```
1    `timescale 1 ms / 100 us
2
3    module camera_control(init, reset, clk, exp_inc, exp_dec, nre1, nre2, adc,
     expose, erase);
4
5        input init, reset, clk, exp_inc, exp_dec;
6
7        output nre1, nre2, adc, expose, erase;
8
9        wire init, reset, clk, exp_inc, exp_dec;

10
11
12
13       reg nre1, nre2, adc, expose, erase;
14       reg exp_finished, adc_conversion_finished, adc_conversion_state;
     //Exp_finished = TRUE when the camera is finished exposing,
     adc conversion finished = TRUE when the ADC-conversion is done
15       //adc_conversion_state = TRUE when the first inputs have been read and
     translated
16
17       parameter state_array_size = 2, exp_array_size = 5;         //The
     number of states is 3, so 2 bit is designated for the state-coutner. The
     exposure-time is max. 30ms, so a 5-bit register is needed.
18       parameter num_adc_read_cycles = 3, adc_array_size = 2;      //The
     number of clock cycles the ADC has to read the input. Assumed to be 3ms.
     The array size for the adc-counter is 2-bit.
19       parameter idle = 2'b00, capture = 2'b01, convert = 2'b11;   //Defining
     the different states of the state-machine.
20
21       reg [state_array_size-1:0] state;                           //Array
     for the different states
22       reg [exp_array_size-1:0] exp_time;                          //Array
     for the exposure time
23       reg [exp_array_size-1:0] exp_count;                         //Array
     for the exposure time counter
24       reg [adc_array_size-1:0] adc_read_cycle;                    //Array
     for the number of read-cycle counter for the ADC
25
26       always @(posedge clk)
27           begin: FSM
28               if (reset == 1'b1) begin                              //Reset
     is active-high. Resets all output variables.
29                       $display ("Reset enabled");
30                       state <= idle;
31                       nre1 <= 1;
32                       nre2 <= 1;
33                       adc <= 0;
34                       expose <= 0;
35                       erase <= 1;
36                       exp_time <= 15;
37               end else
38                   case(state)
39                       idle:
40                           if (init == 1'b1 && reset == 1'b0) begin

41                                   $display ("Initialising");
     //Change state to Capture when idle is high.
42                                   exp_finished <= 0;
     //Reset all internal counters when initiating.
```

```
43                                  exp_count <= 0;
44                                  adc_conversion_finished <= 0;
45                                  adc_conversion_state <= 0;
46                                  adc_read_cycle <= 0;
47                                  state <= capture;
48                              end else if (exp_inc == 1'b1 && exp_dec == 1'b0 &&
    init == 1'b0) begin      //Increase the exposure time with 1ms every clock
    cycle the button is pressed
49                                  if (exp_time < 30) begin
    //as long as the exposure time is smaller than 30.
50                                      $display ("Exposure time increased
    by 1");
51                                      exp_time <= exp_time + 1;

52                                  end
53                              end else if (exp_dec == 1'b1 && exp_inc == 1'b0 &&
    init == 1'b0) begin      //Decrease the exposure time with 1ms every clock
    cycle the button is pressed
54                                  if (exp_time > 2) begin
    //as long as the exposure time is bigger than 2.
55                                      $display ("Exposure time decreased
    by 1");
56                                      exp_time <= exp_time - 1;
57                                  end
58                              end else begin
59                                  $display ("State: Idle");
    //Enable erase when idle.
60                                  erase <= 1;
61
62                              end
63                          capture:
64                              if (exp_finished == 0) begin
65                                  $display ("Expose = 1, Erase = 0");
66                                  expose <= 1;
    //The camera is in expose-mode until the counter exp_count has reached the
    specified exposure time.
67                                  erase <= 0;
68                                  if (exp_count < exp_time) begin
    //The count can be done every clock cycle as long as the clock frequency
    is equal to the change in time.
69                                      exp_count <= exp_count + 1;
70                                      $display ("Exposing");
71                                  end else if (exp_count >= exp_time) begin

72                                      expose <= 0;
73                                      exp_finished <= 1;
74                                      $display("Exposure finished");
75                                      state <= convert;
    //After the exposure is done the state-machien goes into the conversion
    state.
76                                  end
77                              end
78                          convert:
79                              if (adc_conversion_finished == 0) begin
    //As long as the conversion is not done, the program converts from the two
    rows of
80                                  $display("Converting");
    //amplifiers. Which row is read is controlled by adc_conversion_state
    which is 0 for row
81                                  if (adc_conversion_state == 0) begin
    //1 and 1 for row 2.
```

```
82                                          $display("Converting row 1");
83                                          if (adc_read_cycle <
    num_adc_read_cycles) begin             //The ADC has 5 clock cycles to
    read the output of the amplifiers. This is not required

84                                                  if (adc_read_cycle == 1)
    begin                              //by the assignment. When the counter has
    not reached the limit, the ADC and NRE1/2 signal
85                                                          adc <= 1;
    //is active. (The NRE gets 0 as it is a PMOS).
86                                                  end else begin
87                                                          adc <= 0;
88                                                      end
89                                                  nre1 <= 0;

90                                                  adc_read_cycle <=
    adc_read_cycle + 1;
91                                          end else if (adc_read_cycle >=
    num_adc_read_cycles) begin   //After the ADC has read for 5 cycles,  the
    ADC is turned off to give it a rest between reading

92                                                  nre1 <= 1;
    //samples. The conversion state is changed and the cycle counter is reset.
93                                                  adc_conversion_state <= 1;
94                                                  adc_read_cycle <= 0;

95                                              end
96                                      end
97                                  if (adc_conversion_state == 1) begin
    //The same as row 1, but after it is finished reading the ADC-conversion
    is complete
98                                          $display("Converting row 2");
    //and the state is changed to idle.
99                                          if (adc_read_cycle <
    num_adc_read_cycles) begin
100                                                 if (adc_read_cycle == 1)
    begin
101                                                         adc <= 1;
102                                                 end else begin
103                                                         adc <= 0;
104                                                     end
105                                                 nre2 <= 0;
106                                                 adc_read_cycle <=
    adc_read_cycle + 1;
107                                         end else if (adc_read_cycle >=
    num_adc_read_cycles) begin
108                                                 nre2 <= 1;
109                                                 adc_conversion_finished <=
    1;
110                                             end
111                                     end
112                             end
113                         else if (adc_conversion_finished == 1) begin
114                                 $display("Conversion finished");
    //After the conversion is finished the state goes to idle again, resetting
    the counters and erasing
115                                 state <= idle;
    //the charge from the capacitors.
116                                 erase <= 1;
117                             end
```

```
118                     endcase
119             end
120
121
122     endmodule
```