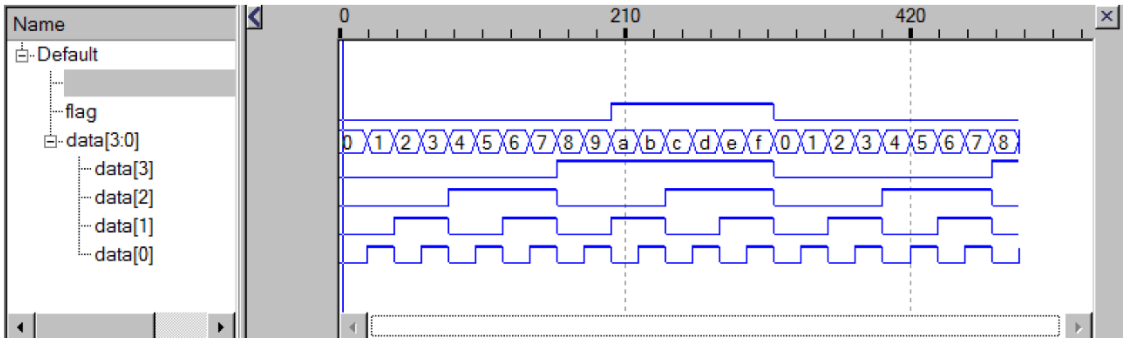
BCD checker example:

Consider the following testbench code and the corresponding simulation results.

```
module t_BCD_checker ();
      wire         flag;
      reg   [3:0] data;
      integer      k;
      BCD_checker     UUT    (data, flag); i
      initial #500 $finish;       // finish simulation after 500 time units (ns)
      initial begin
                  data = 0;
                  forever   begin
                              #10 data = data+1;  end
            end
endmodule
```



(a) Write down the Verilog code for the module **BCD_checker** using a **case** structure.

```
//-------------------------------------------------------------
// The BCD_cheker design using a case structure:
//-------------------------------------------------------------

module BCD_checker_case (data, flag);

input [3:0]  data;
output reg   flag;

always @ (data)

case(data)
   10, 11, 12, 13, 14, 15 : flag = 1'b1;
   default                : flag = 1'b0;
   endcase

endmodule
```

(b) Design the **BCD_checker** as a primitive. Make any necessary adjustment that you need.

```
//-------------------------------------------------------------
// The BCD_checker design using a primitive:
//-------------------------------------------------------------

primitive BCD_primitive (output flag, input d3,d2,d1,d0);

table

 //  d3   d2   d1   d0   :   flag
     0    0    0    0    :   0;
     0    0    0    1    :   0;
     0    0    1    0    :   0;
     0    0    1    1    :   0;
     0    1    0    0    :   0;
     0    1    0    1    :   0;
     0    1    1    0    :   0;
     0    1    1    1    :   0;
     1    0    0    0    :   0;
     1    0    0    1    :   0;
     1    0    1    0    :   1;
     1    0    1    1    :   1;
     1    1    0    0    :   1;
     1    1    0    1    :   1;
     1    1    1    0    :   1;
     1    1    1    1    :   1;

endtable

endprimitive


module BCD_checker (data, flag);     // wrapper module

input [3:0]  data;
output       flag;

BCD_primitive   Mo    (flag, data[3],data[2],data[1],data[0]);

endmodule
```