NTNU
Kunnskap for en bedre verden

Department of Electronic Systems

TFE4152 Design of Integrated Circuits

Project Report

Authors:
Brage Gaasø Samsonsen and Niklas Strømsnes

18th February 2021

Abstract

The purpose of this project is to design and implement the analog part of a digital camera controlled by a digital control logic. This is done by calculating the parameters of the needed components, such as transistor dimensions and the capacitance of the capacitors. The analog circuit will be implemented in the circuit simulation program AIM-Spice. The digital control logic will be implemented using Verilog in Active HDL. The purpose of the analog circuit is to capture light and send out an output signal based on the lights strength. The digital control logic's purpose is to send the needed control signals to the analog circuit so that it can capture the light and create the desired output signal. We end up with a fully functional analog circuit with the correct digital control signals sent from the digital control logic.

Contents

List of Figures

List of Tables

1 Introduction

We are going to design a digital camera using a specific circuit-topology. The digital camera will be composed of an analog and a digital part. To do this, we are going to use a circuit modeling program where we can design our analog circuit. We are also going to use a Hardware Description Language (HDL) to design and simulate the digital part of the camera circuit. The whole system will be composed as shown in Figure 1. The system will first receive an analog signal. This signal will be picked up by the analog sensor. It will then be converted to a digital signal using an analog to digital converter (ADC). All the sub-systems are being controlled by a Digital Control Logic (DCL). We will not be designing the ADC or anything coming after it.

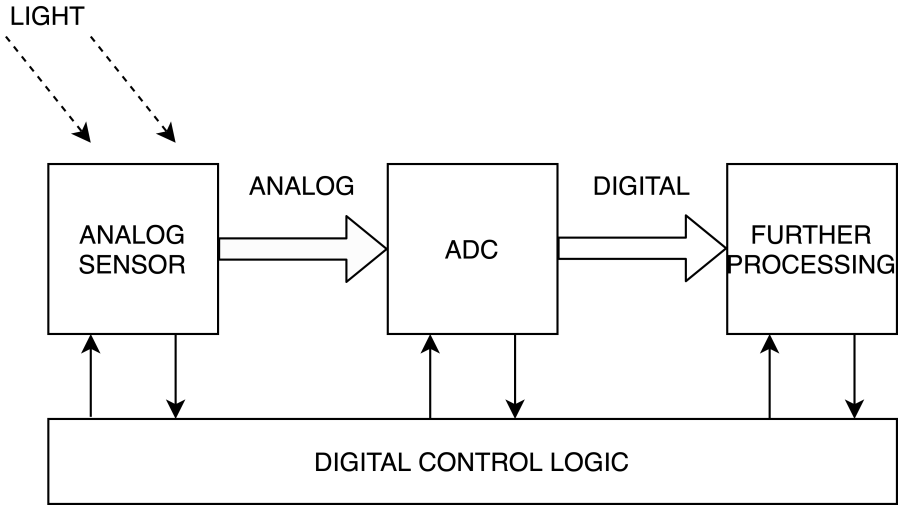

Figure 1: Overview of the system.

2 Theory

The whole camera is composed of an analog and a digital part. We will start off by designing the analog circuitry and then move on to the digital circuitry and logic. We also have to determine the dimensions of all the transistors and the values of the capacitors capacitance to make the analog circuit work as intended.

2.1 Analog Circuit

We have to start designing a circuit to detect light and turn it into an analog voltage. We will therefore design the pixel circuit shown in Figure 2.

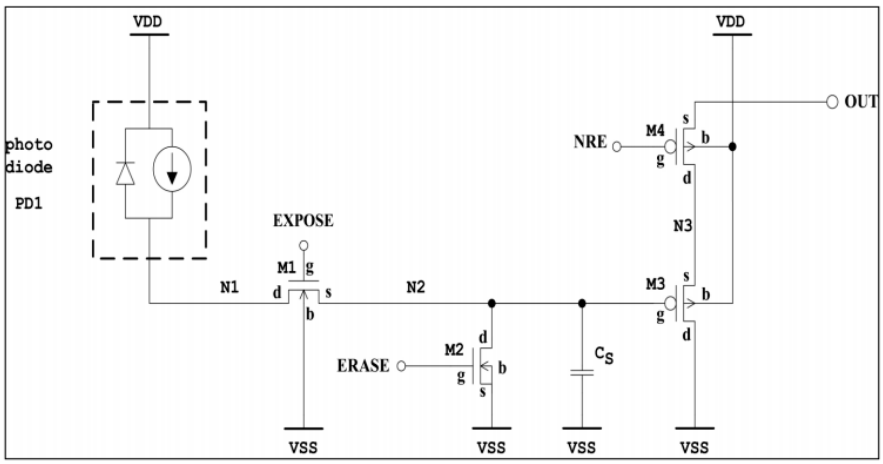

Figure 2: Overview of the pixel circuit [Larsen 2020].

On the left side of the circuit, we have the photo diode $PD1$. The strength of the light hitting the diode determines how strong the current will be. When the signal $EXPOSE$ is high, the transistor $M1$ is active and the current from the photo diode will charge the capacitor C_S. To get an output signal at OUT, the transistor $M4$ has to be active. Since this transistor is active low, it will be active when the read signal NRE is low and deactivates when NRE is high. As long as C_S is charged, the transistor $M3$ will not activate and the output signal will be affected by the gate voltage and not be grounded to VSS. $M3$ is also an active low transistor, hence it will not be active when C_S is charged. If $M2$ activates, the capacitor will discharge, since both sides of the capacitor will be grounded to VSS. This will happen when the signal $ERASE$ is high. $M1$, $M2$ and $M4$ will therefore act as switches. $M3$ will act as an active load.

The pixel circuit is a subcircuit of a larger circuit, shown in Figure 3. This is the analog camera circuit and it is composed of several subcircuits.

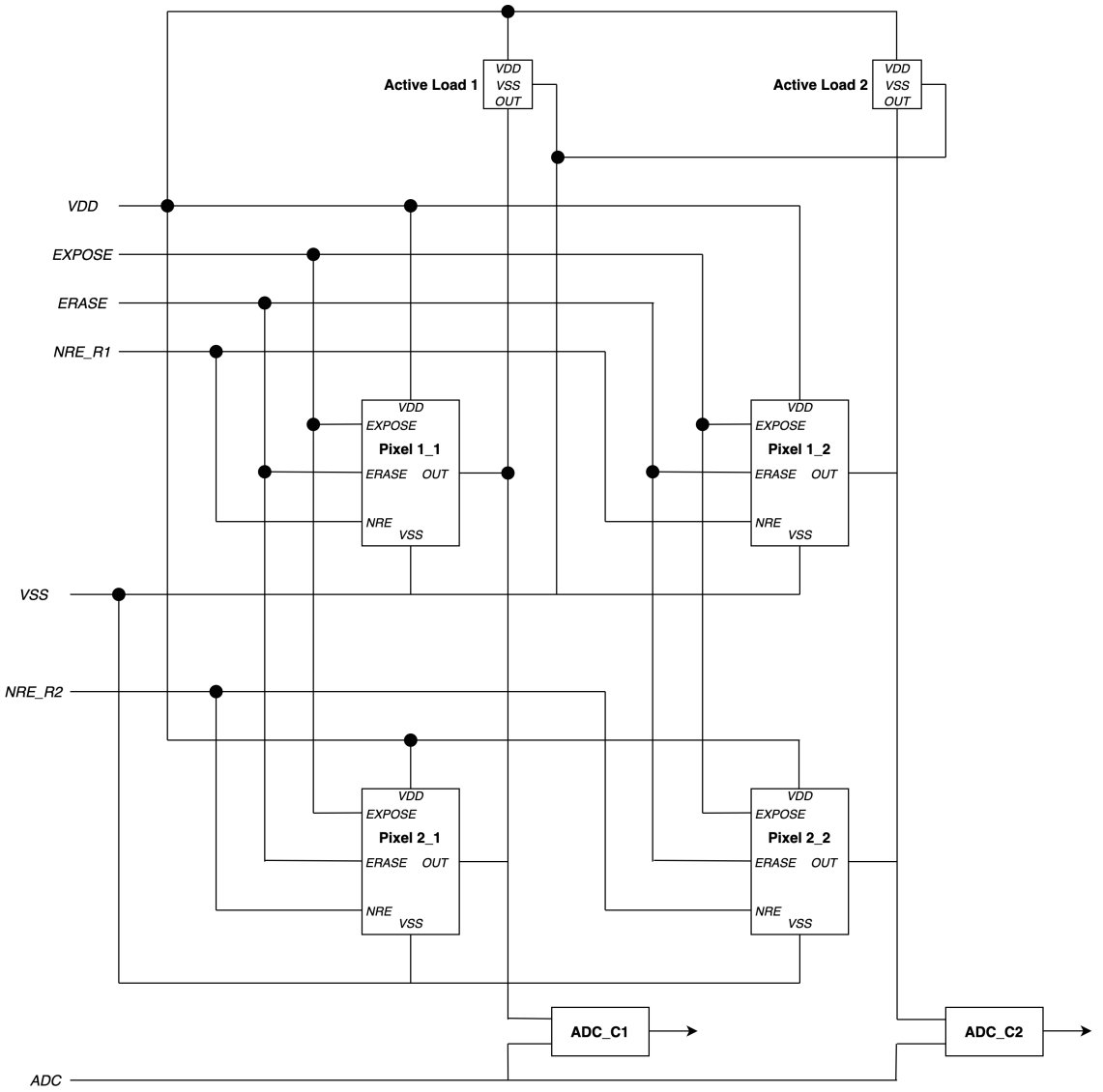

Figure 3: Overview of the 2x2 analog camera circuit.

The pixel circuit shown in Figure 2 is represented by the four boxes in Figure 3, **Pixel 1_1**, **Pixel 1_2**, **Pixel 2_1** an **Pixel 2_2**. These four boxes are individual pixel circuit. The analog camera circuit is composed of two columns, each of them composed of two pixel circuits, an analog to digital converter and an active load. The analog to digital converters are represented by the bottom two boxes, **ADC_C1** and **ADC_C2**. The active loads are represented by the top two boxes, **Active load 1** and **Active load 2**. Column 1 is composed of **Pixel 1_1**, **Pixel 1_2**, **ADC_C1** and **Active load 1**. Column 2 is composed of **Pixel 2_1**, **Pixel 2_2**, **ADC_C2** and **Active load 2**. The inputs of the camera circuit are similar to the inputs of the pixel circuit, being $EXPOSE$, $ERASE$, VDD, VSS, as well as the read signals for each row NRE_R1 and NRE_R2. These input signals are connected to their pixel circuit equivalents. NRE_R1 is connected to **Pixel 1_1** and **Pixel 1_2** NRE and NRE_R2 is connected to **Pixel 2_1** and **Pixel 2_2** NRE. The reason why we have divided our main analog circuit into two columns is so that we can get two OUT signals from two read signals instead of four. The OUT signals from the pixel circuits are then sent to the analog to digital converters to be further processed digitally. The OUT signals are also connected to an active load, shown as the two smaller boxes in the top of the figure. These active loads are better shown and described in Figure 4.

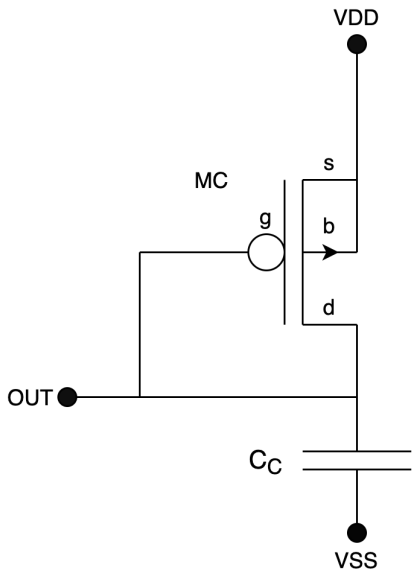

Figure 4: Overview the active load.

The active loads are composed of the PMOS transistor MC and the capacitor C_C. These circuits have three inputs, being VDD, VSS and the OUT signal from the pixel circuits. By including an active load at each output, we stabilize the OUT signals, and if the current through $M4$ is high when NRE is active, we get the functionality of a buffer circuit between $M3$, $M4$, MC and C_C. This is wanted, as we want a linear relationship between the current controlled by the photo diode and the voltage at OUT.

2.1.1 Transistor dimensions

In total, there are 18 different transistors in our analog circuit, however many of them have the same functions and can have similar dimensions. This means that all the $M1$ transistors in the different pixel circuits can share the same dimensions. Having this in mind, we only have to determine the dimensions of five different transistors, being $M1$, $M2$, $M3$ and $M4$ from the pixel circuits, and MC from the active loads.

All the transistor dimensions have to be between the predetermined limits

$$L_{min} < L < L_{max} \tag{1}$$

and

$$W_{min} < W < W_{max} \tag{2}$$

When deciding the parameters, we also have to keep in mind that the body effect might affect the threshold voltages of our transistors. The body effect will change the threshold voltage of a given transistor depending on its source-bulk voltage V_{SB}, surface potential ϕ_F, the body effect constant γ and the zero substrate bias voltage V_{T0}. The threshold voltage V_{TN} is then given by

$$V_{TN} = V_{TN0} + \gamma(\sqrt{|V_{SB} - 2\phi|} - \sqrt{|2\phi_F|}), \tag{3}$$

where the surface potential is given by

$$2\phi_F = 2\left(\frac{kT}{q}\right)\ln\left(\frac{N_A}{n_i}\right) \tag{4}$$

By looking at the surface potential, we can conclude that it will remain constant as long as we don't change the temperature T or the doping concentration N_A. The intrinsic carrier concentration n_i, the elementary charge q, and Boltzmann constant k are all constants. For an NMOS transistor that has its bulk connected to ground and not in the same node as its source, an increase in source-bulk voltage will decrease the threshold voltage. For a PMOS transistor however, that has its bulk connected to VDD, an increase in source-bulk voltage will increase the threshold voltage. Note that because of this, $M2$ a threshold voltage equal to V_{TN0}, and both $M3$ and $M4$ will have a higher threshold voltage. It is possible to neglect the body effect if the doping concentration is low enough, resulting in a threshold voltage almost equal to the zero substrate bias voltage.

The body effect can be neglected if the factor γ, given by

$$\gamma = \frac{\sqrt{2qN_aK_s\epsilon_0}}{C_{ox}}, \tag{5}$$

is small. Here, K_s is the relative permittivity of the semiconductor substrate. The gate capacitance C_{ox} depends on the relative permittivity K_{ox} and the thickness t_{ox} of the oxide layer at the gate, in the relation

$$C_{ox} = \frac{K_{ox}\epsilon_0}{t_{ox}} \tag{6}$$

To determine the dimensions of the transistors, we have to know that the leakage current $I_{D(off)}$ is given by

$$I_{D(off)} \approx I_{D0}\left(\frac{W}{L}\right)e^{q(-V_S-V_{TN})/(nkT)}, \tag{7}$$

where n is the relation

$$n = \frac{C_{ox}+C_{j0}}{C_{ox}} \tag{8}$$

and the current I_{D0} is given by

$$I_{D0} = (n-1)\mu_nC_{ox}\left(\frac{kT}{q}\right)^2 \tag{9}$$

By knowing this, we can minimize the leakage current by adjusting the width and the length of the transistor. By selecting a larger value for the length than the width, we get a low leakage current. When determining the parameters for $M1$ and $M2$, we know that if the leakage in the transistors are too significant, the capacitor C_S will never be fully charged, which is not ideal since it will result in inaccurate readings. We will therefore have to make the width of $M1$ and $M2$ minimal, and the lengths maximal to minimize the leakage current.

To determine the dimensions of the three PMOS transistors $M3$, $M4$ and MC, we have to look at the drain current I_D in saturation given by

$$I_D = \frac{\mu_nC_{ox}}{2}\frac{W}{L}(V_{eff})^2, \tag{10}$$

where V_{eff} is given by difference between the gate-source voltage V_{GS} and V_{TN}

$$V_{eff} = V_{GS} - V_{TN} \tag{11}$$

To get the largest possible drain current, we will have to make the width of the transistor much larger than the length of the transistor. For $M3$ and MC we use that their effective voltages V_{eff} are given by

$$V_{eff,M3} = V_{OUT} - V_{N2} - V_{TN} \tag{12}$$

$$V_{eff,MC} = V_{DD} - V_{OUT} - V_{TN} \tag{13}$$

V_{OUT} is the output voltage, V_{N2} is the voltage in the node $N2$ which is also the gate voltage for $M3$ as well as being the same voltage as V_{CS}. V_{CS} is the voltage of the capacitor C_S. V_{DD} is the supply voltage of the entire circuit. We want the transistors to operate in the saturation region which gives us the following inequalities

$$V_{TN} \leq V_{OUT} - V_{N2} \tag{14}$$

$$V_{TN} \leq V_{DD} - V_{OUT} \tag{15}$$

Since the two transistors have the same drain current I_D, we can get the following equations by using (7), (12) and (13)

$$\frac{\mu_n C_{ox}}{2} \frac{W_{M3}}{L_{M3}} (V_{eff,M3})^2 = \frac{\mu_n C_{ox}}{2} \frac{W_{MC}}{L_{MC}} (V_{eff,MC})^2 \tag{16}$$

which we can simplify to

$$\frac{W_{M3}}{L_{M3}} (V_{eff,M3})^2 = \frac{W_{MC}}{L_{MC}} (V_{eff,MC})^2 \tag{17}$$

By squaring 17, we get

$$\sqrt{\frac{W_{M3}}{L_{M3}}} V_{eff,M3} = \sqrt{\frac{W_{MC}}{L_{MC}}} V_{eff,MC} \tag{18}$$

By inserting (12) and (13) into (18) we get

$$\sqrt{\frac{W_{M3}}{L_{M3}}} (V_{OUT} - V_{N2} - V_{TN}) = \sqrt{\frac{W_{MC}}{L_{MC}}} (V_{DD} - V_{OUT} - V_{TN}) \tag{19}$$

By solving (19) for the output voltage V_{OUT}, we get

$$V_{OUT} = \frac{\sqrt{\frac{W_{M3}}{L_{M3}}}}{\sqrt{\frac{W_{M3}}{L_{M3}}} + \sqrt{\frac{W_{MC}}{L_{MC}}}} V_{N2} + \frac{\sqrt{\frac{W_{MC}}{L_{MC}}}}{\sqrt{\frac{W_{M3}}{L_{M3}}} + \sqrt{\frac{W_{MC}}{L_{MC}}}} V_{DD} + \frac{\sqrt{\frac{W_{M3}}{L_{M3}}} - \sqrt{\frac{W_{MC}}{L_{MC}}}}{\sqrt{\frac{W_{M3}}{L_{M3}}} + \sqrt{\frac{W_{MC}}{L_{MC}}}} V_{TN} \tag{20}$$

To maximize the gain of the output voltage V_{OUT}, we have to make the fraction in front of V_{N2} and V_{TN} as large as possible and the fraction in front of V_{DD} as low as possible. This is because

we don't want to let V_{DD} interfere with the output voltage. This is possible by making the width of $M3$ as long as possible and the length as short as possible, as well as making the width of MC as short as possible and the length as long as possible. We want $M4$ to affect the buffer circuit as little as possible. We will therefore have to make the resistance in the transistor as low as we can by making the current through it as high as possible. This is achieved by maximizing the width and minimizing the length of $M4$.

2.1.2 Capacitor values

There are six capacitors in our main analog circuit: one in each pixel circuit and one in each active load. The capacitances in the active loads **Active load 1** and **Active load 2** are predetermined. We will only have to determine the four capacitors in the pixel circuits and they will all share the same capacitance. These four capacitors are represented as C_S, with their own individual pixel circuit. Their capacitances are decided to be within the constraint

$$C_S \leq C_{max} \tag{21}$$

We know that the voltage V_{CS} over the capacitors is given by

$$V_{CS} = \frac{1}{C_S} \int_0^t I(\tau) d\tau, \tag{22}$$

where t is the expose time for the capacitors to charge up, and $I(\tau)$ is the current charging the capacitors. $I(\tau)$ and t are given the restrictions

$$I_{min} \leq I(\tau) \leq I_{max} \tag{23}$$

and

$$t_{min} \leq t \leq t_{max} \tag{24}$$

Assuming that $I(\tau)$ is a direct current, we get

$$V_{CS} = \frac{It}{C_S} \tag{25}$$

from the integral, which then gives us the expression

$$C_S = \frac{It}{V_{CS}} \tag{26}$$

for the capacitance.

We can find the minimum value of the capacitors C_{min} by looking at the maximum value of the current I_{max}, a minimum time t_{min} and the maximum allowed voltage V_{CS}. C_{min} will therefore be expressed as

$$C_{min} = \frac{I_{max} t_{min}}{V_{CS,max}} \tag{27}$$

We will then have the restrictions

$$C_{min} \leq C_S \leq C_{max} \tag{28}$$

for C_S, where we can give the capacitors any given value within this range.

2.2 Digital part

To make the analog circuit work, we have to send in the control signals. In Figure 5 we can see the timing diagram including the different control signals in the circuit shown in Figure 2. The first signal shown in the diagram is Clk, which operates as the system clock. The signal $Init$ turns high when we want to start the process of taking a picture with our digital camera, and will only be high for a single clock period. $Erase$ starts high since we want C_S to be free from any charge when starting the exposure. After $Erase$ turns low when $Init$ turns low, $Expose$ turns high. This results in the charging of C_S. After the expose time has passed, the overflow flag $Ovf5$ turns high, which leads to NRE_R1 to turn low a clock period later. One clock period later, ADC goes from being low to high in a single clock period before going back to being low. NRE_R1 lasts for three clock periods, before turning back to high. When NRE_R1 turns high again, the overflow flag $Ovf4$ turns high for a single clock period, which turns NRE_R2 to low a clock period later. Equal to NRE_R1, NRE_R2 will last for three clock periods. A single clock period after NRE_R2 turns to low, ADC turns high again for one clock period. When NRE_R2 goes back to being high, $Ovf4$ turns high for a clock period, which sends the camera back to Idle. When NRE_R1 or NRE_R2 is low we'll get a analog signal at the output OUT shown in Figure 2. ADC activates a clock period after a read signal and is only active for a single clock period because the voltage V_{OUT} needs to be stable for us to convert it to a digital signal.

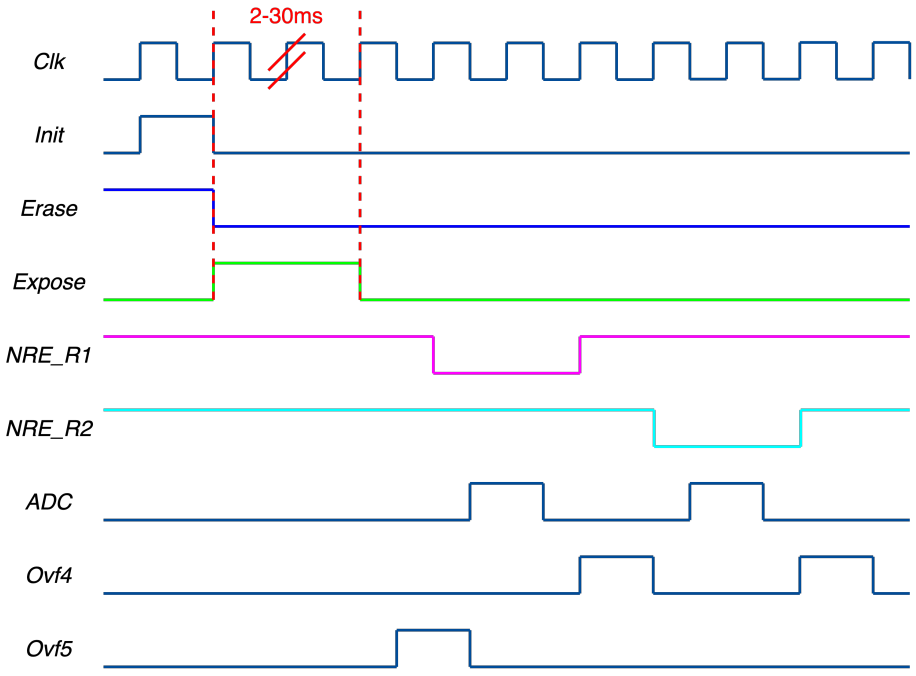

Figure 5: Overview of the timing diagram.

To create these digital control signals, we have to design a digital circuit. In Figure 6 we see a simplified version of the digital circuit's finite state machine (FSM). By implementing the system as an FSM, we can easily send out different signals based on different conditions.

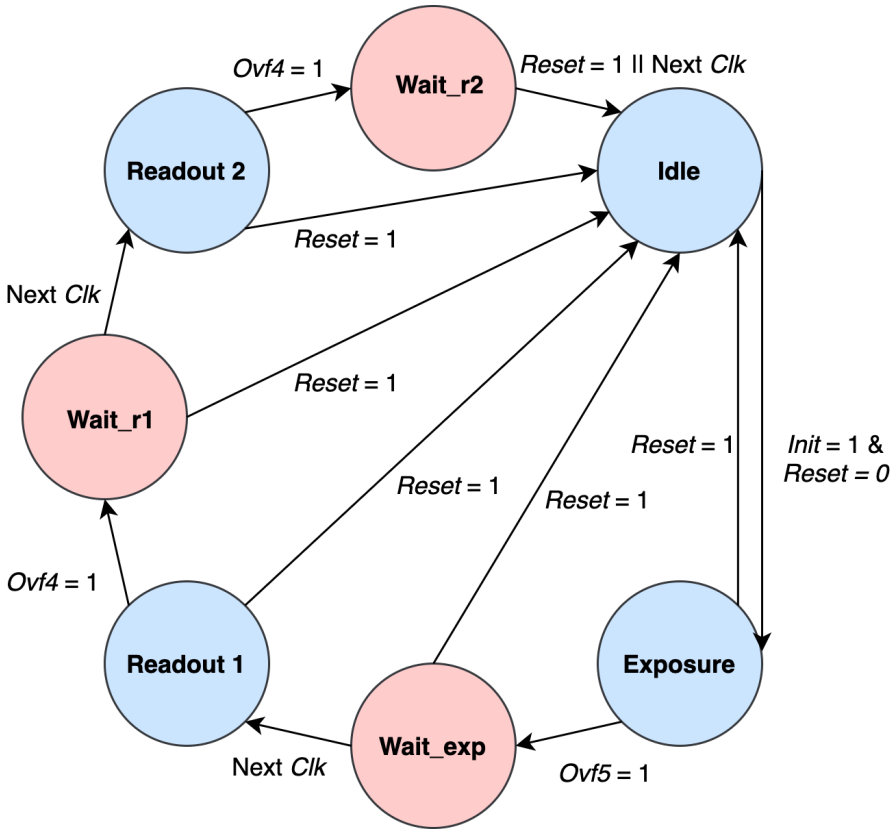

Figure 6: Overview of the FSM.

Initially, the FSM is in the state **Idle**. After the signal $Init$ has been sent into the machine, it goes into the state **Exposure**. If $Reset$ is high, it should stay in **Idle**, and also return to **Idle** if it is in any other state. After a certain amount of clock cycles determined by the signal Clk, the flag $Ovf5$ is set to high. By waiting a bit after the exposure has started, we ensure that the capacitor C_S is fully charged. When $Ovf5$ is set to high, the FSM enters a wait state **Wait_exp** that lasts for one clock period. After this clock period, we enter the state **Readout 1** and $Ovf5$ is set to low. Three clock cycles later, the overflow signal $Ovf4$ will be set to high and the FSM enters the state **Wait_r1**. This state lasts for one clock period and sends the FSM to **Readout 2** and sets $Ovf4$ to low. After three clock periods, $Ovf4$ is set to high again and the FSM enters the next state, **Wait_r2**. As all the other wait states, **Wait_r2** lasts for one clock period before going over to the next state **Idle** and setting $Ovf4$ to low. The system will now remain idle or run another exposure. A more technical overview of the digital circuitry is shown in Figure 7.

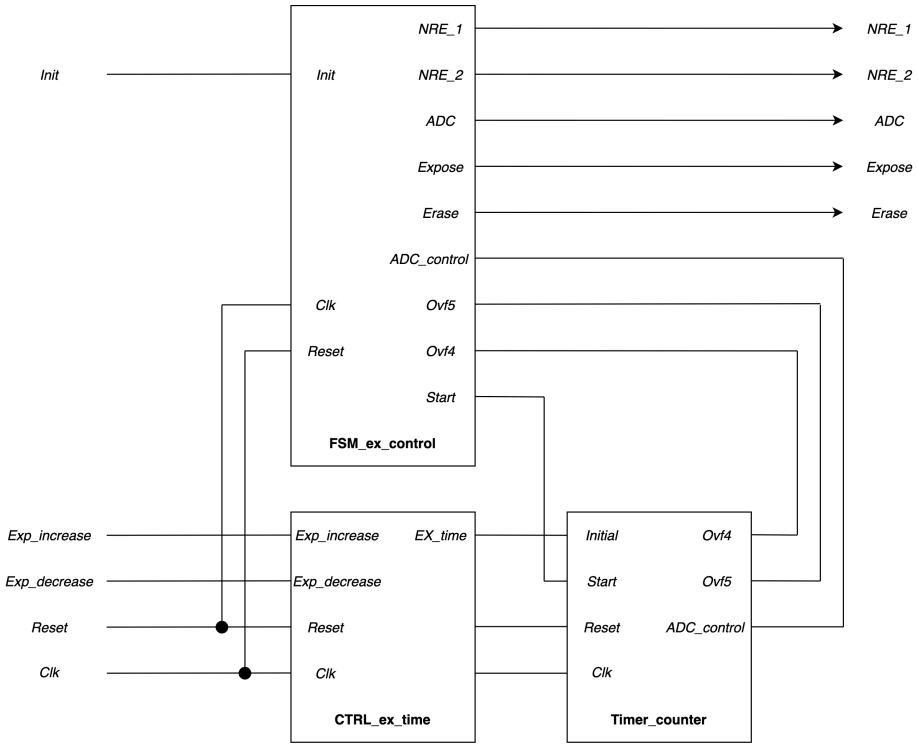

Figure 7: Overview of the block diagram for RE_Control

We see that the digital circuit is composed of three smaller subcircuits, **FSM_ex_control**, **CTRL_ex_time** and **Timer_counter**. In total, there are five input signals to the digital circuit **RE_control**, being $Init$, $Exp\_increase$, $Exp\_decrease$, Clk and $Reset$. These are all external signals. As explained earlier, $Init$ initiates the digital circuit, $Reset$ resets the circuit, Clk is a clock signal and $Exp\_increase$ and $Exp\_decrease$ increases or decreases the exposure time. The function of the **CTRL_ex_time** block is to adjust the exposure- and read-time, wich correlates to the $Expose$, NRE_1 and NRE_2 signals respectively. By adjusting $Exp\_increase$ and $Exp\_decrease$, we adjust the EX_time signal, which stands for exposure-time. The **Timer_counter** box controls when we send out the overflow signal $Ovf5$ and the control signal $ADC\_control$, by taking in EX_time, $Start$, Clk and $Reset$ as inputs. $Ovf5$ is then being sent as an input to the **FSM_ex_control** subcircuit, which will then send out the wanted control signals to the analog circuit, being NRE_R1, NRE_R2, ADC, $EXPOSE$ and $ERASE$. The $ADC\_control$ signal controls when ADC should be high and when it should be low. The inputs of the **FSM_ex_control** subcircuit are $Init$, Clk, $Reset$, $Ovf5$ and $ADC\_control$ signals. When the digital circuit is implemented with the analog circuit, we will have a complete circuit looking like the block diagram in Figure 8.

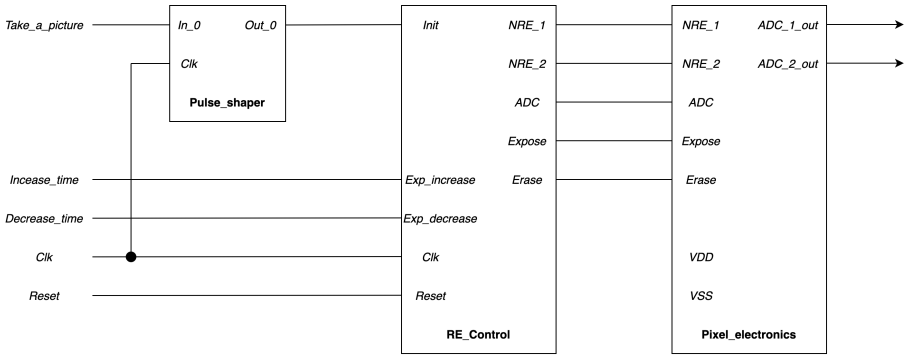

Figure 8: Overview of the block diagram for the entire camera circuit, composed of a **Pulse_shaper**, the **RE_Control** and the **Pixel_electronics**. We are not going to design the **Puls_shaper**.

Here we see the digital circuit's control signal outputs connected to the analog circuit, shown in Figure 3, represented as *RE_Control* and *Pixel_electronics*. The digital circuit shown in Figure 7 is represented as **RE_Control**.

3 Realization

To simulate the digital camera, we use the circuit simulation program AIM-Spice for the analog part and the HDL Verilog in the simulation program Active HDL. This section will be divided in three parts: the calculation of parameters we use, how the AIM-Spice code is implemented and finally how the Verilog code is implemented.

3.1 Parameters

3.1.1 Body effect parameters

In the model files, we have two different carrier concentrations N_A. We find that the carrier concentration $N_{A,NMOS}$ is $2.33 \cdot 10^{17}$ for the NMOS transistors and the carrier concentration $N_{A,PMOS}$ is $4.12 \cdot 10^{17}$ for the PMOS transistors. The thickness of the oxide layer t_{ox} is $4.1 \cdot 10^{-9}$cm for both the NMOS and PMOS transistors. The relative permittivity of silicon K_S is 11.8. The relative permittivity of SiO2 K_{OX} is 3.9Fm^{-2}. The vacuum permittivity ϵ_0 is $8.854 \cdot 10^{-12}\text{Fm}^{-1}$. The elementary charge q is $1.6 \cdot 10^{-19}$C. By using (4) we get the gate capacitance C_{OX} to be approximately 0.01Fm^{-1}. By using the value we found for C_{OX} together with $N_{A,NMOS}$, the elementary charge q, the relative permittivity of silicon K_S, the vacuum permittivity ϵ_0 and by using (5), we find that γ_{NMOS} is approximately $3.5 \cdot 10^{-4}$. By switching $N_{A,NMOS}$ with $N_{A,PMOS}$, we get that γ_{PMOS} is equal to approximately $4.6 \cdot 10^{-4}$. Since both γ_{NMOS} and γ_{PMOS} are extremely small, we can neglect the body effect.

3.1.2 Transistor parameters

The transistor parameters have to be between the following values

$$0.36\mu m < L < 1.08\mu m \tag{29}$$

and

$$1.08\mu m < W < 5.04\mu m \tag{30}$$

Since we want the leakage current to be minimal in the transistors $M1$ and $M2$, we set L_{M1} and L_{M2} to be maximal 1.07μm and the widths W_{M1} and W_{M2} to be 1.09μm. Since we want the transistors $M4$ to have minimal impact on the buffers, we minimize the length L_{M4} to 0.37μm and maximize the width W_{M4} to 5.03μm. To maximize the gain on the voltage in OUT, we will have to set the length L_{M3} to a minimum of 0.37μm and the length L_{MC} to a maximum of 1.07μm, as well as setting the width W_{M3} to a maximum of 5.03 and the width W_{MC} to a minimum of 1.09μm. These parameters are shown in table 1 in subsubsection 3.1.4.

3.1.3 Capacitor parameters

The capacitors C_S parameters have to be under the following value

$$C_S \leq 3pF \tag{31}$$

and the capacitors C_{C1} and C_{C2} in the active loads have the given value

$$C_{C1} = C_{C2} = 3pF \tag{32}$$

The current $I(\tau)$ is expected to be between the following values

$$50pA \leq I(\tau) \leq 750pA \tag{33}$$

and the exposure time has to be between

$$2ms \leq t \leq 30ms \tag{34}$$

The threshold voltage for the PMOS transistors is 0.39V and the supply voltage V_{DD} is 1.8V. By knowing this, we can use (12) and (13) to find that the maximum voltage V_{N2}, which is the same as V_{CS}, has to be lower than 1.02V. We will therefor set $V_{CS,max}$ to be 1V. To calculate the minimum value for C_S, we have to insert the maximum voltage $V_{CS,MAX}$, maximum current $I_{max}(\tau)$, which is 750pA, and the minimum time t_{min}, which is 2ms, into (26). We the get the following restriction to the capacitors C_S

$$1.5pF \leq C_S \leq 3pF \tag{35}$$

We will therefore choose the capacitors C_S to have a capacitance of 2.75F, since this is in the middle of the two values. These parameters are shown in table 1 in subsubsection 3.1.4.

3.1.4 Parameters summary

The entire overview of the parameters of the components that we will implement in our analog circuit is shown in the table 1.

Parameter	Parameter value
L_{M1}	1.07μm
L_{M2}	1.07μm
L_{M3}	0.37μm
L_{M4}	0.37μm
L_{MC}	1.07μm
W_{M1}	1.09μm
W_{M2}	1.09μm
W_{M3}	5.03μm
W_{M4}	5.03μm
W_{MC}	1.09μm
C_S	2.75pF
C_1	3pF
C_2	3pF

Table 1: Table of transistor and capacitor parameters.

3.2 Analog circuitry

A premade test bench was used in the testing, as shown in the code. We implement the pixel circuits and the active loads by making two sub circuits containing the needed components. We can then create the four pixel circuits and two active loads in the configuration shown in Figure 3. We can then simulate the entire circuit with the test bench. The results from the simulation are shown in Figure 9.

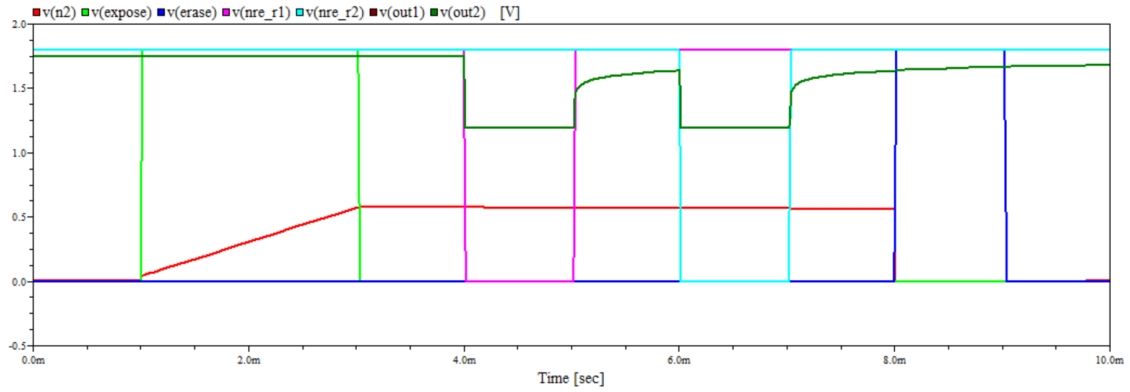

Figure 9: The voltages of the signals *Expose*, *Erase*, *NRE_R1*, *NRE_R2*, the here identical output signals, and the voltage over C_S.

We can see that the $EXPOSE$ signal enables the capacitors C_S to charge up and that the $ERASE$ signal enables the capacitors C_S to discharge. We can also see that the OUT signal depends on the capacitor C_S when the NRE_R1 or NRE_R2 signal is active. In figure Figure 10 we see a worst case scenario with parameter variations that result in a slower digital circuit. This does not change the ideal output signal that much as we see when comparing Figure 9 wit Figure 10.

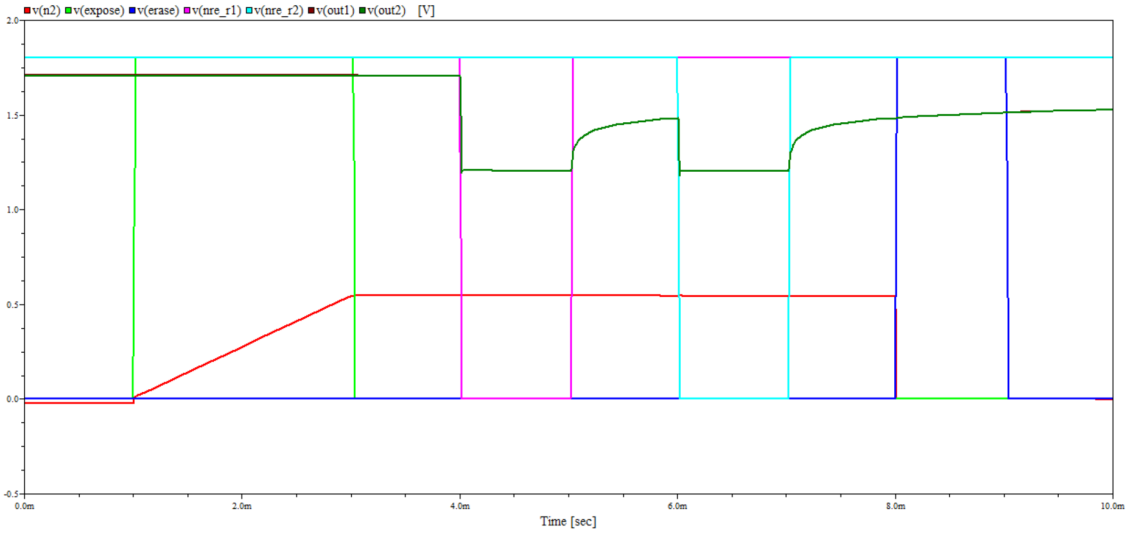

Figure 10: The voltages of the signals simulated with slow corners.

In figure Figure 10 we see a worst case scenario with parameter variations that result in a faster digital circuit. This changes the output signal by such a large amount that it would not operate as expected. We can clearly see this by comparing Figure 9 with Figure 11.

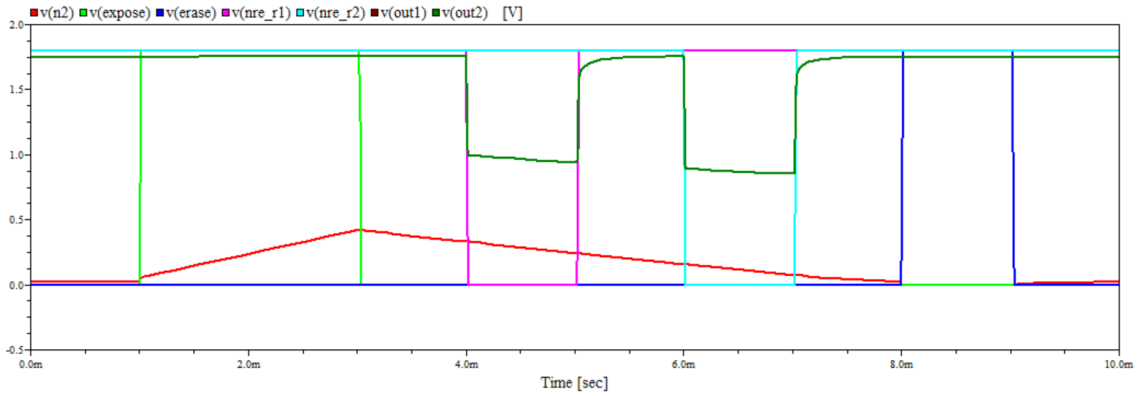

Figure 11: The voltages of the signals simulated with fast corners.

3.3 Digital circuitry

We implement the digital circuit by making each digital component shown in Figure 7. This is shown in the code as FSM_ex_control, CTRL_ex_time and Timer_counter in the Verilog code. The control signals are shown in Figure 12. The *Expose* signal is set to be 3ms in this instance. We can see that the signals look like in Figure 5.

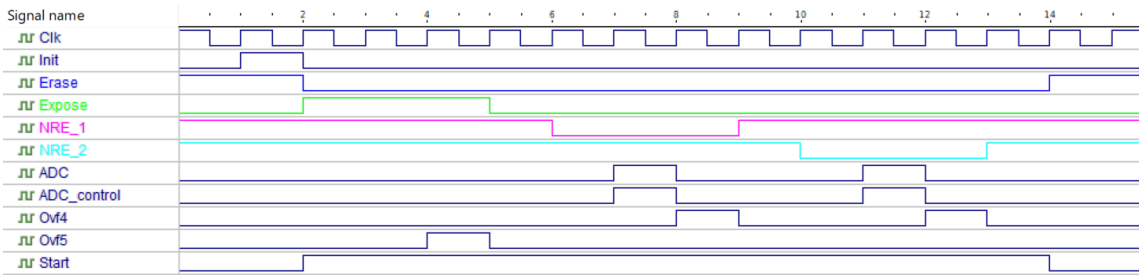

Figure 12: Overview of the timing simulated control signals.

The *Exp_increase* and *Exp_decrease* signals work as well. This is shown in Figure 13, where we start with an exposure time of 3ms and decrease it over two clock periods until it is at the lower limit of 2ms.

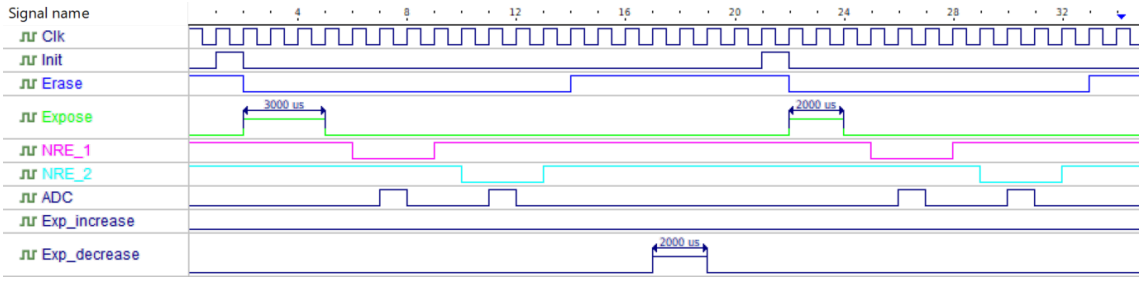

Figure 13: Overview of the timing diagram when decreasing time.

The reset signal works as shown in Figure 14. Here we see that the *reset* signal is set to high and that the digital control logic goes back to Idle. We then see that the digital control signals start again when we set the *Init* signal to high.

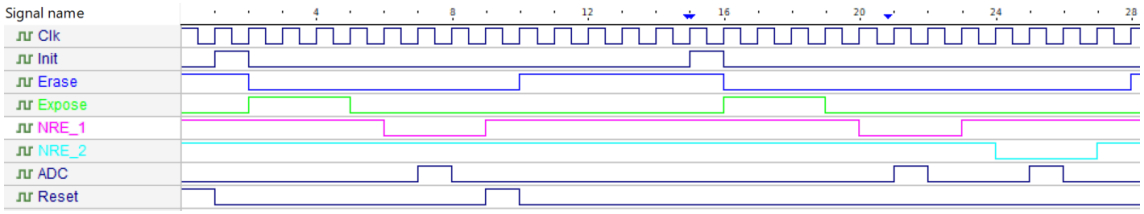

Figure 14: Overview of the timing diagram when resetting the signal.

4 Conclusion

We have designed both the analog part of the digital camera as well as the digital control logic. We implemented the analog circuitry in the simulator program AIM-spice. The digital control logic was implemented using Verilog in Active HDL. This resulted in a fully functional analog circuit with the correct digital control signals.

Bibliography

Carusone, Tony Chan, David A. Johns and Kenneth W. Martin (2012). *Analog Integrated Circuit Design*. WILEY.

Larsen, Bjørn B. (2020). *TFE4152 - Design of integrated circuits 2020, Project description, Digital camera*. URL: https://drive.google.com/file/d/1TBPxhWup2TgUId_N7HUfzBDP56rRkvY5/view?usp=sharing (visited on 16th Nov. 2020).

NTNU, Department of Marine Technology (2020). *IMT Software Wiki - LaTeX*. URL: https://www.ntnu.no/wiki/display/imtsoftware/LaTeX (visited on 15th Sept. 2020).

Appendix

A AIM-Spice code

```
Project code

.include p18_model_card.inc
.include p18_cmos_models_tt.inc

* Photo Diode handout *

.subckt PhotoDiode VDD N1_R1C1
        I1_R1C1 VDD N1_R1C1 DC Ipd_1 ! Photo current source
        d1 N1_R1C1 VDD dwell 1 ! Reverse biased Diode
        .model dwell d cj0=1e-14 is=1e-12 m=0.5 bv=40 ! Diode model
        Cd1 N1_R1C1 VDD 30f! Photo diode capacitor
.ends

* Parameters *

.param L_M1 = 1.07u
.param W_M1 = 1.09u
.param L_M2 = 1.07u
.param W_M2 = 1.09u
.param L_M3 = 0.37u
.param W_M3 = 5.03u
.param L_M4 = 0.37u
.param W_M4 = 5.03u

.param L_MC = 1.07u        ! L_MC1 = L_MC2
.param W_MC = 1.09u        ! W_MC2 = W_MC2

.param CS_C = 2.75p
.param CC_C = 3p

* Test bench handout *

.param Ipd_1 = 50p ! Photodiode current, range [50 pA, 750 pA]
.param VDD = 1.8 ! Supply voltage
.param EXPOSURETIME = 2m ! Exposure time, range [2 ms, 30 ms]

.param TRF = {EXPOSURETIME/100} ! Risetime and falltime of EXPOSURE and ERASE
↪  signals
.param PW = {EXPOSURETIME} ! Pulsewidth of EXPOSURE and ERASE signals
.param PERIOD = {EXPOSURETIME*10}! Period for testbench sources
.param FS = 1k; ! Sampling clock frequency
.param CLK_PERIOD = {1/FS} ! Sampling clock period
.param EXPOSE_DLY = {CLK_PERIOD} ! Delay for EXPOSE signal
.param NRE_R1_DLY = {2*CLK_PERIOD + EXPOSURETIME} ! Delay for NRE_R1 signal
.param NRE_R2_DLY = {4*CLK_PERIOD + EXPOSURETIME} ! Delay for NRE_R2 signal
.param ERASE_DLY = {6*CLK_PERIOD + EXPOSURETIME} ! Delay for ERASE signal

VDD VDD 0 dc VDD
VEXPOSE EXPOSE 0 dc 0 pulse(0 VDD EXPOSE_DLY TRF TRF EXPOSURETIME PERIOD)
VERASE ERASE 0 dc 0 pulse(0 VDD ERASE_DLY TRF TRF CLK_PERIOD PERIOD)
VNRE_R1 NRE_R1 0 dc 0 pulse(VDD 0 NRE_R1_DLY TRF TRF CLK_PERIOD PERIOD)
VNRE_R2 NRE_R2 0 dc 0  pulse(VDD 0 NRE_R2_DLY TRF TRF CLK_PERIOD PERIOD)
```

```
* Pixel Circuit *

.subckt PixelCircuit VDD VSS EXPOSE ERASE NRE OUT N2
    XPD1 VDD N1 PhotoDiode

    M1 N1 EXPOSE N2 VSS NMOS L=L_M1 W=W_M1
    M2 N2 ERASE VSS VSS NMOS L=L_M2 W=W_M2
    M3 VSS N2 N3 VDD PMOS L=L_M3 W=W_M3
    M4 N3 NRE OUT VDD PMOS L=L_M4 W=W_M4

    CS N2 VSS CS_C
.ends

* Active Load *

.subckt ActiveLoad VDD VSS OUT
    MC OUT OUT VDD VDD PMOS L=L_MC W=W_MC

    CC OUT VSS CC_C
.ends

* Camera Circuit *

XPixel_1_1 VDD 0 EXPOSE ERASE NRE_R1 OUT1 N2 PixelCircuit
XPixel_2_1 VDD 0 EXPOSE ERASE NRE_R2 OUT1 N2 PixelCircuit
XPixel_1_2 VDD 0 EXPOSE ERASE NRE_R1 OUT2 N2 PixelCircuit
XPixel_2_2 VDD 0 EXPOSE ERASE NRE_R2 OUT2 N2 PixelCircuit
XActiveLoad_1 VDD 0 OUT1 ActiveLoad
XActiveLoad_2 VDD 0 OUT2 ActiveLoad

* Plot *

.plot V(N2) V(EXPOSE) V(ERASE) V(NRE_R1) V(NRE_R2) V(OUT1) V(OUT2)
```

B Verilog code

```
`timescale 1ms / 1ps


module FSM_ex_control (
        input logic Clk,
        input real Clk_half_period,
        input logic Reset,
        input logic Init,
        input logic Ovf4,
        input logic Ovf5,
        input reg [4:0]        Exp_time,
        input logic ADC_control,
        output logic NRE_1,
        output logic NRE_2,
        output logic ADC,
        output logic Expose,
        output logic Erase,
        output logic Start
        );

        typedef enum logic [2:0] { Idle, Exposure, Readout1, Readout2, Wait_exp,
        ↪  Wait_r1, Wait_r2 } State;

        State currentState, nextState;

        always_ff @(posedge Clk) begin
                currentState = nextState;
        end

        always@(*) begin
                ADC = ADC_control;

                case(currentState)
                        Idle: if(Init && !Reset)
                                nextState = Exposure;
                            else
                                nextState = Idle;
                        Exposure: if (Reset)
                                nextState = Idle;
                            else if (Ovf5)
                                nextState = Wait_exp;
                            else
                                nextState = Exposure;
                        Wait_exp: if (Reset)
                                        nextState = Idle;
                                else begin
                                        #(2*Clk_half_period);
                                        nextState = Readout1;
                                end
                        Readout1: if(Reset)
                                nextState = Idle;
                            else if (Ovf4)
                                nextState = Wait_r1;
                            else
                                nextState = Readout1;
                        Wait_r1: if (Reset)
```

```
                                        nextState = Idle;
                                    else begin
                                        #(2*Clk_half_period);
                                        nextState = Readout2;
                                    end
                            Readout2: if (Reset)
                                        nextState = Idle;
                                    else if(Ovf4)
                                        nextState = Wait_r2;
                                    else
                                        nextState = Readout2;
                            Wait_r2: if (Reset)
                                        nextState = Idle;
                                    else begin
                                        #(2*Clk_half_period);
                                        nextState = Idle;
                                    end
                            default:
                                    nextState = Idle;
                    endcase
            end

            assign Expose = (currentState == Exposure);
            assign Erase = (currentState == Idle);
            assign NRE_1 = !(currentState == Readout1);
            assign NRE_2 = !(currentState == Readout2);
            assign Start = !(currentState == Idle);

endmodule


module CTRL_ex_time (
            input logic Clk,
            input logic Reset,
            input logic Exp_increase,
            input logic Exp_decrease,
            output reg [4:0] Exp_time
            );

            initial
                    Exp_time = 3;

            always @ (posedge Clk) begin
                    if (Exp_increase && Exp_time < 30)
                            Exp_time <= Exp_time+1;
                    else if (Exp_decrease && Exp_time > 2)
                            Exp_time <= Exp_time-1;
            end
endmodule


module Timer_counter(input logic Start,
            input logic Clk,
            input logic Reset,
            input reg [4:0] Initial,
            output logic Ovf4,
            output logic Ovf5,
            output logic ADC_control);
```

```
integer Exp_counter = 0;
integer Read_counter = 0;
logic Read_count1 = 0;
logic Read_count2 = 0;
integer Read_time = 3;

initial begin
        Ovf4 = 0;
        Ovf5 = 0;
end

always_comb begin
        if ((Read_count1 || Read_count2) && Read_counter == 3 && Start)
                ADC_control = 1;
        else
                ADC_control = 0;
end

always @(posedge Clk) begin
        if ((Ovf5 || Read_count1) && Start) begin
                if (Start && (Read_counter < Read_time)) begin
                        Read_counter = Read_counter + 1;
                        Read_count1 = 1;
                        Ovf5 = 0;
                end
                else if (Start && (Read_counter == Read_time)) begin
                        Ovf4 = 1;
                        Read_counter = 0;
                        Read_count1 = 0;
                        end
        end
        else if (Start && (Ovf4 || Read_count2)) begin
                if (Start && (Read_counter < Read_time)) begin
                        Read_counter = Read_counter + 1;
                        Read_count2 = 1;
                        Ovf4 = 0;
        end
        else if (Start && (Read_counter == Read_time)) begin
                Ovf4 = 1;
                Read_counter = 0;
                Read_count2 = 0;
                end
        end
        else if (Start && (Exp_counter < Initial-2 && !Ovf4)) begin
                Exp_counter = Exp_counter + 1;
        end
        else if (Start && (Exp_counter == Initial-2 && !Ovf4)) begin
                Ovf5 = 1;
                Exp_counter = 0;
        end
        else begin
                Ovf4 = 0;
                Read_counter =
                ↪  0;
                Read_count1 = 0;
                Read_count2 = 0;
        end
```

```
        end
endmodule
```

C Verilog test bench

```
`timescale 1ms / 1ps

module system_testbench;

        logic Init, Exp_increase, Exp_decrease, Clk, Reset; // input
        logic NRE_1, NRE_2, ADC, Expose, Erase;                                 //
        ↪  output
        logic Ovf, Start;
        real Clk_half_period = 0.5;
        reg [4:0] Initial;


        always
                begin
                        Clk <= 1; #Clk_half_period;
                        Clk <= 0; #Clk_half_period;
                end

        FSM_ex_control dut_FSM(
                .Clk(Clk),
                .Clk_half_period(Clk_half_period),
                .Reset(Reset),
                .Init(Init),
                .Ovf4(Ovf4),
                .Ovf5(Ovf5),
                .ADC_control(ADC_control),
                .Exp_time(Initial),
                .NRE_1(NRE_1),
                .NRE_2(NRE_2),
                .ADC(ADC),
                .Expose(Expose),
                .Erase(Erase),
                .Start(Start));

        CTRL_ex_time dut_CTRL_ex_time(
                .Clk(Clk),
                .Reset(Reset),
                .Exp_increase(Exp_increase),
                .Exp_decrease(Exp_decrease),
                .Exp_time(Initial));


        Timer_counter dut_Timer(
                .Start(Start),
                .Clk(Clk),
                .Reset(Reset),
                .Initial(Initial),
                .Ovf4(Ovf4),
                .Ovf5(Ovf5),
                .ADC_control(ADC_control));


        initial
                begin
```

```
                        Reset <= 1; Init <= 0; Exp_increase <= 0; Exp_decrease <=
                        ↪  0; #(2*Clk_half_period);

                        Reset <= 0; Init <= 1;  #(2*Clk_half_period);

                        Init <= 0; #(14*Clk_half_period);

                end

endmodule
```