پروژه فرکانس متر

سیاوش دوانی

محمدجواد حاجی زاده

گروه 20

عملکرد مدار به این صورت است که یک شمارنده تعداد posEdge های کلاک که بین دو posEdge سیگنال ورودی رخ داده است را می شمرد. سپس دوره ی سیگنال ورودی با تناسب بدست می آید.

توجه شود که فرکانس کلاک خیلی بالاتر از فرکانس سیگنال است. برای این کار از کلاک خود FPGA استفاده شد که 10MHz است.

بازه ی فرکانس هایی که قرار است اندازه گیری شود بین 100 هرتز تا 10 هزار هرتز است. دوره باید با دو رقم و به میلی ثانیه نمایش داده شود.

بلاک always اصلی به این صورت است

```
always @(posedge clk) begin
  //// counter logic
  if (signalPast == 6'b000000 && signal == 1) begin     /// if input signal is at posEdge
    counted = counter;                                  ///   save counter value
    counter <= 0;                                       ///   reset counter
  end else begin                                        /// else
    counter <= counter + 1;                             ///   count
  end                                                   ///
  //// Storing signal last 6 values
  signalPast = signalPast << 1;
  signalPast[0] = signal;
  //// Creating c signal for showing numbers on 7seg
  c <= c+1;
  /////////// Show on 7seg
  case (c[15:14])
    2'b00: begin d[1]<=1'b1; d[2]<=1'b0; d[3]<=1'b1; d[4]<=1'b1; w=4'b1111; z[0] <= isLowPeriod ? 1 : 0; end
    2'b01: begin d[1]<=1'b1; d[2]<=1'b1; d[3]<=1'b0; d[4]<=1'b1; w=x[7:4];  z[0] <= isLowPeriod ? 0 : 1; end
    2'b10: begin d[1]<=1'b1; d[2]<=1'b1; d[3]<=1'b1; d[4]<=1'b0; w=x[3:0];  z[0] <= 0;                    end
  endcase
  case (w)
    4'b0000 :  z[7:1] <= 7'b1111110;
    4'b0001 :  z[7:1] <= 7'b0110000;
    4'b0010 :  z[7:1] <= 7'b1101101;
    4'b0011 :  z[7:1] <= 7'b1111001;
    4'b0100 :  z[7:1] <= 7'b0110011;
    4'b0101 :  z[7:1] <= 7'b1011011;
    4'b0110 :  z[7:1] <= 7'b1011111;
    4'b0111 :  z[7:1] <= 7'b1110000;
    4'b1000 :  z[7:1] <= 7'b1111111;
    4'b1001 :  z[7:1] <= 7'b1111011;

    4'b1111 :  z[7:1] <= 7'b0000000;
  endcase
end
```

چالش اولیه این بود که چگونه لبه ی سیگنال ورودی را تشخیص دهیم.

برای این کار ابتدا یک رجیستر برای ذخیره کردن مقادیر گذشته سیگنال قرار دادیم. سپس برای شناسایی لبه سیگنال به عنوان مثال اگر چند مقدار نمونه برداری شده قبلی 0 باشد و مقدار فعلی 1 باشد لبه ی بالا رونده سیگنال است.

به این خاطر چندین مقدار قبل را ذخیره میکنیم چون اگر هنگام تغییر از 0 به 1 نوسان وجود داشته باشد تشخیص را دچار مشکل نکند.

دو بخش دیگر این بلاک یکی ساختن رجیستر مقادیر قبلی و دیگری پیاده سازی تکنیک نمایش روی 7seg های FPGA است که در آن دو متغیر x و isPeriodLow است که در ادامه توضیح داده میشود.

بلاک always بعدی به این صورت است

```
////////////////////////////// Period Computing
always @(counted) begin
  period = counted / 10;
  isLowPeriod = period < 1000;
end
```

که در هر بار نمونه برداری از شمارنده در هنگام لبه سیگنال، دوره و متغیر مقایسه گر برای نمایش به دو صورت مختلف را محاسبه میکند.

دوره به میکروثانیه محاسبه و ذخیره میشود

بلاک بعدی پس از هربار محاسبه دوره، دو رقم BCD مورد نیاز برای نمایش را از روی دوره ی محاسبه شده و isLowPeriod محاسبه و در x ذخیره میکند

```
////////////////////////////// BCD Computing
always @(period) begin
  if (isLowPeriod) begin
    ////  957 -> .95
    x[7:4] = (period       - period % 100)/100;
    x[3:0] = (period % 100 - period % 10 )/10 ;
  end else begin
    //// 3589 -> 3.5
    x[7:4] = (period        - period % 1000)/1000;
    x[3:0] = (period % 1000 - period % 100 )/100 ;
  end
end
```

به این ترتیب مشخص میشود که میتوان با استفاده از متغیر isLowPeriod مکان نقطه را که با z[0] نمایش داده میشود در هنگام نمایش در بلاک always اصلی مشخص کرد

به این صورت

```
//// Creating c signal for showing numbers on 7seg
c <= c+1;
////////// Show on 7seg
case (c[15:14])
  2'b00: begin d[1]<=1'b1; d[2]<=1'b0; d[3]<=1'b1; d[4]<=1'b1; w=4'b1111; z[0] <= isLowPeriod ? 1 : 0; end
  2'b01: begin d[1]<=1'b1; d[2]<=1'b1; d[3]<=1'b0; d[4]<=1'b1; w=x[7:4];  z[0] <= isLowPeriod ? 0 : 1; end
  2'b10: begin d[1]<=1'b1; d[2]<=1'b1; d[3]<=1'b1; d[4]<=1'b0; w=x[3:0];  z[0] <= 0;                   end
endcase
```

کد به صورت زیر است

```
module freqMeter(clk, signal, d, z);
 input clk;                              ///////  FPGA Clock at 10MHz
 input signal;                           ///////  Input signal at P97
 output reg [1:4]d;                      ///////  7seg  Enables
 output reg [7:0]z;                      ///////  7seg  LEDs     z = a_b_c_d_e_f_g_point
 reg [19:0]counter = 0;                  ///////  posEdge counter
 reg [19:0]counted = 10000;              ///////  Last Counted posEdges
 reg [15:0] period = 1000;               ///////  Computed period in micro seconds
 reg isLowPeriod   = 0;                  ///////  True if Period < 1000
 reg [7:0] x;                            ///////  two BCD Numbers
 reg [5:0]signalPast = 6'b010101;        ///////  signal last 6 values

 //// Variables used for showing numbers on 7segs
 reg [15:0] c=0;
 reg [3:0] w;

 ////////////////////////////// Main Program logic
 always @(posedge clk) begin
  //// counter logic
  if (signalPast == 6'b000000 && signal == 1) begin     /// if input signal is at posEdge
    counted = counter;                                  ///   save counter value
    counter <= 0;                                       ///   reset counter
  end else begin                                        /// else
    counter <= counter + 1;                             ///   count
  end
```

```
//// Storing signal last 6 values
signalPast = signalPast << 1;
signalPast[0] = signal;

//// Creating c signal for showing numbers on 7seg
c <= c+1;
////////// Show on 7seg
case (c[15:14])
  2'b00: begin
        d[1]<=1'b1; d[2]<=1'b0; d[3]<=1'b1; d[4]<=1'b1; w=4'b1111; z[0] <= isLowPeriod ? 1 : 0;
   end
  2'b01: begin
        d[1]<=1'b1; d[2]<=1'b1; d[3]<=1'b0; d[4]<=1'b1; w=x[7:4];  z[0] <= isLowPeriod ? 0 : 1;
   end
  2'b10: begin
        d[1]<=1'b1; d[2]<=1'b1; d[3]<=1'b1; d[4]<=1'b0; w=x[3:0];  z[0] <= 0;
  end
endcase

case (w)
  4'b0000 :  z[7:1] <= 7'b1111110;
  4'b0001 :  z[7:1] <= 7'b0110000;
  4'b0010 :  z[7:1] <= 7'b1101101;
  4'b0011 :  z[7:1] <= 7'b1111001;
  4'b0100 :       z[7:1] <= 7'b0110011;
  4'b0101 :       z[7:1] <= 7'b1011011;
  4'b0110 :       z[7:1] <= 7'b1011111;
  4'b0111 :       z[7:1] <= 7'b1110000;
  4'b1000 :  z[7:1] <= 7'b1111111;
```

```
    4'b1001 :  z[7:1] <= 7'b1111011;

    4'b1111 :  z[7:1] <= 7'b0000000;
   endcase
  end

  ////////////////////////////// Period Computing
  always @(counted) begin
   period = counted / 10;
   isLowPeriod = period < 1000;
  end

  ////////////////////////////// BCD Computing
  always @(period) begin
   if (isLowPeriod) begin
    ////  957 -> .95
    x[7:4] = (period       - period % 100)/100;
    x[3:0] = (period % 100 - period % 10 )/10 ;
   end else begin
    //// 3589 -> 3.5
    x[7:4] = (period        - period % 1000)/1000;
    x[3:0] = (period % 1000 - period % 100 )/100 ;
   end
  end
endmodule
```