3'b101:
228             begin
```

```
229                 MINUTE[6:0] <= VALUE[6:0];
230             end
231         3'b110:
232             begin
233                 SECOND[6:0] <= VALUE[6:0];
234             end
235         default:
236             begin
237             end
238         endcase
239     end
240
241     else begin
242         if (counter == NCLKSEC) begin // see if 1sec event
243             for(i=0;i<NCOUNTER;i=i+1)
244                 counter[i] <= 1'b0;
245             if (SECOND == 7'b1011001) begin // see if second is 59
246                 SECOND <= 7'b0000000;
247                 if (MINUTE == 7'b1011001) begin // see if minute is 59
248                     MINUTE <= 7'b0000000;
249                     if (HOUR == 6'b100011) begin // see if hour is 23
250                         HOUR <= 6'b000000;
251                         if (MONTH == 5'b00010) begin // see if month is feb
252                             if ((isLeapYear(YEAR) && (DAY == 6'b101001)) ||
253                                 (!isLeapYear(YEAR) && (DAY == 6'b101000))) begin // see if day is 29th of lea
    p year or 28th
254                                 DAY <= 6'b000001;
255                                 MONTH[3:0] <= MONTH[3:0]+1'b1; // should become march 1st
256                             end
257                             else begin
258                                 if (DAY[3:0] == 4'b1001) begin // see if day is x9
259                                     DAY[3:0] <= 4'b0000;
260                                     DAY[5:4] <= DAY[5:4]+1'b1;
261                                 end
262                                 else begin // day is not x9
263                                     DAY[3:0] <= DAY[3:0]+1'b1;
264                                 end
265                             end
```

```
266                         end
267                         else begin // month is not feb
268                             if ((MONTH == 5'b00001) || (MONTH == 5'b00011) || (MONTH == 5'b00101) ||
269                                 (MONTH == 5'b00111) || (MONTH == 5'b01000) || (MONTH == 5'b10000) ||
270                                 (MONTH == 5'b10010)) begin // see if month is jan, march, may, july, aug, oct
    or dec
271                                 if (DAY == 6'b110001) begin // see if day is 31th
272                                     DAY <= 6'b000001; // should become 1st
273                                     if (MONTH == 5'b10010) begin // see if month is dec
274                                         MONTH <= 5'b00001; // should become jan
275                                         if (YEAR[3:0] == 4'b1001) begin // see if year is xxx9
276                                             YEAR[3:0] <= 4'b0000; // should become xxx0
277                                             if (YEAR[7:4] == 4'b1001) begin // see if year is xx99
278                                                 YEAR[7:4] <= 4'b0000; // shoud become xx00
279                                                 if (YEAR[11:8] == 4'b1001) begin // see if year is x999
280                                                     YEAR[11:8] <= 4'b0000; // should become x000
281                                                     if (YEAR[15:12] == 4'b1001) begin // see if year is 9999
282                                                         YEAR[15:12] <= 4'b0000; // should become 0000
283                                                     end
284                                                     else begin
285                                                         YEAR[15:12] <= YEAR[15:12]+1'b1;
286                                                     end
287                                                 end
288                                                 else begin
289                                                     YEAR[11:8] <= YEAR[11:8]+1'b1;
290                                                 end
291                                             end
292                                             else begin
293                                                 YEAR[7:4] <= YEAR[7:4]+1'b1;
294                                             end
295                                         end
296                                         else begin // year is not xxx9
297                                             YEAR[3:0] <= YEAR[3:0]+1'b1;
298                                         end
299                                     end
300                                     else begin // month is not dec
301                                         MONTH[3:0] <= MONTH[3:0]+1'b1;
302                                     end
```

```
303                             end
304                             else begin // day is not 31th
305                                 if (DAY[3:0] == 4'b1001) begin // see if day is x9
306                                     DAY[3:0] <= 4'b0000;
307                                     DAY[5:4] <= DAY[5:4]+1'b1;
308                                 end
309                                 else begin // day is not x9
310                                     DAY[3:0] <= DAY[3:0]+1'b1;
311                                 end
312                             end
313                         end
314                         else begin // month is arp, jun, sep, nov
315                             if (DAY[5:0] == 6'b110000) begin // see if day is 30th
316                                 DAY[5:0] <= 6'b000001; // should become 1st
317                                 if (MONTH[4:0] == 5'b01001) begin // see if month is sep
318                                     MONTH[4:0] <= 5'b10000; // should become oct
319                                 end
320                                 else begin // month is not sep
321                                     MONTH[3:0] <= MONTH[3:0]+1'b1; // of next month
322                                 end
323                             end
324                             else begin // day is not 30th
325                                 if (DAY[3:0] == 4'b1001) begin // see if day is x9th
326                                     DAY[3:0] <= 4'b0000; // should become x0th
327                                     DAY[5:4] <= DAY[5:4]+1'b1;
328                                 end
329                                 else begin // day is not x9th
330                                     DAY[3:0] <= DAY[3:0]+1'b1;
331                                 end
332                             end
333                         end
334                     end
335                 end
336                 else begin
337                     if (HOUR[3:0] == 4'b1001) begin // see if hour is x9
338                         HOUR[3:0] <= 4'b0000; // should become x0
339                         HOUR[5:4] <= HOUR[5:4]+1'b1;
340                     end
```

```
341                         else begin // hour is not x9
342                             HOUR[3:0] <= HOUR[3:0]+1'b1;
343                         end
344                     end
345                 end
346                 else begin // minute is not 59
347                     if (MINUTE[3:0] == 4'b1001) begin // see if minute is x9
348                         MINUTE[3:0] <= 4'b0000; // should become x0
349                         MINUTE[6:4] <= MINUTE[6:4]+1'b1;
350                     end
351                     else begin // minute is not x9
352                         MINUTE[3:0] <= MINUTE[3:0]+1'b1;
353                     end
354                 end
355             end
356             else begin // second is not 59
357                 if (SECOND[3:0] == 4'b1001) begin // see if second is x9
358                     SECOND[3:0] <= 4'b0000; // should become x0
359                     SECOND[6:4] <= SECOND[6:4]+1'b1;
360                 end
361                 else begin // second is not x9
362                     SECOND[3:0] <= SECOND[3:0]+1'b1;
363                 end
364             end
365         end
366         else begin // not 1sec event
367             counter <= counter+1'b1;
368         end
369     end
370     if (dispcnt == NCLKDISP) begin // see if display change event
371 
372         for(i=0;i<NDISPCNT;i=i+1)
373             dispcnt[i] <= 1'b0;
374 
375         if (dispsel == 4'b1101) begin // see if dispsel is 13
376             dispsel <= 4'b0000;
377         end
378         else begin // dispsel is not 13
```

```
379             dispsel <= dispsel+1'b1;
380         end
381     end
382     else begin // not display change event
383         dispcnt <= dispcnt+1'b1;
384     end
385 end
386 
387 assign nDIGIT = digit(dispsel);
388 assign SEG = bcddecode(selbcd(dispsel));
389 
390 endmodule
391 
392 
```