TFE4152 Design of Integrated Circuits

Digital Camera

Sara Roberg Ghabeli, Sigurd Hagen Tullander

Fall 2020

1 Summary

In this project we have designed a digital camera controller and an analog exposure and readout circuit for a 2x2 pixel digital camera. The implemented digital camera controller allows the user to adjust exposure time, initiate exposure and reset the camera. The individual pixel circuits contain NMOS transistors used as switches for the digital circuit to control, a photodiode to sense light, and a capacitor that the photodiode to charges up for the digitally set exposure time. The switches were dimensioned to minimize leakage current, and the capacitor's capacitance was chosen for maximal dynamic range. The analog circuit writes the first row of pixel voltages to two ADCs before writing the second row of pixel voltages to the same ADCs. The camera controller is designed to ensure that the voltages over the pixel capacitors are erased before a new exposure is initiated. Through Verilog and AIM-Spice simulations, it is shown that the camera works according to specifications. However, if process variations make the pixel switch transistors faster than typical, the images will be extremely poorly exposed due to vast leakage currents.

Contents

2 Introduction

A digital camera can be divided into three base parts: a digital circuit taking user input and controlling the camera, an analog circuit responsible for turning varying light into varying voltages for each pixel of the camera, and another digital circuit processing and applying the voltages into a digital image. In this report, we will adress the design, simulation and verification of the digital camera controller and the analog circuit for a 2x2 pixel digital camera. The camera controller is a fully digital system that take input from 2 buttons to increase and decrease the exposure time, 1 button to reset the camera and 1 button to take a picture. The camera controller will based on this information provide the necessary control signals for the analog circuit to control the camera according to the specifications. The analog circuit is as the name suggests a fully analog system that captures light intensity in capacitors and send data about the image taken to an ADC. What happens with the signal after being sent to the ADC is beyond the scope of this report.

3 Theory

3.1 Analog circuit

The analog circuit has input signals delivered by the digital circuitry, and delivers its output to two ADCs. The analog circuit diagram is given in figure 1, where each of the four pixel block has the circuit given in figure 2.

In detail, the analog circuit consists of 4 pixels in a 2x2 array. Each pixel in the same row shares the same *NRE* (Not-REad) input which means they are sending data to the ADC at the same time. Each pixel in the same column share the same active load and readout wire, which means they need to send data to the ADC at different times.

3.1.1 Conceptual operation of pixel circuit

Schematics for the pixel circuit is given in figure 2. It consists of a photo diode PD1, three switch transistors M1, M2 and M4, one buffer transistor M3 and a charge storage capacitor C_S. The photo diode is modeled by a diode and an ideal current source in parallel. The current source outputs a current I_D that is proportional to the illumination intensity on that particular pixel. This current is used to charge the transistor C_S. Ideally, we want that all of I_D flows through M1 and into C_S when *EXPOSE* is `HIGH` and all of I_D flows back through the diode in the photo diode model when *EXPOSE* is `LOW`. In that way the voltage on node N2 when *EXPOSE* has been `HIGH` for a time t is given by

$$V_{CS}(t) = \frac{1}{C_S} \int_0^t I_D(t)dt. \tag{1}$$

This means the voltage V_{CS} is proportional to the total illumination on the pixel throughout

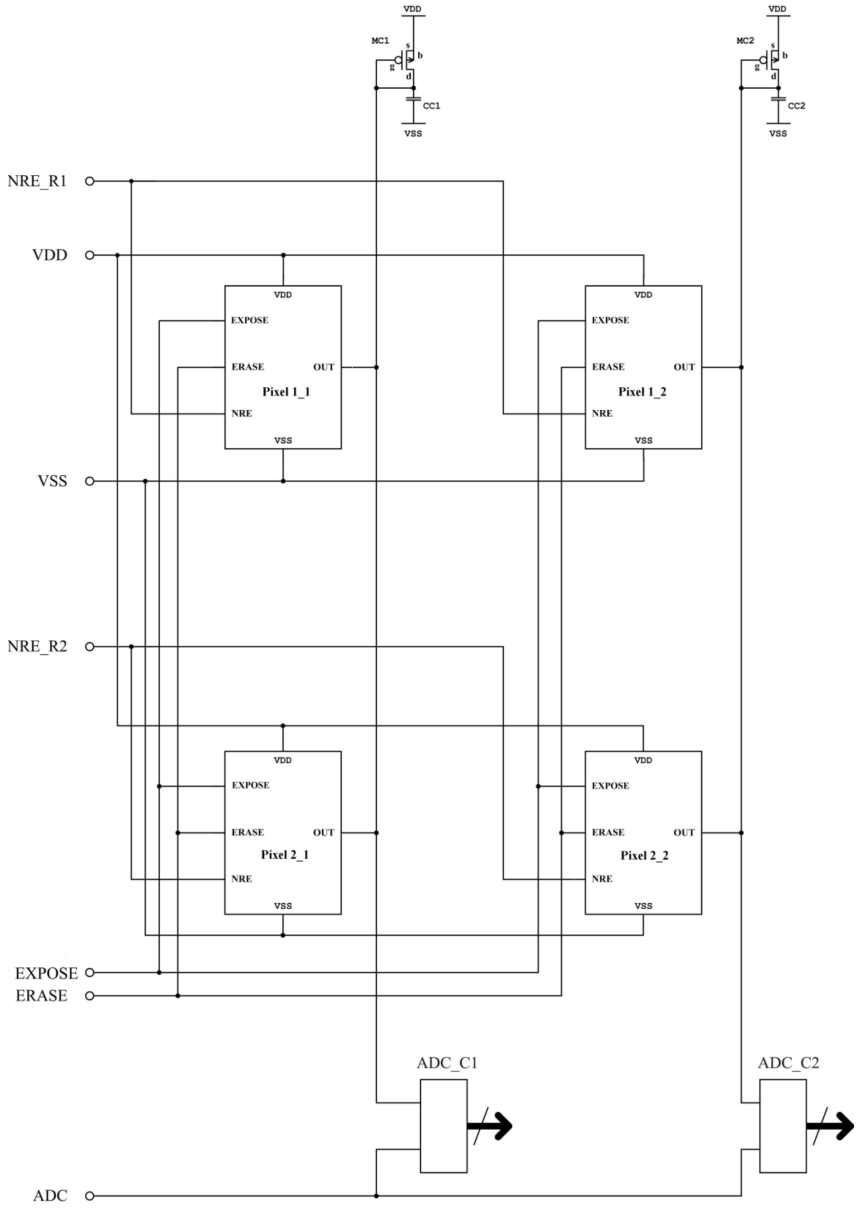

Figure 1: Analog circuit schematic.

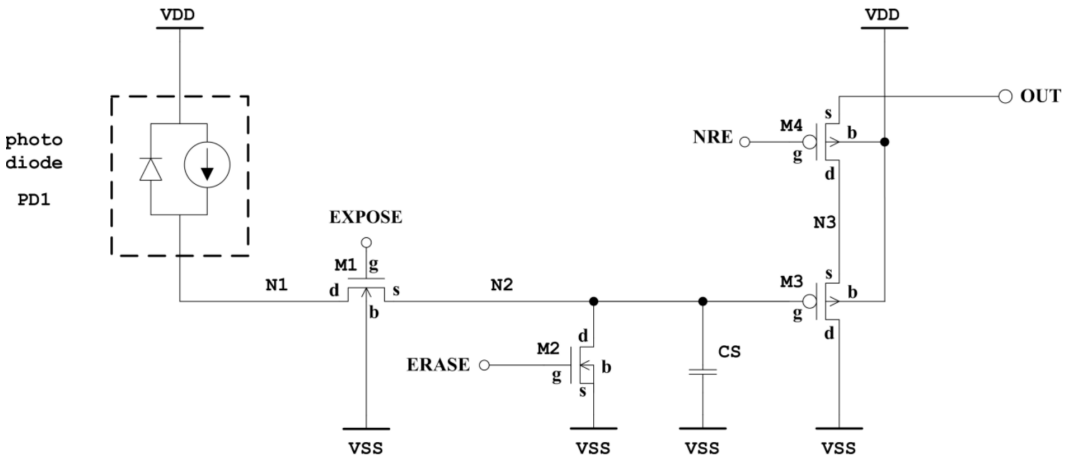

Figure 2: Pixel cicuit schematic.

the exposure time, which is exactly what we want.

But since the camera is supposed to be reusable, we need a way to erase that charge before taking a new picture. That is what M2 is for. Ideally we want M2 to be a perfect switch, so that when *ERASE* is `HIGH`, C_S is instantly uncharged through M2 and when *ERASE* is `LOW`, M2 does not conduct any current at all.

For the readout we do not want to connect N2 directly to OUT, since the output wire might be very long and have a big capacitance. Instead, the voltage V_{CS} controls the conductance through a transistor M3. The higher V_{CS}, the lower conductance through M3 and higher voltage on OUT. In that way, the current required to control the voltage on the output wire is supplied through the active load in the top of each output wire, outside the pixel circit.

But since all pixels in the same column share the same output wire we need a way to unconnect N3 from OUT, and that is what M4 is for. M4 is supposed to be an ideal switch so that the pixel is driving OUT only when *NRE* is `LOW`.

3.1.2 Dimensions for switch transistors

The transistors M1, M2 and M4 all function as switches, where a digital gate input decides whether the transistor should act as a short-circuit between drain and source, or as an open circuit. Since these transistors simply should have two possible states, it is key that the leakage current is minimized. This is particularly important for M1 and M2 to ensure that the voltage over C_s is as constant as possible during readout. In *Analog Circuit Design* by Tony Chan Carusone, one can read in section 1.4.1 that the subthreshold leakage current is given by

$$I_{off} = (n-1)\mu_n C_{ox}\left(\frac{W}{L}\right)\left(\frac{kT}{q}\right)^2 \exp\left(-qV_t/nkT\right). \quad (2)$$

In order to minimize this leakage current, $\frac{W}{L}$ need to be as small as possible, meaning the smallest possible width W and the largest possible length L is desired. The technology used will limit the possible width and length of the transistor, thus W and L can be chosen accordingly.

3.1.3 Value for C_s

To choose a suitable value for C_s spice simulations can be used. To know what values are the best we will look at the four corner cases for exposure time and light conditions. The corner cases will be denoted exposure-light, so for example max-min is maximum exposure time and minimum light. The voltage over C_S after exposure in each corner will get different values for different values of C_S. There are many possible approaches to how these corners should be tuned, and the one that we will apply is the following:

- Max-max corner should make C_s fully charged.
- Min-min corner should leave C_s uncharged.
- Min-max and max-min corners should make C_s half full charged.

The reason for why we want the corners to be like that is beyond the scope of this report.

3.1.4 Values for M3 and active load

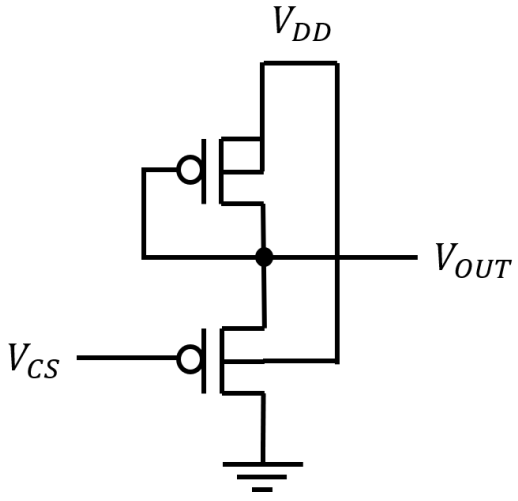

Figure 3: Simplified schematic for output voltage driving circuit.

Values for M3 and active load should be tuned for maximum dynamic range, but it will not have a very big impact since it is a very robust and self-regulating system. If we assume C_{C1} and C_{C2} are very small, the output system can be simplified to figure 3. Essentially, the output voltage V_{OUT} depends on how the conductance in the uppermost and downmost transistor are relative to each other. These conductances are proportional to W/L in the transistors and they also vary with the gate-source voltages. To avoid unnecessary simplifications, it is best to decide these W/L-ratios from SPICE-simulations. The dynamic range is the difference between the output voltage when uncharged and fully charged, and it is preferable that it is as big as possible.

3.1.5 Process variations

It is important to look at how process variations will impact an analog design. One way to do this is to look at the what is called FF, FS, SF and SS corners. F means fast, S means slow, the first letter is for NMOS and the second is for PMOS. For this design it is enough to look at how the switch transistors are affect by process variations. The values to measure in each corner is R_{DS} when switched off and on.

For curiosity we will also have a look at how these two corners affect the analog circuit as a whole.

3.2 Digital camera controller

The digital input signals in the analog circuit need to be controlled by a digital circuit - the digital camera controller. The module interface for the digital camera controller is described in the figure below.

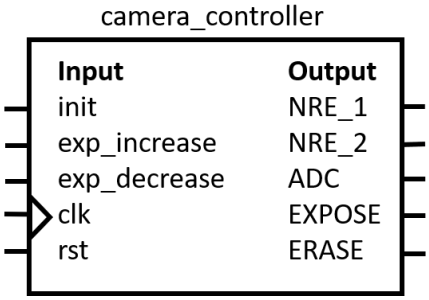

Figure 4: Overview of input and output pins for the camera controller module.

It has inputs *init*, *exp_increase*, *exp_decrease*, *clk* and *rst*. *clk* is controlled by an internal camera module, while the four other signals are controlled by the user. *init* is `HIGH` when the user presses the shutter button, *exp_increase* is `HIGH` when the user wants to increase the exposure time, *exp_decrease* is `HIGH` when the user wants to decrease the exposure time (and is overridden by *exp_increase* if they conflict, i.e. if they are `HIGH` at the same time),

and *rst* is `HIGH` when the user wishes to cancel any ongoing process, or reset the exposure time.

The outputs are *NRE_1*, *NRE_2*, *ADC*, *EXPOSE* and *ERASE*, all of which were described in section 3.1.

3.2.1 Finite state machine

The wanted behaviour of the digital camera controller is described in the finite state machine (FSM) in figure 5.

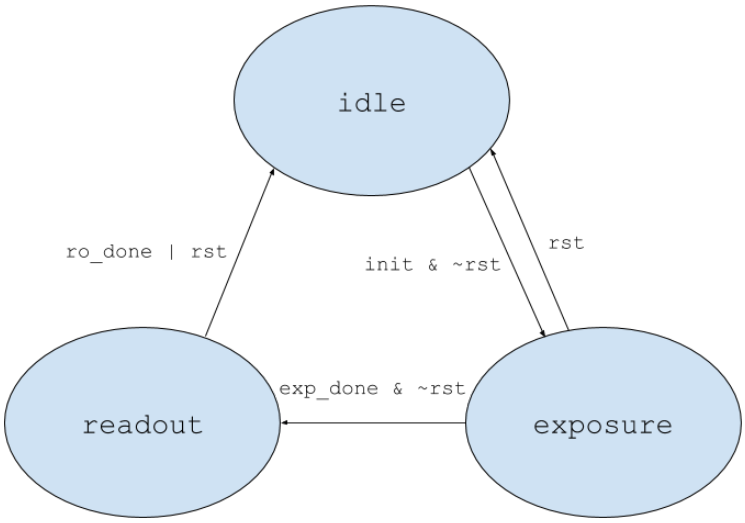

Figure 5: The finite state machine for the digital camera controller.

In the FSM, there are three states. The camera starts in the *idle* state. In this state, the camera isn't in the process of taking any picture, so output signals *EXPOSE* and *ADC* should be `LOW`, and *NRE_1* and *NRE_2* should be `HIGH`. The camera should also prepare itself for a new picture in this state, so *ERASE* should be `HIGH` to erase any voltage from the potential previous exposure. The user should be able to adjust the exposure time with the signals *exp_increase* and *exp_decrease* in the *idle* state. These signals will respectively increase or decrease the exposure time with 1ms for each clock cycle they're `HIGH` to a minimum of 2ms and a maximum of 30ms. When the signal *init* is `HIGH`, the camera will enter the *exposure* state. The exception is if the *rst* input signal is `HIGH`. In any state, *rst* being `HIGH` should override any process and take the camera back to *idle*.

When the camera is entering the *exposure* state, *ERASE* should be set `LOW` and *EXPOSE* should be set `HIGH` to begin the exposure. If *rst* is `HIGH` at any time during *exposure*, the camera should return to the *idle* state. When the preset exposure time has passed, the camera should enter the *readout* state if *rst* is `LOW`. Exposure being finished is called *exp_done* in figure 5.

The *readout* state is the final state, in which the data from the individual pixels are read and inputted in the ADC. First, *EXPOSE* should be set `LOW` since the exposure is done. Then, *NRE_1* should be `LOW` first at the same time as *ADC* is `HIGH`, to read from the top two pixels and input it to the ADC. Lastly, *NRE_2* should be `LOW` at the same time as *ADC* is `HIGH` to do the same for the bottom two pixels. Exactly how the output signals should be will be properly explored in the next section. When the readout sequence is finished, called *ro_done* in figure 5, the camera should enter the *idle* state again. As with the *exposure* state, *rst* being `HIGH` at any time should return the camera to *idle*. It is important that the camera stays in *idle* for at least one clock cycle before the user is able to take another picture, to ensure that the data of the previous photo is erased.

3.2.2 Output timing

The desired time chart is illustrated in figure 6.

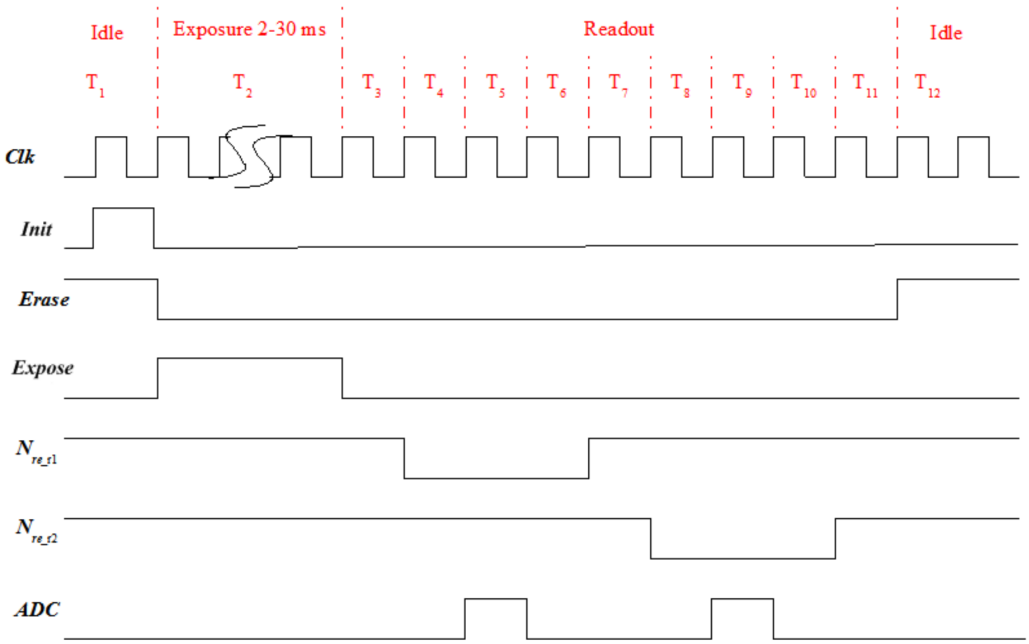

Figure 6: The desired time chart for the output signals of the digital camera controller.

In the *idle* state and *exposure* state, the output signals are as described in the previous subsection. Notably, *ERASE* is `HIGH` during *idle* and *EXPOSE* is `HIGH` during *exposure*. The *readout* state is more complex, as output signals are altered in every clock cycle. In the first cycle, T_3 in the figure, *EXPOSE* is set `LOW` to end exposure before readout begins. From T_4 to T_6, *NRE_1* is set `LOW`, with *ADC* being `HIGH` in the middle T_5. This is to ensure that *NRE_1* is not transienting from `HIGH` to `LOW` or `LOW` to `HIGH` while the ADC is reading.

Table 1: Chosen component values based on simulations. N/A means not applicable.

Component	W	L	C
M1	1.08μm	1.08μm	N/A
M2	1.08μm	1.08μm	N/A
M3	5.04μm	0.36μm	N/A
M4	1.08μm	1.08μm	N/A
MC1	1.08μm	1.08μm	N/A
MC2	1.08μm	1.08μm	N/A
C_S	N/A	N/A	2pF

In T_7, *NRE_1* is set `HIGH` again. From T_8 to T_{10}, the reading process repeats for *NRE_2* to read the bottom two pixels. In the last clock cycle of the state, T_{11}, *NRE_2* is set `HIGH` again to end the readout. It can be noted that the *readout* state has a fixed duration of 9 clock cycles, as the only state with a fixed duration.

4 Results

4.1 Analog pixel circuit

4.1.1 Values and dimensions

The transistor technology used limits the width to

$$1.08\mu\text{m} \leq W \leq 5.04\mu\text{m} \tag{3}$$

and limits the length to

$$0.36\mu\text{m} \leq L \leq 1.08\mu\text{m}. \tag{4}$$

As explained in section 3.1.2, W/L should be as small as possible to reduce leakage current. Therefore the length and width of the switch transistors will be 1.08μm.

CS was tuned to 2pF. The output from simulations with this value for the exposure-light corners min-min, min-max, max-min and max-max are shown respectively in figures 9, 10, 11 and 12 in appendix A. These simulations were done by transient analysis of the voltage over C_S while applying photodiode current and exposure time for the corner cases. The resulting voltage over C_S is the value at the end of the exposure time.

For M3 and the active load the dynamic range was largest when W/L for M3 was as large as possible and W/L for MC1 and MC2 as small as possible. The simulation used for this is as follows: Simulate a transient analysis on the netlist for the full analog circuit, set *NRE_1*

Table 2: R_{DS} for an NMOS switch in the fast, typical and slow corners.

Corner	Off	On
FF	5GΩ	500kΩ
TT	500GΩ	750kΩ
SS	50TΩ	1MΩ

`LOW` and *NRE_2* `HIGH`. *ERASE* should first be `HIGH` a little while, then when it gets `LOW`, *EXPOSE* goes `HIGH`, and it should be `HIGH` long enough for the voltage on the OUT wires to reach its maximum value. The dynamic range is then the difference between the maximum and minimum of this curve. The graph from this simulation with final value is shown in figure 13 in appendix A.

4.1.2 Analog circuit simulation

The simulation of the full analog circuit is shown in figure 17. In this simulation the exposure time was 5ms and the pixels 11, 12, 21 and 22 were exposed by photodiode currents of 750pA, 50pA, 300pA and 100pA respectively.

4.1.3 Process variation simulations

The simulation results for R_{DS} as a function of V_{GS} for a single NMOS are shown in figures 14, 15 and 16 for FF, TT and SS corners respectively. Table 2 shows what R_{DS} values this gives when switched fully on and off.

Results from full analog circuit simulation in FF and SS corners are shown in figures 18 and 19 respectively.

4.2 Digital circuit

The Verilog code for the digital camera controller is included in appendix 6, and the module interface is shown in figure 7.

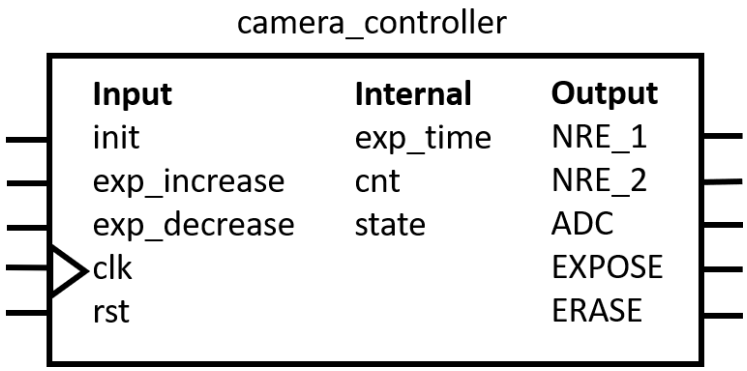

Figure 7: Overview of input and output pins as well as internal registers for the camera controller module.

exp_time is a 5-bit internal register that holds the exposure time in milliseconds. The exposure time is 16ms by default, which is the middle value between the minimum 2ms and the maximum 30ms. *exp_time* is only adjusted if *exp_increase*, *exp_decrease* or *rst* is `HIGH`. This means that after a picture is taken, the exposure time is not reset, allowing the user to take multiple pictures in the same lighting without having to tune the exposure time back after each picture. If the user presses the reset button in any of the three states, *exp_time* will return to its default value of 16.

cnt is, like *exp_time*, an internal, 5-bit register. When the camera state switches to *exposure* or *readout*, *cnt* is set to the duration of the state in clock cycles minus one. For every clock cycle in the state thereafter, *cnt* will decrement. After *cnt* has reached 0, the state will switch to the next one. The clock frequency is specified as 1kHz in this camera, meaning that one clock cycle lasts for 1ms. The state's duration in clock cycles thus equals its duration in milliseconds. This means that *cnt* will be set to $exp\_time - 1$ when the state switches to *exposure*, and 8 when the state switches to *readout*.

state is a 2-bit internal register that holds the state the camera will be in the next clock cycle, respresented by an integer. 0 represents the *idle* state, 1 respresents *exposure*, and 2 represents *readout*. It is important to note that whenever *state* is set to 0, *ERASE* is set to `HIGH` at the same time, i.e. at the rising edge of the same clock cycle. The camera will therefore not operate as being in the *idle* state and allowing a new picture to be taken before the next rising edge of the clock. This ensures that *ERASE* always is `HIGH` for at least one clock cycle after a picture is taken or cancelled underway, as described in section 3.2.1.

4.2.1 Digital circuit simulation

A showcase of the properties of the implemented camera controller is shown in figure 8. The first five signals in the list to the left are the input signals, the following five are the output signals, and the last three are the internal registers.

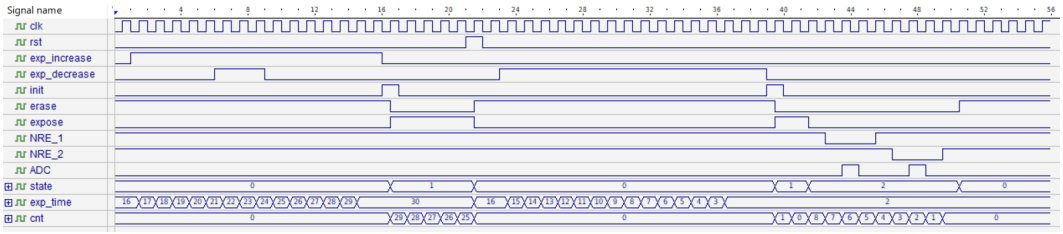

Figure 8: A waveform for the digital camera controller showcasing its properties.

Up until the 16th clock cycle in figure 8, the camera is in the *idle* state. All output signals are true to the *idle* state as it is described in section 3.2.1; *ERASE*, *NRE_1* and *NRE_2* are `HIGH`, and *EXPOSE* and *ADC* are `LOW`. From the second clock cycle to the 16th, *exp_increase* is `HIGH`. *exp_time* increments for each clock cycle in this interval, until it reaches 30 at the 15th cycle and stays constant. This shows that the exposure time will not exceed 30ms, as specified in section 3.2.1. It has also been made a point from clock cycle seven to nine that *exp_increase* overrides *exp_decrease* where they collide, a behaviour which also was specified as intended in section 3.2.1.

In the 17th clock cycle, *init* is `HIGH`, which sends the camera into the *exposure* state. The output signals act accordingly as *ERASE* goes `LOW` and *EXPOSE* goes `HIGH`. *cnt* is set to 29 and begins decrementing for each clock cycle. The *exposure* state is stopped short at cycle 22, where *rst* goes high for one cycle. As desired, this returns the camera to the *idle* state and sets *exp_time* back to its default, 16.

Then, from cycle 24 to 39, *exp_decrease* is set `LOW`. *exp_time* decrements as intended, and stays constant after it reaches the minimum of 2. In the 40th cycle, *init* is `HIGH` again, causing the camera to enter the *exposure* state. *cnt* is set to 1, and decrements for one cycle, after which it has reached 0. True to the FSM illustrated in figure 5, this prompts the camera to enter the *readout* state.

The *cnt* register is set to 8 as the camera switches to *readout*. It rightfully decrements for each clock cycle, and the output signals act exactly as specified for the *readout* state in figure 6. When the readout is finished, i.e. after *cnt* has reached 0, the camera enters the *idle* state again.

5 Discussion

5.1 Process variation

The values for R_{DS} for a switch transistor turned off is 5GΩ in FF, 500GΩ in TT and 50TΩ. For a voltage of 1V this means a leakage of 200pA in FF, and that is actually quite dramatic in our circuit. In SS however, a voltage of 1V means a leakage current of 20fA, and that is actually better for our circuit than the TT case.

The R_{DS} values when the transistors are switched on, however are not varying that much. The R_{DS} values for FF, TT and SS respectively are 500kΩ, 750kΩ and 1MΩ. To conduct a current of 1nA, the required voltages are respectively 500μV, 750μV and 1mV. All these values are very small compared to V_{DD}, so when switched on, our transistors can be considered ideal conductors for practical purposes.

For the full analog circuit, we can see as expected that the way in which the FF corner increases leakage current has a dramatic impact on the system as a whole, while the way SS increases the on resistance does not make a big change. And the cool thing is that SS also reduces leakage current compared to TT, so that SS in total improves behaviour of our design.

5.2 Modular digital design

In this project's digital circuit, only one module is used, which is illutrated in figure 7 and implemented in Verilog in appendix C. This choice may be controversial, as many prefer a more modular design. However, having several modules for this project seemed excessive and unnecessarily complex. This is a very simple, low-resolution, black-and-white camera, and the single-module implementation is not particularly challenging to read or understand. Modulisation was therefore opted out in this project. However, in a natural continuation of the project, like increasing the resolution or enabling colour, dividing the digital design into modules is very much relevant.

6 Conclusion

This project has proven through simulations that one can successfully implement a simple 2x2 digital camera controller in Verilog and an analog circuit for exposure and readout in AIM-Spice. Through user inputs, the camera controller will increase and decrease the exposure time to a minimum of 2ms and a maximum of 30ms, and initialize a photo. Each pixel successfully outputs voltages varying with the light conditions and exposure time, and the camera is able to read all the different pixel values individually to the ADCs. The camera suffers when process variations cause the switch transistors to be faster than typical - this will lead to a much darker photography than desired. It will however work exceptionally well if process variations cause slower switch transistors.

Appendices

A Analog simulation graphs

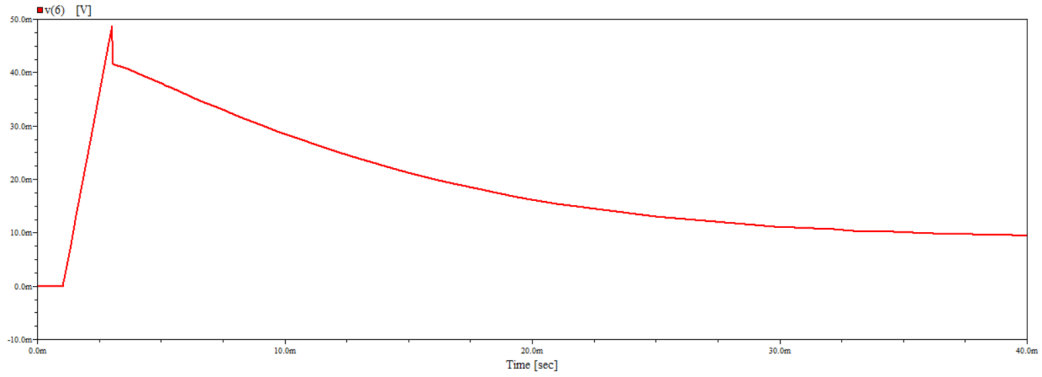

Figure 9: Minimum exposure time - minimum light

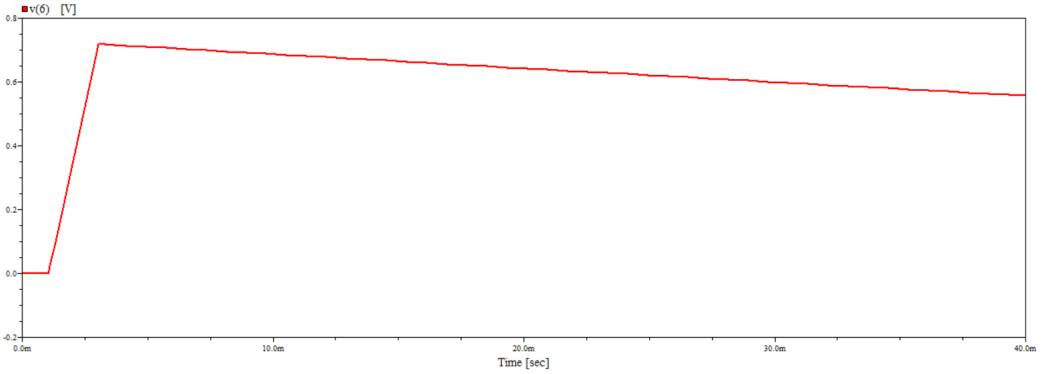

Figure 10: Minimum exposure time - maximum light

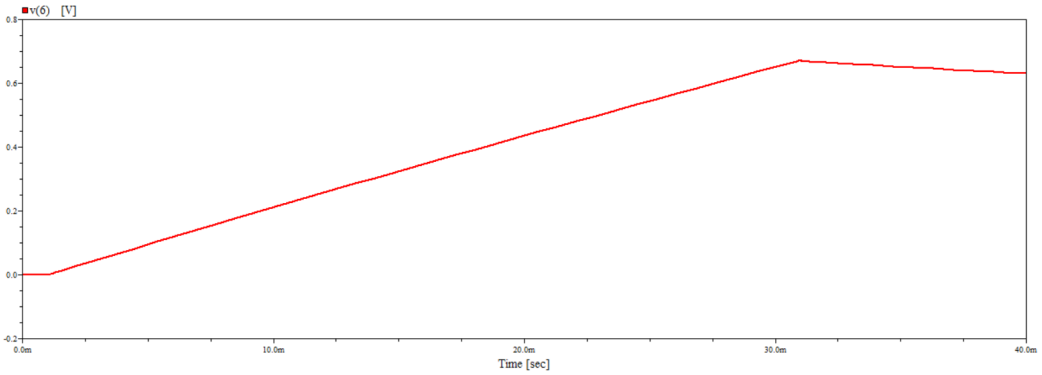

Figure 11: Maximum exposure time - minimum light

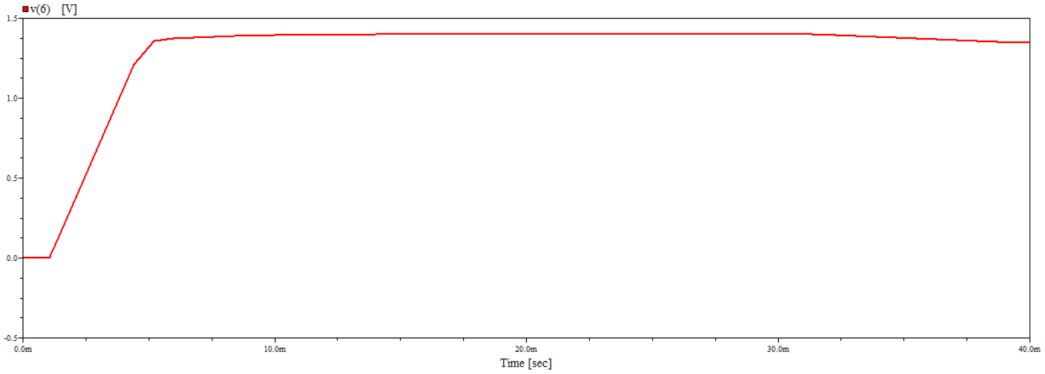

Figure 12: Maximum exposure time - maximum light

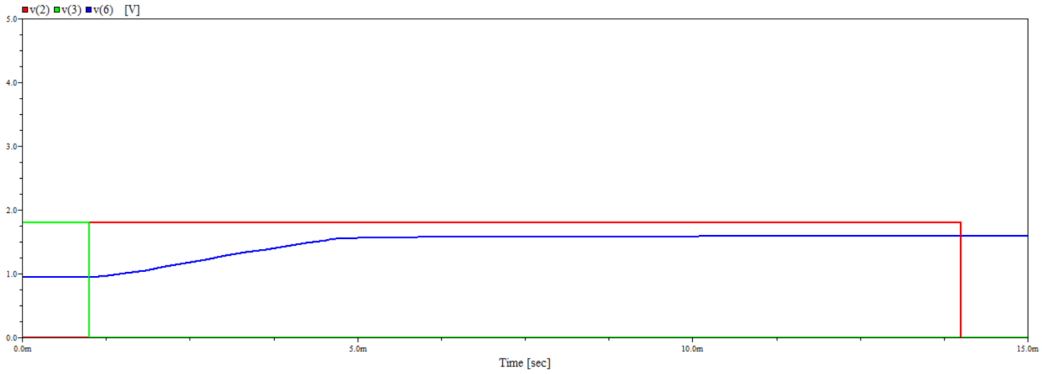

Figure 13: Simulation for maximizing dynamic range of M3 and active load.

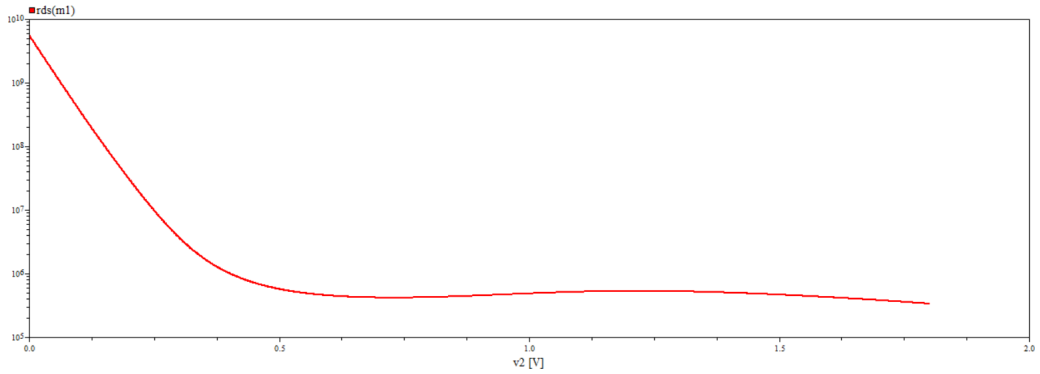

Figure 14: FF corner simulation of R_{DS} of NMOS switch as function of V_{GS}. $V_{DS} = 1.8$V.

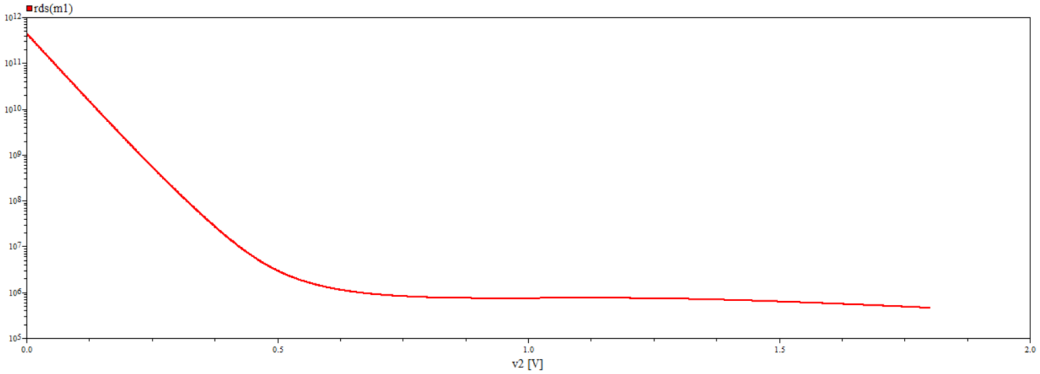

Figure 15: TT corner simulation of R_{DS} of NMOS switch as function of V_{GS}. $V_{DS} = 1.8$V.

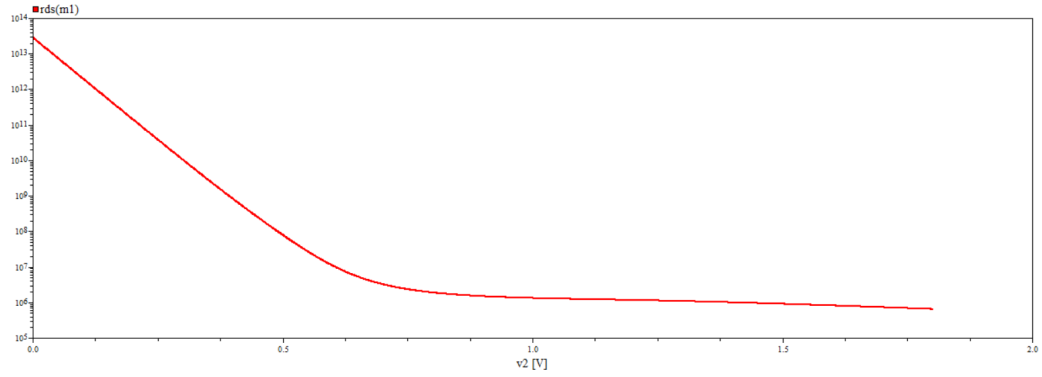

Figure 16: SS corner simulation of R_{DS} of NMOS switch as function of V_{GS}. $V_{DS} = 1.8$V.

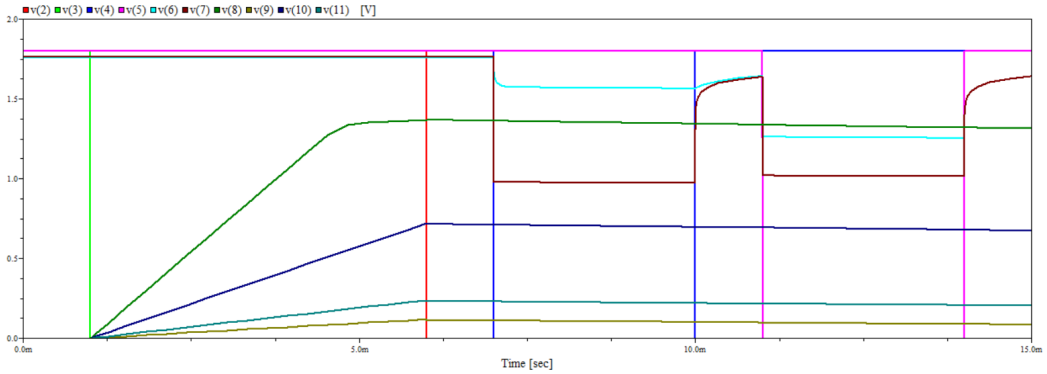

Figure 17: Simulation of analog circuit. The color(node nr)'s are: red(2): EXPOSE, light green(3): ERASE, blue(4): NRE_1, magenta(5): NRE_2, light blue(6): OUT1, brown(7): OUT2, dark green(8): V_{C11}, mustard(9): V_{C12}, dark blue(10): V_{C21}, teal(11): V_{C22}.

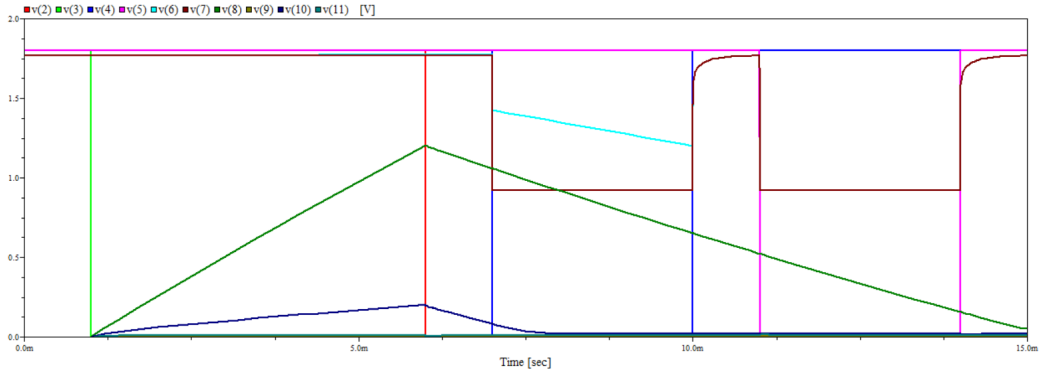

Figure 18: FF corner simulation of full analog circuit. Colors are the same as in figure 17.

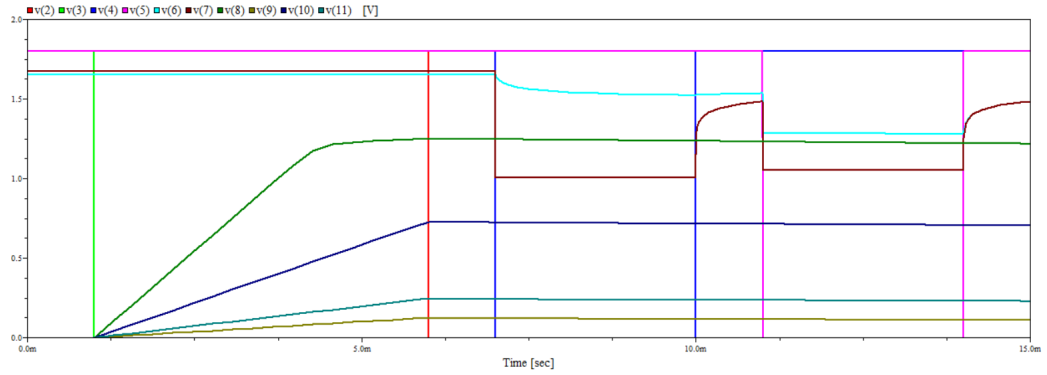

Figure 19: SS corner simulation of full analog circuit. Colors are the same as in figure 17.

B Spice code

Listing 1: SPICE code for full analog simulation.

```
[aimspice]
[description]
2050
* Pixel subcircuit

.include model_files/p18_cmos_models_ss.inc
.include model_files/PhotoDiode.cir

.param minW 1.08u
.param maxW 5.04u
.param maxL 1.08u
.param minL 0.36u

.param M1W minW
.param M1L maxL
.param M2W minW
.param M2L maxL
.param M3W maxW
.param M3L minL
.param M4W minW
.param M4L maxL

.param MCW minW
.param MCL maxL
.param CC_value 3p

.param CS_value 2p

.subckt pixel VDD VSS EXPOSE ERASE NRE OUT N2
.param I_D 150p
xPhotoDiode VDD 1 PhotoDiode Ipd_1 = I_D
* How to make NMOS: MX drain gate source bulk NMOS W= L=
M1 1 EXPOSE N2 VSS NMOS W=M1W L=M1L
M2 N2 ERASE VSS VSS NMOS W=M2W L=M2L

* How to make capacitor: CX node1 node2 value
CS N2 VSS CS_value

* How to make PMOS: MX source gate drain bulk PMOS W= L=
M3 3 N2 VSS VDD PMOS W=M3W L=M3L
M4 OUT NRE 3 VDD PMOS W=M4W L=M3L
.ends

.subckt load VDD VSS OUT
MC VDD OUT OUT VDD PMOS W=MCW L=MCL
```

```
CC OUT VSS CC_value
.ends

* How to make pixel:
* xName      VDD  VSS  EXPOSE  ERASE  NRE  OUT   N2    pixel  I_D=
  xPixel11    1    0      2       3     4    6    8     pixel  I_D=750p
  xPixel12    1    0      2       3     4    7    9     pixel  I_D=50p
  xPixel21    1    0      2       3     5    6   10     pixel  I_D=300p
  xPixel22    1    0      2       3     5    7   11     pixel  I_D=100p

xLoad1 1 0 6 load
xLoad2 1 0 7 load

VDD 1 0 dc 1.8v
* How to make pulse:
* V1 - initial voltage; V2 - peak voltage; TD - initial delay time;
* Tr - rise time; Tf - fall time; pwf - pulse-wise; and Period - period.
* Vname   N1   N2   pulse(   V1     V2     TD    Tr    Tf     PW   Period)
  vERASE   3    0   pulse(   0v    1.8V    0m    1u    1u     1m     15m)
  vEXPOSE  2    0   pulse(   0v    1.8v    1m    1u    1u     5m     15m)
  vNRE1    4    0   pulse( 1.8v     0v    7m    1u    1u     3m     15m)
  vNRE2    5    0   pulse( 1.8v     0v   11m    1u    1u     3m     15m)

* vNRE1 4 0 dc 1.8v
* vNRE2 5 0 dc 0v

* How to make transient analysis: .TRAN TSTEP TSTOP
.plot TRAN V(2) V(3) V(4) V(5) V(6) V(7) V(8) V(9) V(10) V(11)


[tran]
100n
15m
X
X
0
[ana]
4 0
[end]
```

Listing 2: SPICE code for CS value simulation.

```
[aimspice]
[description]
2030
* Simulation to tune CS

.include model_files/p18_cmos_models_tt.inc
.include model_files/PhotoDiode.cir
```

```
.param minW 1.08u
.param maxW 5.04u
.param maxL 1.08u
.param minL 0.36u

.param M1W minW
.param M1L maxL
.param M2W minW
.param M2L maxL
.param M3W maxW
.param M3L minL
.param M4W minW
.param M4L maxL

.param MCW minW
.param MCL maxL
.param CC_value 3p

.param CS_value 2p

.subckt pixel VDD VSS EXPOSE ERASE NRE OUT N2
.param I_D 150p
xPhotoDiode VDD 1 PhotoDiode Ipd_1 = I_D
* How to make NMOS: MX drain gate source bulk NMOS W= L=
M1 1 EXPOSE N2 VSS NMOS W=M1W L=M1L
M2 N2 ERASE VSS VSS NMOS W=M2W L=M2L

* How to make capacitor: CX node1 node2 value
CS N2 VSS CS_value

* How to make PMOS: MX source gate drain bulk PMOS W= L=
M3 3 N2 VSS VDD PMOS W=M3W L=M3L
M4 OUT NRE 3 VDD PMOS W=M4W L=M3L
.ends

.subckt load VDD VSS OUT
MC VDD OUT OUT VDD PMOS W=MCW L=MCL
CC OUT VSS CC_value
.ends

* How to make pixel:
* xName      VDD  VSS  EXPOSE  ERASE  NRE  OUT   N2    pixel  I_D=
  xPixel11    1    0       2      3     4    6    8     pixel  I_D=750p
  xPixel12    1    0       2      3     4    7    9     pixel  I_D=50p
  xPixel21    1    0       2      3     5    6   10     pixel  I_D=300p
  xPixel22    1    0       2      3     5    7   11     pixel  I_D=100p
```

```
xLoad1 1 0 6 load
xLoad2 1 0 7 load

VDD 1 0 dc 1.8v
* How to make pulse:
* V1 - initial voltage; V2 - peak voltage; TD - initial delay time;
* Tr - rise time; Tf - fall time; pwf - pulse-wise; and Period - period.
* Vname   N1   N2   pulse(   V1    V2    TD   Tr   Tf    PW  Period)
  vERASE   3    0   pulse(   0v  1.8V    0m   1u   1u    1m     40m)
  vEXPOSE  2    0   pulse(   0v  1.8v    1m   1u   1u   30m     40m)
* vNRE1    4    0   pulse( 1.8v   0v    7m   1u   1u    3m     15m)
* vNRE2    5    0   pulse( 1.8v   0v   11m   1u   1u    3m     15m)

  vNRE1    4    0   dc     1.8v
  vNRE2    5    0   dc     1.8v

* How to make transient analysis: .TRAN TSTEP TSTOP
.plot TRAN V(8)
[tran]
100n
40m
X
X
0
[ana]
4 0
[end]
```

Listing 3: SPICE code for M3 and active load values simulation.

```
[aimspice]
[description]
2050
* M3 and active load simulation

.include model_files/p18_cmos_models_tt.inc
.include model_files/PhotoDiode.cir

.param minW 1.08u
.param maxW 5.04u
.param maxL 1.08u
.param minL 0.36u

.param M1W minW
.param M1L maxL
.param M2W minW
.param M2L maxL
.param M3W maxW
.param M3L minL
```

```
.param M4W minW
.param M4L maxL

.param MCW minW
.param MCL maxL
.param CC_value 3p

.param CS_value 2p

.subckt pixel VDD VSS EXPOSE ERASE NRE OUT N2
.param I_D 150p
xPhotoDiode VDD 1 PhotoDiode Ipd_1 = I_D
* How to make NMOS: MX drain gate source bulk NMOS W= L=
M1 1 EXPOSE N2 VSS NMOS W=M1W L=M1L
M2 N2 ERASE VSS VSS NMOS W=M2W L=M2L

* How to make capacitor: CX node1 node2 value
CS N2 VSS CS_value

* How to make PMOS: MX source gate drain bulk PMOS W= L=
M3 3 N2 VSS VDD PMOS W=M3W L=M3L
M4 OUT NRE 3 VDD PMOS W=M4W L=M3L
.ends

.subckt load VDD VSS OUT
MC VDD OUT OUT VDD PMOS W=MCW L=MCL
CC OUT VSS CC_value
.ends

* How to make pixel:
* xName      VDD  VSS  EXPOSE  ERASE  NRE  OUT   N2    pixel  I_D=
  xPixel11    1    0       2      3     4    6    8     pixel  I_D=750p
  xPixel12    1    0       2      3     4    7    9     pixel  I_D=50p
  xPixel21    1    0       2      3     5    6   10     pixel  I_D=300p
  xPixel22    1    0       2      3     5    7   11     pixel  I_D=100p

xLoad1 1 0 6 load
xLoad2 1 0 7 load

VDD 1 0 dc 1.8v
* How to make pulse:
* V1 - initial voltage; V2 - peak voltage; TD - initial delay time;
* Tr - rise time; Tf - fall time; pwf - pulse-wise; and Period - period.
* Vname   N1   N2   pulse(   V1     V2    TD   Tr   Tf    PW   Period)
  vERASE   3    0   pulse(   0v   1.8V    0m   1u   1u    1m     15m)
  vEXPOSE  2    0   pulse(   0v   1.8v    1m   1u   1u   13m     15m)
* vNRE1    4    0   pulse( 1.8v    0v    7m   1u   1u    3m     15m)
* vNRE2    5    0   pulse( 1.8v    0v   11m   1u   1u    3m     15m)
```

```
  vNRE1     4     0    dc         0v
  vNRE2     5     0    dc       1.8v

* How to make transient analysis: .TRAN TSTEP TSTOP
.plot TRAN V(2) V(3) V(6)

[tran]
100n
15m
X
X
0
[ana]
4 0
[end]
```

Listing 4: SPICE code for process variations simulation.

```
[aimspice]
[description]
352
* Corner simulations for camera

.include model_files/p18_cmos_models_ss.inc

.param minW 1.08u
.param maxW 5.04u
.param maxL 1.08u
.param minL 0.36u

* How to make NMOS:
* MX drain gate source bulk NMOS W=      L=
  M1      1     2       0     0 NMOS W=minW L=maxL

* VX + - DC(value)
  V1 1 0 DC( 1.8v)
  V2 2 0 DC(    0v)

.plot rds(M1)

[dc]
1
V2
0
1.8
0.001
[ana]
1 0
```

[end]

C Verilog code

Listing 5: Verilog code for camera controller.

```
//------------------------------------------------------------------------------
//
// Title         : camera_controller
// Design        : camera_controller
// Author        : sigurdht
// Company       : NTNU
//
//------------------------------------------------------------------------------
//
// File          : C:\Users\Sigurd\OneDrive - NTNU\elsys\semester5\ic\ic_project\verilo
// Generated     : Tue Nov  3 15:17:41 2020
// From          : interface description file
// By            : Itf2Vhdl ver. 1.22
//
//------------------------------------------------------------------------------
//
// Description :
//
//------------------------------------------------------------------------------
`timescale 1 ns / 1 ps

//{{ Section below this comment is automatically maintained
//   and may be overwritten
//{module {camera_controller}}
module camera_controller ( init ,exp_increase ,exp_decrease ,
    clk ,rst ,NRE_1 ,NRE_2 ,ADC ,expose ,erase );

//}} End of automatically maintained section

input wire init ;
input wire exp_increase ;
input wire exp_decrease ;
input wire clk ;
input wire rst ;
output reg NRE_1 ;
output reg NRE_2 ;
output reg ADC ;
output reg expose ;
output reg erase ;

reg [4:0] exp_time;
reg [4:0] cnt;
reg [1:0] state;

// -- Enter your statements here -- //
```

```
initial begin
    NRE_1 = 1;
    NRE_2 = 1;
    ADC = 0;
    expose = 0;
    erase = 1;

    exp_time = 5'd16;
    cnt = 5'd0;
    state = 2'd0;
end

always @(posedge clk) begin
    if (rst) begin
        // Reset: Go to idle state
        NRE_1 = 1;
        NRE_2 = 1;
        ADC = 0;
        expose = 0;
        erase = 1;

        exp_time = 5'd16;
        cnt = 5'd0;
        state = 2'd0;
    end else begin
        case (state)
            2'd0: begin
                // State is idle
                if (init) begin
                    // Go to exposure state
                    erase = 0;
                    expose = 1;
                    cnt = exp_time - 1;
                    state = 2'd1;
                end else begin
                    if (exp_increase) begin
                        if (exp_time < 30) exp_time++;
                    end else if (exp_decrease) begin
                        if (exp_time > 2) exp_time--;
                    end
                end
            end
            2'd1: begin
                // State is exposure
                if (cnt > 0) begin
                    cnt--;
                end else begin
```

```
                    // Go to readout state
                    expose = 0;
                    cnt = 5'd8;
                    state = 2'd2;
                end
            end
            2'd2: begin
                // State is readout
                if (cnt > 0) begin
                    case (cnt)
                        5'd8: NRE_1 = 0;
                        5'd7: ADC = 1;
                        5'd6: ADC = 0;
                        5'd5: NRE_1 = 1;
                        5'd4: NRE_2 = 0;
                        5'd3: ADC = 1;
                        5'd2: ADC = 0;
                        5'd1: NRE_2 = 1;
                    endcase
                    cnt--;
                end else begin
                    // Go to idle state
                    erase = 1;
                    state = 2'd0;
                end
            end
        endcase
    end
end

endmodule
```

Listing 6: Verilog code for camera controller testbench.

```
//------------------------------------------------------------------------------
//
// Title        : testbench
// Design       : camera_controller
// Author       : sigurdht
// Company      : NTNU
//
//------------------------------------------------------------------------------
//
// File         : C:\Users\Sigurd\OneDrive - NTNU\elsys\semester5\ic\ic_project\verilo
// Generated    : Tue Nov  3 15:07:12 2020
// From         : interface description file
// By           : Itf2Vhdl ver. 1.22
//
//------------------------------------------------------------------------------
```

```
//
// Description :
//
//----------------------------------------------------------------------------------
`timescale 1 ms / 1 us

//{{ Section below this comment is automatically maintained
//   and may be overwritten
//{module {testbench}}
module testbench ();

//}} End of automatically maintained section

// -- Enter your statements here -- //

reg init;
reg exp_increase;
reg exp_decrease;
reg clk;
reg rst;

wire NRE_1;
wire NRE_2;
wire ADC;
wire expose;
wire erase;

camera_controller camera_controller1(init, exp_increase, exp_decrease,
    clk, rst, NRE_1, NRE_2, ADC, expose, erase);

initial begin
    clk = 0;
    rst = 0;
    exp_increase = 0;
    exp_decrease = 0;
    init = 0;
    #1
    exp_increase = 1;
    #5
    exp_decrease = 1;
    #3
    exp_decrease = 0;
    #7
    exp_increase = 0;
    init = 1;
    #1
    init = 0;
    #4
```

```
        rst = 1;
        #1
        rst = 0;
        #1
        exp_decrease = 1;
        #16
        exp_decrease = 0;
        init = 1;
        #1
        init = 0;
        #16
        $finish;
    end

    always begin
        #0.5
        clk = ~clk;
    end

endmodule
```