```
1   /*-------------------------------------------------------------------------------
2   Name Lamin Jammeh
3   CLass: EE417 Summer 2024
4   Lesson 09 HW Question 2
5   Group: Ron Kalin/ Lamin Jammeh
6   Project Description: This module is Differentiator, it finds the difference between
7   adjacent sample
8   -------------------------------------------------------------------------------*/
9
10  module Differentiator #(parameter word_size = 8)
11                         (
12                          output  [word_size-1:0] Data_out,
13                          input   [word_size-1:0] Data_in,
14                          input                   hold, clock, reset
15                         );
16
17  // Internal registers and wires
18  reg [word_size-1:0] buffer; // Internal register for buffer
19
20  // Assign the formula for Data_out
21  assign Data_out = Data_in - buffer;
22
23  always @(posedge clock)
24     begin
25        if (reset)
26
    /*-------------------------------------------------------------------------------------
    --------
27           @ high reset the following will occur
28           ** buffer goes to zero
29           ** Data_in will keep its value since it is an external input but the buffer will
    remain zero
30           ** Data_out will still be [Data_in - buffer] but since buffer is zero therefore
    Data_out will be Data_in
31           ** This Means that no filtering is occurring at high reset
32
    -------------------------------------------------------------------------------------
    --------*/
33
34            buffer <= 0; // When reset is high, and clock is rising, buffer <= 0
35        else if (hold)
36           /*-------------------------------------------------
37           @ low rest and hold = 1 the following will happen
38           **buffer Keeps it current value
39           ** Data_in is not transfered to buffer
40           ** Data_out is still equal to [Data_in - buffer] but
41           ** therefore @ hold = 1 the buffer is not updated
42           -------------------------------------------------*/
43            buffer <= buffer; // When hold is high, reset is low, and clock is rising, buffer
    remains unchanged
44        else
45            /*-------------------------------------------------------------
46            @ hold = 0 and reset = 0 the following will occur
47            ** buffer will store the previous Data_in value
48            ** Data_out will be (Data_in - buffer)
49            ** therefore the buffer is update when hold is low and reset is low
50            -------------------------------------------------------------*/
51            buffer <= Data_in; // When hold is low, reset is low, and clock is rising, buffer
    <= Data_in
52     end
53
54  endmodule
```