Design and Test of 16x16 Pixel CMOS Image Sensor Chip with On-Chip ADC in 0.18um Process

Line Le

Thesis submitted for the degree of
Master in Electrical Engineering, Informatics and Technology
(Microelectronics and Sensor Technology)
60 credits

Department of Physics
The Faculty of Mathematics and Natural Sciences

UNIVERSITY OF OSLO

Autumn 2022

Design and Test of 16x16 Pixel CMOS Image Sensor Chip with On-Chip ADC in 0.18um Process

University of Oslo

Oslo, Norway

Master Thesis

Design and Test of 16x16 Pixel CMOS Image Sensor Chip with On-Chip ADC in 0.18um Process

Line Le

Supervised by:

Johannes Sølhusvik

Associate professor

Philipp Dominik Häfliger

Professor

Research Group for Nanoelectronic Systems

Design and Test of 16x16 Pixel CMOS Image Sensor Chip with On-Chip ADC in 0.18um Process

`http://www.duo.uio.no/`

Printed: X-press printing house

Contents

List of Figures

List of Tables

Abstract

A part of the NANO group at the University of Oslo delves into the research and development of CMOS image sensors. Several iterations of image sensors have been fabricated over the years, focusing on different sub-systems.

For the ”CMOS Image Sensor Design” course, providing a CMOS image sensor chip for demonstration and analysis is useful in educational contexts. Industrial chips are non-disclosed and thus, unavailable for analysis by students participating in the course. Therefore, a disclosed image sensor chip was designed in this project with the objective of creating an entire image sensor system from scratch, following conventional architectures, methods, and topologies.

A prototype ASIC is fabricated in 0.18 µm technology. It features a 16 x 16 pixel array with a simple readout circuit and column-parallel ADCs. Row and column decoders are implemented to address the pixels. The ASIC is manufactured and packaged in a J-Leaded Chip Ceramic package with 84 pins.

A PCB is designed and produced as an interface between the chip and the external world. A characterization system was to be made by another student, but this system was not available during testing. Hence, to provide control and measurement signals, a PYNQ-Z1 board featuring a hard core processor and an FPGA is used. Preliminary test and debugging results verify the circuits’ basic functionalities. However, full system verification and image capturing is not possible, and some future work is needed to establish such a procedure.

Acronyms

ADC Analog-to-Digital Converter

APS Active-Pixel Sensors

APSoc All-Programmable System on Chip

ASIC Application-Specific Integrated Circuit

CCD Charge-Coupled Devices

CDS Correlated Double Sampling

CG Conversion Gain

CIS CMOS Image Sensor

CMOS Complementary Metal–Oxide–Semiconductor

COL Column Voltage

DCDS Digital Correlated Double Sampling

DFF D-Flip-Flop

DNL Differential Non-Linearity

DR Dynamic Range

FD Floating Diffusion

FPGA Field Programmable Gate Array

FPN Fixed Pattern Noise

ICMR	Input Common Mode Range
INL	Integral Non-Linearity
MUX	Multiplexer
PCB	Printed Circuit Board
PD	Photodiode
REF	Reference Voltage
RST	Reset Transistor
SAR	Successive-Approximation ADC
SEL	Selection Transistor
SF	Source Follower
SIG	Signal Voltage
SS-ADC	Single-Slope ADC
TX	Transfer Gate Transistor

Chapter 1

Introduction

Image sensors are used everywhere; from mobile phone cameras to satellite imagery, as well as the autonomous industry. Today, we are able to capture the image of everything from small microsopic cells to large astronomical objects in space. To the average consumer, it would seem like the imaging sensor industry had already reached its peak. Such a thought would hardly be questioned when we think of the monumental advancements since the early technologies.

In 1969, the CCDs were invented at the Bell Labs by Drs. Williard Boyle and George E. Smith [4]. CCDs have since been utilized in all areas of image sensor applications, and are still in use today – especially in high end applications.

As the microfabrication technologies got more advanced in the 90s, the modern CMOS image sensor was adopted. The Active-Pixel Sensors (APS) were implemented in CMOS, which allowed integration of the pixel array, controlling circuits, signal processing units and ADCs on a single integrated chip. This is commonly referred to as monolithic integration, and could very well mark a new era for image sensors.

The well known digital camera can be considered as the first generation of image sensor applications. The second generation is the incorporation of image sensors in mobile devices, surveillance, and a myriad of other areas. Today, the third generation is emerging, where the objectives range from the development of image sensors for biomedical sensors or other applications which emphasize the necessity of smaller and smaller chips.

Modern image sensor technology has reached advanced levels, but challenges remain – e.g. where low light intensities are of concern, or simply when optimization of frame rate, fill factor, dynamic range, low power, and small size are important factors. Hence, there is a strong reason to argue that the industry will always need expertise in the development and research of CMOS image sensors.

1.1 Preface

This thesis encapsulates a 60 credit-project for the completion of a master's degree in the Electronics Engineering, Informatics, and Technology programme at the University of Oslo. With guidance from Adjunct Professor, Johannes Sølhusvik and Professor Philipp Dominik Häfliger, an image sensor chip was designed and fabricated.

With advice, aid, and collaboration with post-docs and fellow students, the chip was tested and verified.

It's certainly been a challenging journey, but I learned more than I could ever imagine within mixed signal design, PCB design, FPGA programming, electronics/chip testing and verification. Of course, not without the support and advice available to me.

In this regard, I wanna extend my gratitude to everyone that were present during this period. To my supervisors, for pushing me out of my boundaries and guiding my project in the right direction. To fellow students, PhD candidates and post-docs who would be open to any discussion over the millionth coffee break that would stretch into the latest of hours. Without those, the past years would not be as memorable.

Last but not least, I would like to thank my friends and family who gave me encouragement and support along the way.

1.2 Objectives

- Present the theory detailing the background of the system
- Define functional specifications of the system
- Design and fabricate the ASIC in a CMOS process
- Design and produce the PCB
- Program an APSoC
- Test and verify the system

1.3 Thesis Outline

Diagram 1.1 illustrates the main tasks within this thesis.

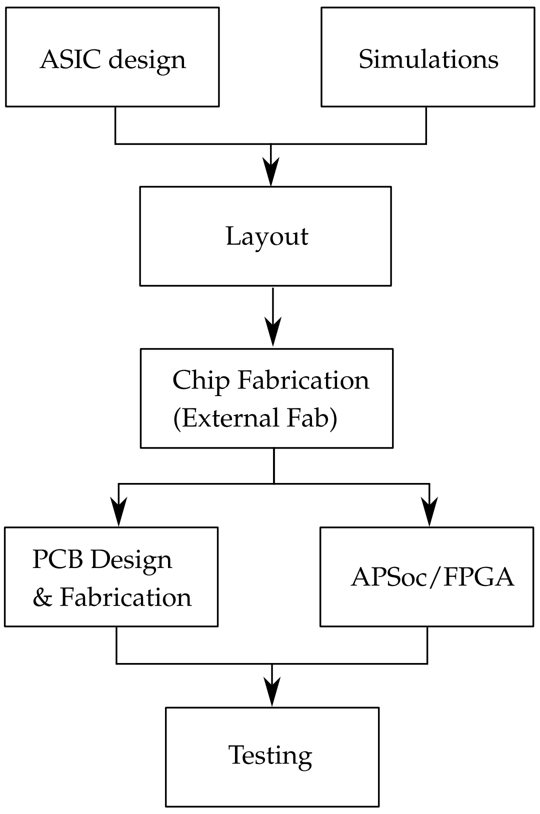


Figure 1.1: Thesis outline diagram

The portion involving the ASIC includes design, simulation, and testing of the following components:

- Pixel array including:
  - Pixel
  - Source follower
- Level clamper circuit
- Analog-to-digital converter including:
  - Comparator
  - Counter
- Row decoder
- Column decoder
- Pad frame

The tasks involving testing are:

- PCB design and production
- Programming on an All Programmable System on Chip (APSoC)

1.4 Scope of Work

The project scope is quite large, as it involves mixed signal design, PCB design, and embedded system development. In light of this, a part of the work is re-use from projects done by other students.

In the ASIC design, some of the digital circuitry is re-used. The row decoder was designed and verified by Mathias H. F. Wilhelmsen, as detailed in [16].

A custom lens mount is required to hold the lens and fit on top of the chip socket on the PCB. Such a mount was designed and 3D printed by Jens Bulukin, and also re-used in this project.

It was originally the aim to re-use as much as possible from previous works done, especially by copying some of the layout, as this would free up a lot of time to delve deeper into specific topics and sub-systems. However, building everything from scratch turned out to be the easier option since previous projects were designed in the AMS OPTO 0.35 µm process, rather in the XFAB 0.18 µm process. Hence, copying the layout was virtually impossible.

It should therefore be noted that this project covers the breadth of the subject of image sensors, rather than the depth. Designing, fabricating, and testing a complete image sensor chip would, under normal circumstances, be a collaborative work between at least two students who could dive deeply into separate topics. Needless to say, this project is an exploration of the overall CMOS Image Sensor (CIS) system design rather than single block/component design.

Chapter 2

Background

2.1 CMOS Image Sensors

An image sensor consists of elements that convert light into electrical signals. CMOS image sensors are composed of complex systems with devices such as photodetectors and transistors in each pixel. On an industrial image sensor, there are typically millions of transistors along with analog and digital drivers and readout circuitry. Normally, the the readout circuitry is placed outside the sensor area, and the image sensor function and image processing function can be combined and created within the same integrated circuit.

There are a number of metrics that are used for image sensors, and this thesis will take only a fraction into account. Categorizing the metrics of an image sensor can be done in three ways; metrics related to the pixel layout, metrics related to the pixel physics, and the metrics related to the pixel readout.

2.1.1 Silicon in Image Sensors

Almost all image sensors are produced using silicon, because silicon, which is a semiconductor, has a band-gap energy of 1.1 eV that is perfect for capturing light in the visible and near infrared spectrum. If a photon with an energy larger than the band-gap energy of 1.1 eV hits the silicon, it will be absorbed in the material and produce a charge. The resulting charge depends on the sensitivity of the sensor (or the quantum efficiency, which will soon be elaborated on).

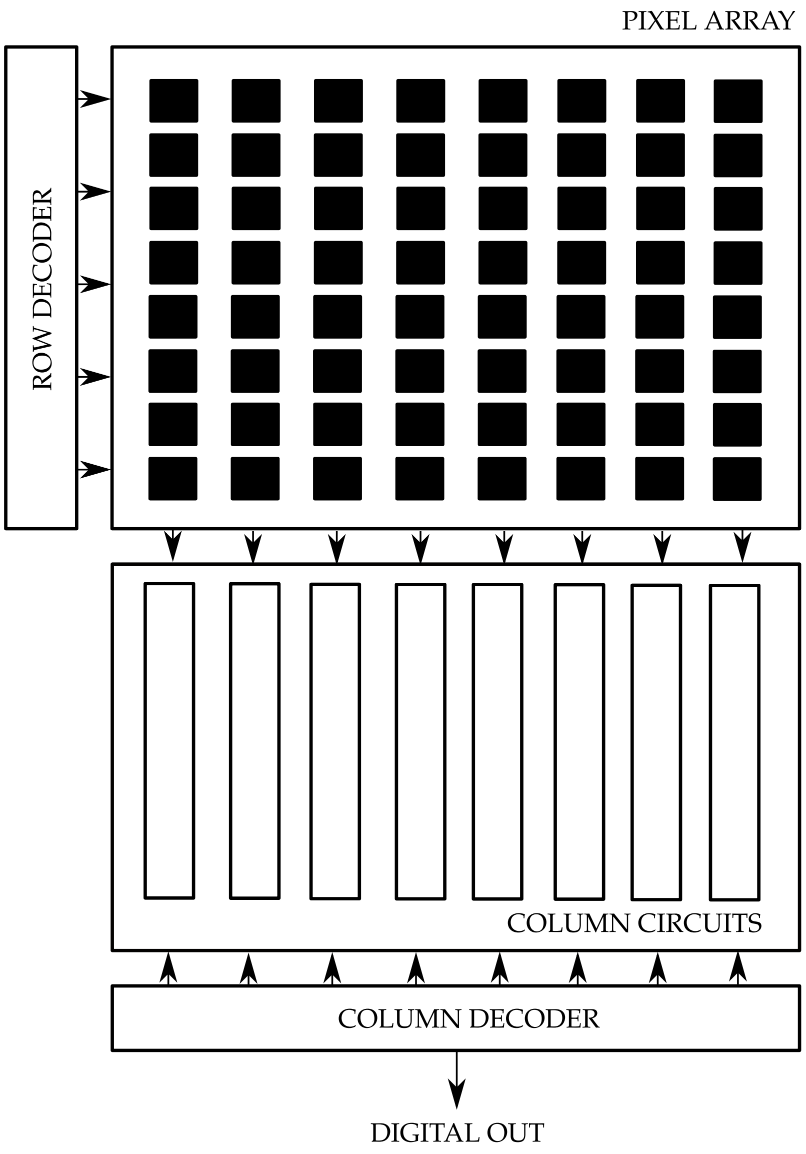


Figure 2.1: Schematic overview of image sensor with pixel array, readout circuitry, depicting rows and columns

The energy of a photon is expressed in equation 2.1, where Planck's constant, h, multiplied by the speed of light, c, is divided by the wavelength of the light, λ.

$$E = \frac{hc}{\lambda} \tag{2.1}$$

The wavelengths within the visible spectrum are between rougly 450 nm to

650 nm, which corresponds to 2.75 eV and 1.9 eV, respectively. These wavelengths are absorbed exponentially. Hence, blue light is absorbed closer to the surface of the silicon, while red light is absorbed deeper within the silicon layers.

2.1.2 Photodiodes

A photodiode in reverse bias is essentially two capacitive plates, and therefore the photocurrent is integrated onto the photodiode's own capacitance. The voltage change across the capacitor is subsequently read out.

Integration mode is the most convenient detection method for photodetectors. Firstly, in solid-state imagers, the number of generated electron-hole pairs are small, which results in a photocurrent smaller than 1 pA [12]. It is not easy to design interface electronics with sufficiently small size that is able to detect such small current changes.

Secondly, in an imager there is an array of photodiodes, and read-out is most commonly done row-by-row. There is usually a very large number of rows in which each diode must be read out in a short time span. By this reason, it is easier to read out from an integrating capacitor, as the energy stored on the capacitor is larger than the instantaneous energy generated by the photodiode.

A photodiode is shown in figure 2.2. The operation of this element is as follows: A reset voltage is applied on the reversed diode by the means of a reset switch, resulting in an efficient method of sampling a reset voltage onto the parasitic capacitance (C_d in the figure). When the photodiode is exposed to light, the biasing voltage will decrease, and the photocurrent is integrated onto the capacitor. This photocurrent is directly proportional to the light intensitity, meaning that the voltage decrease across the photodiode also is directly proportional to the light in addition to the integration time. Hence, measuring the voltage over the diode after an integrating time is a measuring of the amount of light.

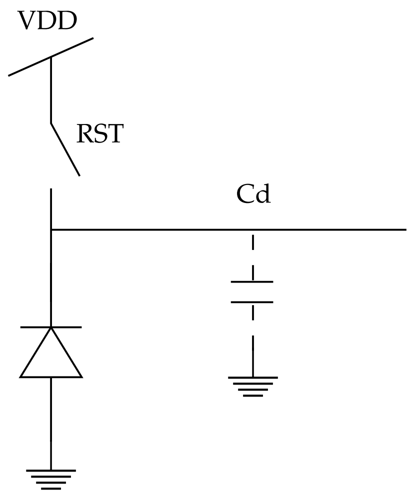


Figure 2.2: Reverse photodiode with reset switch and parasitic capacitance, Cd

2.1.3 Photo-Electric Effect

The photo-electric effect is the underlying principle of photosensitive elements, such as the aforementioned photodiode. The phenomena is illustrated in figure 2.3, and involves the interaction between electromagnetic radiation and matter [12]. In solid-state imagers, the radiation is visible light and the matter is a semiconductor. Fundamentally, the semiconductor is exposed to light, and the incident photons will transfer their energy to individual atoms. This will result in the generatation of electron-hole pairs. The process requires that the wavelength of light is within thresholds for electron-hole pair generation.

When separating electrons and holes, recombination of pairs must be avoided in order to generate a charge. The easiest way is to create an electric field in the form of a p-n junction of a diode. This junction is called the depletion region where electrons drift to the surface of the semiconductor while holes are repelled by it. A current, (often referred to as the photocurrent), is generated.

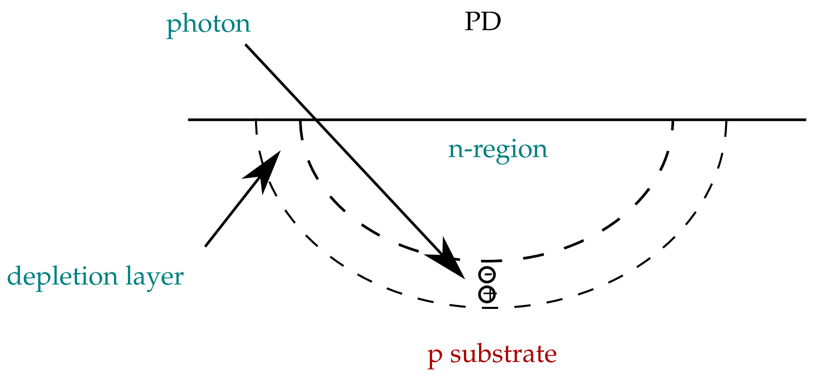


Figure 2.3: An illustration of the photo-electric effect

2.1.4 Photodiode With a Transfer Gate

Figure 2.4 is an illustration of the cross-section of a photodiode with a transfer gate. It consists of the photosensitive element itself, the photodiode (PD), a transfer gate (TX), the reset gate (RST), a diffusion layer and the floating diffusion point (FD).

The photosensitive element, PD, is positively biased in order to generate an electric field that attracts electrons to the surface in the diffusion layer. By the same concept, the holes are repelled from the surface.

The FD is first and foremost reset to a bias voltage. It can be considered as a capacitance sampling the reset voltage. In effect, a reset noise is introduced. The readout circuit will sample this reset noise to compensate digitally.

Photons will ”kick” electrons to the surface in the diffusion layer, which creates a current proportional to the amount of electrons. By pulsing the TX gate, the charge will be transferred to the FD node. The voltage in the FD node is proportional to the PD charge added with the reset noise.

As a result, due to the floating diffusion capacitance, sampling both the reset noise and the value generated by light is possible. This technique is called Correlated Double-Sampling (CDS), and is used in both CCD and CMOS image sensors to eliminate the reset noise.

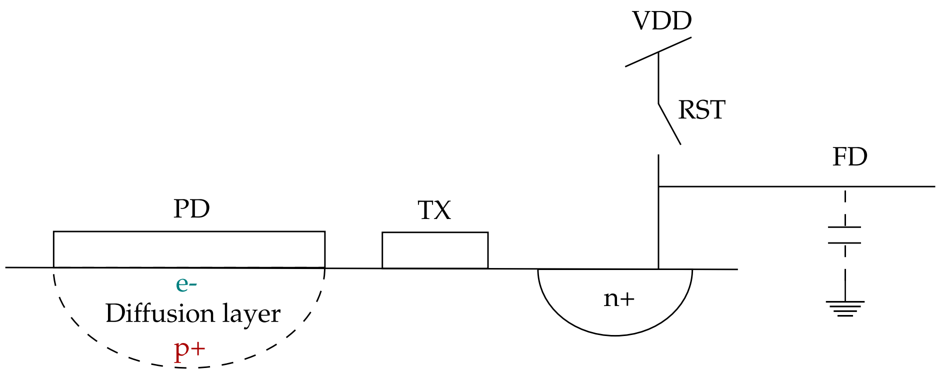


Figure 2.4: Cross-section of a photodiode with a transfer gate, reset transistor, and row select transistor.

2.1.5 Pinned Photodiode

The invention of pinned photodiodes resulted in a dramatic reduction in dark current noise. Figure 2.5 shows a cross-section of a pinned photodiode. A shallow p+ layer has been implanted near the silicon surface, which "pins" the cathode of the photodiode to the substrate. This results in higher optical sensitivity and lower dark current. On the other hand, because the n-/p junction must be extended into the depletion layer of p+/n- junction, and n- has to be fully depleted, this element is very difficult to fabricate. Doping levels must be controlled very accurately. Nevertheless, pinned photodiodes are very commonly used in commercial image sensors.

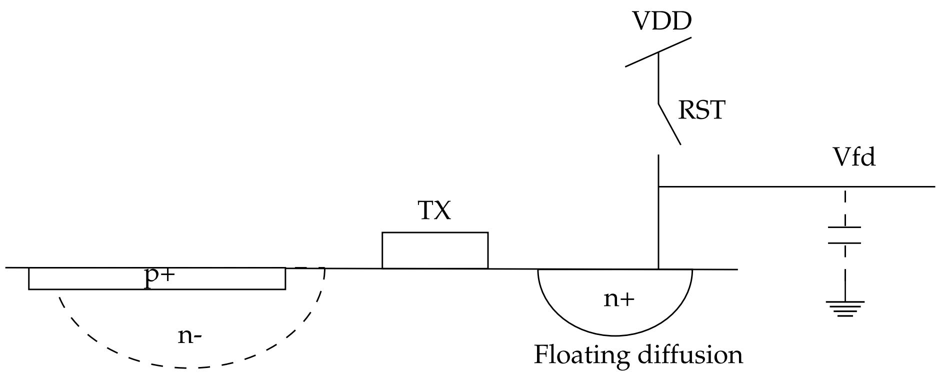


Figure 2.5: Cross-section of a pinned photodiode

2.2 Pixel Layout Metrics

2.2.1 Pixel Count

At first glance, when comparing image sensors, the pixel count might be the most interesting metric. The pixel count is subject to the available chip area, and the amount of chip area that can be dedicated to pixels is dependent on the area that must be allocated for digital logic and readout circuitry.

Hence, when talking about pixel arrays, the pixel size must be considered. This is commonly expressed in μm^2, and the rule is that the smaller the size, the more pixels can fit on the sensor.

A drawback is that smaller pixels means smaller area for incident photons, which can be significant for certain applications such as sensors for low light observation [18].

2.2.2 Fill Factor

The pixel fill factor is the percentage of pixel area that is dedicated to the actual photodiode. A fill factor of 100 % is ideal, making the entire pixel area available for photon collection. However, most designers need to compromise on the fill factor in order to make space for the charge transfer circuitry.

Using backside illuminated (BSI) sensors, the readout circuitry can be placed below the photodiode layer. Hence, the whole pixel area can be used for photon collection. On the other hand, the manufactoring process is more complicated and costly.

Most modern image sensors utilize microlenses as a means of improving the fill factor. This is achieved by placing individual lenses on top of each pixel in order to focus the incident light onto the photosensitive portion of the pixel.

2.3 Pixel Physics Metrics

2.3.1 Quantum Efficiency

Quantum Efficiency is an important metric that describes how many photons are absorbed and turned into electron-hole pairs. It is measured as a percentage, and 100 % is the ideal. This would imply that every photon would be converted into charge, regardless of wavelength.

Gathering of the photocurrent is an integrative process, and is subject to the capacitance of the photodiode. This capacitance depends on the dopings, and the dopings are mainly based on three factors; the process by sensor manufacturers, the pixel area made by the layout designers, as well as the bias operating voltage.

2.3.2 Well Capacity

The capacitance of the photodiode sets a limit on the total charge one pixel can hold during the integration process, which gives rise to the concept of the well capacity. It is usually given in electrons, and describes how well a sensor can measure an accurate signal from low intensities up to the maximum intensitites – known as the full-well capacity.

Equation 2.2 is a mathematical representation of the full-well capacity, where N_{sat} is the total amount of electron charge that can be accumulated on a saturated photodiode.

$$N_{sat} = \frac{1}{q} \int_{V_{max}}^{V_{reset}} C_{PD}(V) \cdot dV \tag{2.2}$$

Dynamic Range

What follows the concept of full well capacity is the dynamic range, measured in decibels. This is a metric that describes how well the sensor can operate in varied light conditions. The higher the dynamic range, the better it will operate.

The true definition can be rather ambigious, as the DR can be defined in two ways: the range from saturation to the noise floor, or the ratio between the highest linear intensity and the noise floor.

$$DR = 20 \cdot log\left(\frac{N_{sat}}{n_{read}}\right) \tag{2.3}$$

$$DR = 20 \cdot log\left(\frac{N_{lin}}{n_{read}}\right) \tag{2.4}$$

It has been shown that the dynamic range is determined by the maximum signal swing and the noise of the front-end circuit [12].

Conversion Gain

The dynamic range gives rise to the conversion gain (CG). CG is an important parameter for charge detection, and is synonymous to the sensitivity of the detection. It is by definition a measure of the change in the output voltage, caused by a single absorption of charge – or an electron. CG is determined by the electrostatic capacitance (C_{FD}), of the FD node and the source follower circuitry gain.

Equation 2.5 is an expression of CG, where A_v is the pixel signal gain seen from the output, q is the elementary charge, C_{FD}, is the floating diffusion capacitance.

$$CG = A_v \cdot \frac{q}{C_{FD}} \left[\frac{\mu V}{e^-}\right] \tag{2.5}$$

Equation 2.6 shows a dependency on the capacitance. A smaller well capacity means a lower capacitance, resulting in a larger voltage change when a charge is absorbed. This suggests that the conversion gain is proportional to the well size, i.e. $A_v \propto \frac{1}{C}$.

$$V_{out} = \frac{q}{C} \tag{2.6}$$

Increasing CG also indicates stronger pixel signal, which gives potential to improved SNR. On the other hand, higher CG reduces effective full-well capacity when there is a limited power supply voltage, which puts a restraint on the dynamic range.

Dark Current Noise

With smaller and smaller pixels, there is an increasing problem of more and more significant noise. Dark current arises from thermal energy generated from three main sources; firstly, the irregularities in the silicon lattice at the surface of the photodiode, secondly, the currents generated from depletion region formation, and lastly, the currents caused by diffusing charges in the bulk of silicon [3].

Dark current is the charge that is integrated over a photodiode, but is not caused by photon absorption in the silicon. It is coming from other sources within the sensor, and is measured in fA. Equation 2.7 indicates that dark current is dependent on the exposure or the chosen integration time.

$$N_{dark} = \frac{Q_{dark}}{q} = \frac{I_{dark} \cdot t_{exposure}}{q} \tag{2.7}$$

As dark current is independent from external light, it will still appear in zero light conditions. Therefore, it can be easily observed when capturing images in total darkness.

Fixed Pattern Noise

Non-uniformities in the sensor manufacture is inevitable. Threshold voltages are observed to be varying across the readout transistors. Widths and lengths tend to vary, and there are observations of gradients in doping. In addition, electron mobility behaves different from pixel to pixel. The resulting artifact is known as the fixed pattern noise, and is known as a type of spatial noise.

Other noteworthy variations are seen in power supplies which can be a cause horizontal fixed pattern noise, and diversities in the row reset voltages and readout ciruitry may contribute a significant amount to the vertical fixed pattern noise.

Temporal Noise

Temporal noise is also a significant contributor to image artifacts. Sources include white noise and ficker noise from transistors in the source follower, noise in the MOSFET switch that is sampled on sample-and-hold (S/H) capacitors, and the reset noise originating from resetting the floating diffusion at pixel level [17]. Photon shot noise and dark current noise are also a part of the temporal noise sources.

2.4 Source Follower

Every CIS comes with in-pixel nearly unity-gain source follower, which is used to drive the array bitline. An illustration is shown in figure 2.6. The source follower consists of two NMOS transistors in series. M1 is a follower, and M2 is a current sink. Hence, V_{gs} "follows" V_{in}.

The gain is given by equation

$$A_v = \frac{g_m}{g_m g_m + g_{mb} + \frac{1}{r_{o1}} + \frac{1}{r_{o2}}}$$

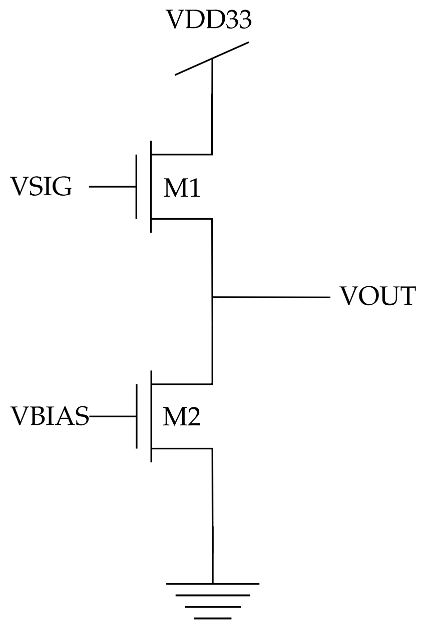


Figure 2.6: Schematic of .source follower

2.5 4T Pixel

The 4T pixel is depicted in figure 2.7 and is comprised of a photodiode (PD) with a floating diffusion (FD) output. PD and FD are separated by a transfer gate (TX), and a reset transistor (RST) is included for resetting the FD node. The FD node is then followed by the input transistor to a source follower (SF). Lastly, a row selection transister (SEL) is connected to the output.

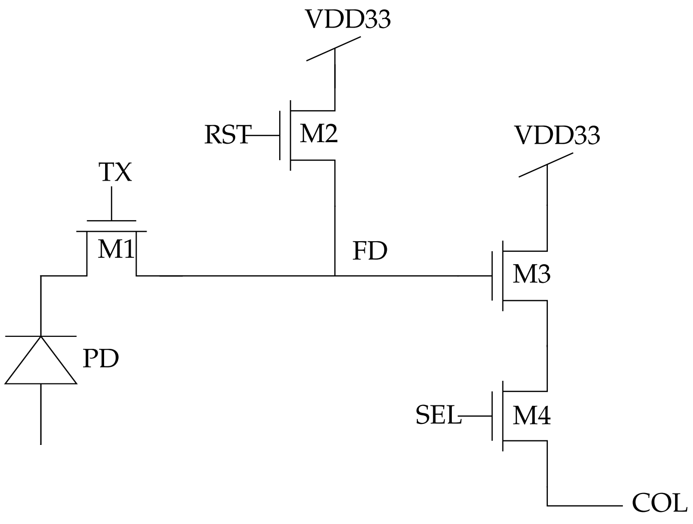


Figure 2.7: Schematic diagram of 4T Pixel

2.6 Row Decoder

A row decoder translates a given address to a specific row in the pixel array, and makes sure that the routing of reset and row select signals are applied to the desired pixels.

2.7 Column Circuit Readout

Since the pixel array is read out row-by-row, a column readout circuitry is needed. Each sample is stored on a capacitor in the form of a clamper circuit. The clamper circuit will "clamp" a signal to a different DC voltage level. As seen in figure 2.8, it consists of a capacitor and an NMOS that functions as a switch. There exists a variety of clamper circuit configurations. In this design, the clamper binds the upper extreme of a waveform to a fixed DC voltage level. A bias voltage is applied to the gate of the NMOS, and binds the upper voltage limit to this bias voltage.

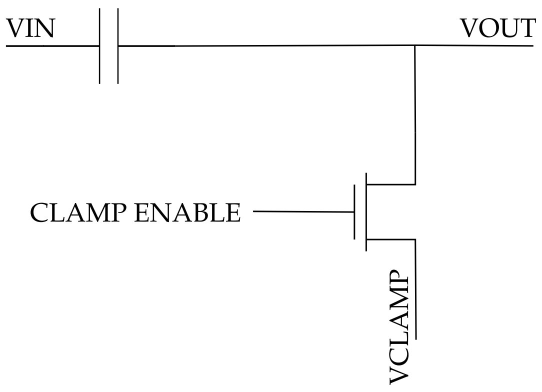


Figure 2.8: Circuit diagram of a clamper circuit

2.7.1 Column Decoder

In addition to row decoding, column decoders are needed as well for the pixel readout. Column decoders are usually schematically simpler than the row decoders. On the other hand, it has much higher speed requirements as it needs to address every pixel in a row.

2.8 Electronical Shutter

One of the important camera components is its shutter, which can be mechanical or electronic. CMOS cameras use electronic shutters, and there are mainly two different techniques: rolling shutter and global shutter. With rolling shutter, the captured scene is scanned across very rapidly and is done row by row.

Reading out a row takes a certain amount of time, which introduces a small delay between each of them. Hence, with larger pixel counts, the readout time can easily add up. A drawback of the rolling shutter mechanism is the so-called rolling shutter artifacts. These are image artifacts that occur due to objects moving during a frame capture. To circumvent this, global shutter can be used. All pixels in an array are then read out simultaneously, meaning that each pixel signal is captured from one single point in time. As a result, there is no distortion to moving objects. On the other hand, the designer must sacrifice the pixel fill rate or resign with larger pixel structures.

2.9 Correlated Double Sampling

Correlated double sampling (CDS) is a noise suppression method where the difference between a reset voltage and a signal voltage is calculated to find the actual charge on a pixel. Figure 2.9 is an illustration that shows a reset level (RST), followed by a reference voltage (REF) and a signal voltage (SIG). The process of obtaining the actual voltage on a pixel, is to subtract the signal from the reference voltage as follows:

$$V_{PIX} = V_{REF} - V_{SIG}$$

CDS is widely used to suppress flicker ($1/f$) noise, and a commonly implemented in analog or mixed-signal circuits [6].

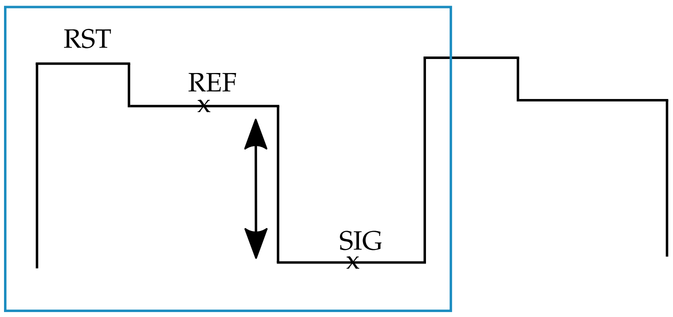


Figure 2.9: Time diagram of correlated double sampling

2.10 Analog-to-Digital Converter (ADC)

There are generally three architectures for ADC integration in CIS: a single-channel ADC, pixel-level ADC, and column-level ADC.

Single-channel ADC involves implementing one ADC for an entire array, which sets a strict requirement on the ADC speed for achieving high frame rates. On the other hand, pixel-level ADC makes use of one ADC per pixel which compromises silicon area and power consumption. Pixel-level ADCs can also carry non-uniformities, but these are usually less visible. Column-level ADCs consist of implementing an ADC per pixel column, and results in a good compromise between important parameters such as frame rate, fill factor, silicon area, and power consumption [8]. However, the main draw back of column-level ADCs are column non-uniformities which cause image artifacts that can be very visible to the naked eye. In addition, compared to chip-level ADCs, larger chip area is required [13].

2.10.1 Dynamic Range of ADC

The dynamic range is defined as the ratio between the largest and smallest values that the ADC can reliably measure. It is in other words, the number of bits that are used for the A/D conversion.

$$DR = 6.021 \cdot \mathrm{N} + 1.763\ \mathrm{dB} \tag{2.8}$$

where N is the number of bits.

2.10.2 Analog Comparator

A comparator is an intermediate element between the analog and digital domain. The analog comparator is a circuit that compares one analog signal with another analog signal or a reference voltage, and outputs a binary signal based on the comparison [1].

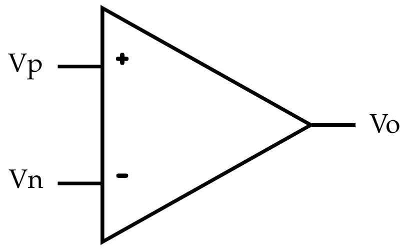


Figure 2.10: Circuit symbol for a comparator

2.10.3 Single-slope (Ramp) ADC

The single-slope ADC is widely used in CIS applications. It provides sufficiently high resolution and smaller area. However, according to [**Waldemar**, 8], the SS-ADC requires very high clock frequency which increases the power consumption.

An SS-ADC operates by receiving an input signal that is compared with a centrally generated ramp voltage. A digital counter is connected to latches in each column that stores the digital data. An illustration is seen in figure 2.11

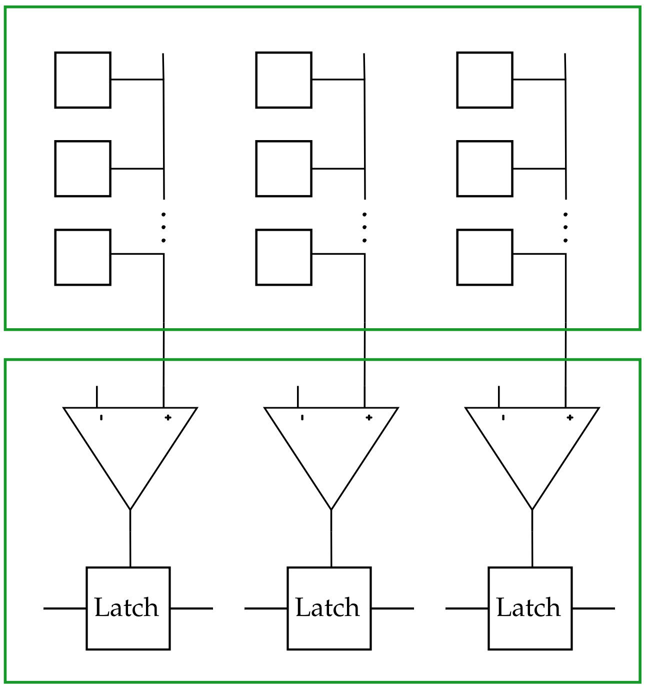


Figure 2.11: Column-Parallel ADC block diagram

Chapter 3

System Overview

An overview of the project components and its specifications will be presented in this chapter. Figure 3.1 on the following page shows the blocks implemented for this project. The first block is the image sensor implemented as an ASIC. The ASIC is designed with Cadence Virtuoso, simulated using the built-in Analog Design Environment tool (ADE Assembler), while the layout is designed using Virtuoso Layout Suite XL. The fabrication of the ASIC is done in a 0.18 um process, using XFAB's XH018 technology through EUROPRACTICE.

The second block involves the design of a printed circuit board (PCB), which will provide an interface between the ASIC and the external signals needed to operate the system. It also serves the important function of reading the signal output from the sensor, and making it easier to probe the signals with measurement instruments.

The last block is an All Programmable System on Chip (APSoC) that is essential for providing the PCB with digital control signals and for processing the data output from the ASIC.

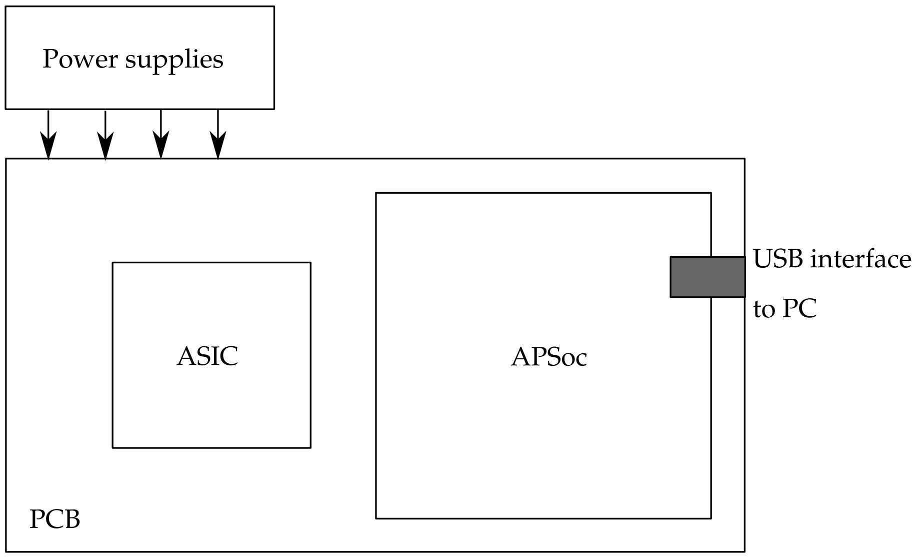


Figure 3.1: Block diagram of the system of this project featuring a PCB, ASIC, APSoC and external connections.

3.1 ASIC Image Sensor

Over the years, there have been a large number of possible variations in designing image sensors. The variations serve to maintain certain requirements within size, frame-rate, power-consumption, noise optimization, and resolution, just to name a few. All of these features are meant to be integrated onto one chip, and as transistor size has decreased, the complexity has steadily increased.

For this project, it was decided that a less complex implementation would be sufficient, meaning the pixel array itself, a small analog readout circuitry, and a more elaborate digital readout chain with an on-chip ADC. The complete readout chains, both analog and digital, are located outside of the photosensitive core of the ASIC.

To keep it simple, some parts, especially involving the digital readout circuitry is taken from already tested topologies implemented by previous master's students over the years. In analog design, ADC design is within a different scope on its own and would normally involve extensive use of theoretical, analytical, and experimental research and methods. For this project, a simple and proven ADC topology is used without any deep-diving into the topic.

Conclusively, the goal of this project is not to optimize any metrics within the image sensor, but to create a full system that takes the very first analog signal and propagates it into a detectable, digital bit.

Therefore, the focus is not on re-inventing the wheel or to add any additional layers of complexity when achieving the goal, but rather to stick to existing and proven methods of creating an image sensor.

Figure 3.2 is an illustration of the final chip with the components that were implemented. The components include a pixel array with a digital row decoder for addressing each row, analog readout circuitry consisting of a clamper circuit, followed by another chain of digital readout circuitry consiting of ADCs and a column decoder.

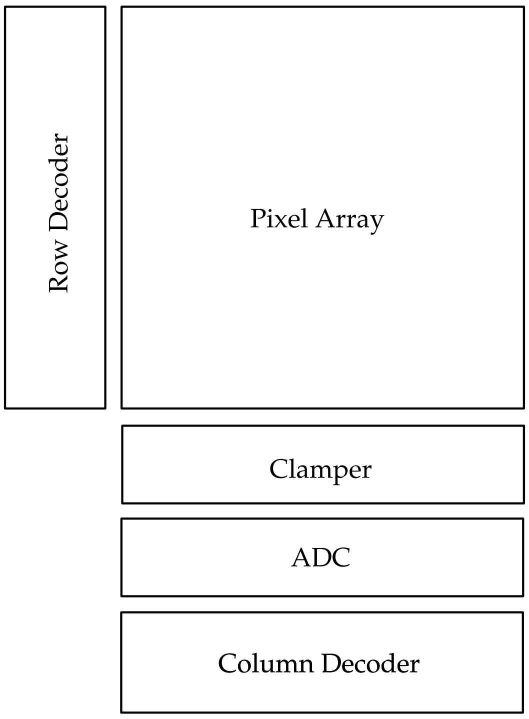


Figure 3.2: System overview of the ASIC, showing the pixel array, decoders, and readout chain consisting of clamper and ADC.

3.1.1 Specifications of ASIC Image Sensor

Table 3.1 shows the specifications of the final image sensor chip.

Table 3.1: Design specifications of the ASIC image sensor

Specification	**Value**	**Unit**
Process Technology	0.18 um XFAB XH018	
Package	JLCC84	
Chip Size	10 mm^2	
Resolution	16 x 16	
Operating Voltage	3.3 and 1.8	V
Pixel Size	6.52 x 6.52	μm
Pixel Fill Rate	46.58 %	
Output Signal	Digital	
Control and Timing	External	
Frame Rate	15	Hz
ADC	On-chip	

3.2 Analog-to-Digital Converter (ADC)

The ADC is one of the most essential components in a CIS, as more or less all modern image sensor devices will perform some data processing or storage of data in the digital domain. As the CIS resolution is ever-increasing, more demands are put on the ADC designer. Early CIS utilized only one ADC for readout, but the increasing resolution resulted in a need for higher readout speed. Alternatives involve moving the A/D conversion, resulting in column-level or pixel-level ADCs. The former is oftentimes associated with parallel readout, in which it is possible to digitize a full row of pixels simultaneously.

There are numerous ADC designs available, and within the CIS world; the single-slope (ramp), cyclic, SAR, and delta-sigma ADCs are the most commonly used topologies.

For this project, it is the aim to choose an architecture that has already been tested and proven. The single-slope ADC was decided to be the best option, and the design process is described in the next chapter.

3.2.1 PCB

A custom 4-layer PCB was designed and fabricated to conduct measurements on the chip prototype. The PCB is comprised of several blocks, including digital logic provided by an off-board FPGA, a DAC controlled by the FPGA, and numerous potentiometers providing analog bias signals.

A 4-layer PCB was not strictly necessary, but anything less than that would make the routing complicated, considering the large amount of signals involved. In addition, the CIS has eigth separate power domains (four to the chip core, and four supplying the ESD rings), so an entire PCB layer dedicated to the power supplies is convenient for the routing and floor planning.

3.2.2 Lens and Lens Mount

The lens is a critical part of obtaining an image, and thus, an appropriate lens mount is required to keep the lens in front/on top of the sensor. In the past, several iterations have been made to create CIS' with varying specifications, and the same lens has been used throughout those years. This is a Raspberry Pi camera lens with the specifications outlined in table 3.2.

A lens mount was designed and 3D printed by fellow master's student, Jens Bulukin for his project using the same chip package. The same mount was used for this project, however with slight modifications.

Table 3.2: Lens parameters

Specification	**Value**
Model	PT361060M3MP12
Mount	CS-mount
Focal Length	6 mm
Aperture	F1.2
M.O.D	0.2 m

3.2.3 All Programmable Soc

A SoC with an FPGA module is needed for controlling the timing and sequencing of the CIS and to read out the resulting data. The PYNQ-Z1 board was selected to perform this task, and features Xilinx' Zync SoC that integrates an ARM Cortex-A9 processor with a traditional FPGA into a single integrated circuit. This architecture is what Xilinx refers to as an All Programmable System-on-Chip (APSoC).

The board gives access to the PYNQ (Python Productivity for Zynq) system, which, for many applications, makes it easier to design embedded systems by the means of hardware libraries (called overlays) to create programmable logic. As the name suggests, the programming can be done in Python, which raises the level of programming abstraction and eliminates the need to delve into HDL. This feature can be accessed by simply connecting to the to board (e.g. by SSH) and using the browser that incorporates the open-source Jupyter notebook infrastructure in order to run an Interactive Python kernel.

For someone who's not an FPGA expert, the PYNQ boards are an attractive choice, as it could reduce the development time significantly.

Chapter 4

Design and Implementation

This chapter describes the ASIC design, component by component - including the layout and simulation results. The first part details the pixel array and the design of the pixel itself. The subsequent designs will be described in succeeding order according to the signal chain; from pixel level design, to the readout and digitization levels consisting of clamper, ADC and decoders.

4.1 ASIC Technology

In the CMOS world, there are a wide range of technologies and processes to choose from. Some years prior, X-FAB Silicon Foundries introduced XS018, the first specialised 0.18 μm CMOS process for fast and large image sensors. This relatively new process included 4-transistor pixel cells with pinned photo diodes that offered lower noise and dark current. It also included transistors with low-threshold voltages for usage in the SF, RST, or SEL devices, as well as a low-noise buried n-channel transistor. These devices offer diminished image lag (arising from incomplete charge transfer from PD to FD) and increasing transfer speed. XS018 sounded promising, and this process was therefore chosen for the ASIC design.

Due to unforseen circumstances, X-FAB canceled the XS018 chip fabrication for the following Autumn delivery. Resultantly, the ASIC was fabricated in X-FAB's XH018 process instead, and the specialized image sensor components delivered by the XS018 process was replaced with standard components. On account of this, there will be two versions of some designs: one made in the XS018 process, and the other one in the XH018 process.

4.2 Basic Configuration

The basic configuration of the ASIC is divided into two large parts: the analog and digital domains. The analog domain contains the pixel array, clamper circuit, and comparator circuit. The rest falls into the digital domain, consisting of decoders and counters.

There are four domains of core supplies on the ASIC, as listed in table 4.1.

Table 4.1: Power domains of the ASIC

Type	**Voltage [V]**
Analog	3.3
Digital	3.3
Analog	1.8
Digital	1.8

Generally speaking, most of the analog circuitry on the chip is supplied with 3.3 V, while the digital circuitry is supplied with 1.8 V. A higher voltage supply for the analog parts will provide a wider linear working range which improves the SNR, output swing, and gain of certain sub-blocks.

On the other hand, digital circuits can function with a lower supply because it only needs the sufficient voltage to differentiate between binary states. With a lower power supply, the ASIC consumes less power and the transistors will provide faster switching.

4.3 Source Follower Design

The source follower is found in every CIS, owing to its small size and simple implementation. As always, these benefits come with trade-offs. The threshold voltage can easily vary with temperature, and sufficient linearity might be hard to achieve. As a result, the linearity sets a limit to the output swing.

In this project, only a few select parameters are optimized for: the output swing, gain and linearity.

The output swing should be between 0.5 V and 1 V. A swing larger than 1 V could lead to charge going back into the pixel, while too small of a swing results in low dynamic range. Therefore, settling with 0.7-0.8 V output swing is an acceptable compromise.

Another performance parameter is the gain. An ideal source follower has unity gain, but in reality, it is limited by the body effect and output resistance. In practice, due to the aforementioned limitations, unity gain is unachievable. Settling with $A_v \geq 0.8$ is sufficient.

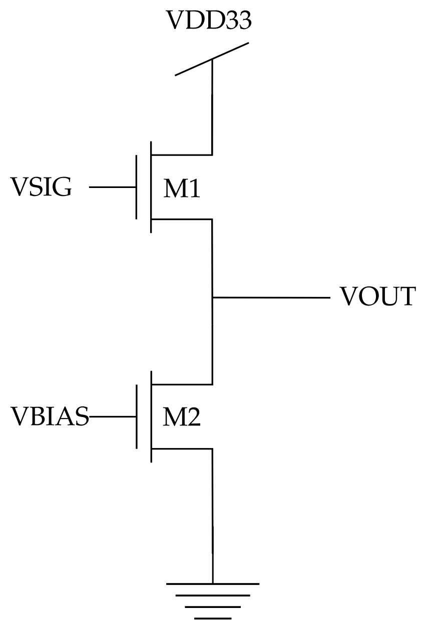


Figure 4.1: Schematic of source follower

The conversion from photon to electron is inherently a linear process. However, the components that must process the signal current can be nonlinear. The source follower is known to possess such nonlinearity, which can be destructive to the image capturing process. When designing the source follower, the linearity error has been taken into account. According to [9], the linearity error is defined as a deviation of a measurement characteristic from the ideal.

In state-of-the-art CIS, a linearity error of less than $\frac{1}{2^{10}} = \frac{1}{1024}$ is ideal if it includes a 10-bit ADC. On the other hand, this is very difficult to achieve. Hence, it was decided that a linearity error of less than 1 % of the output range would be sufficient. Since the rail-to-rail output range is about 0.8 V, the linearity error must be less than 0.8 mV.

The linearity error is given by the following equation:

$$y_{ideal} - y = m(t_1 - t_0) + y_0$$

4.3.1 Design of Source Follower in XS018 Process

The use of CDS does not fully eliminate 1/f noise which is significant under low-light conditions, and no adequate techniques are currently known to reduce it [15]. In this regard, a source follower featuring a buried-channel NMOS is commonly used, and such a device comes with the XS018 process. The buried-channel transistor pushes the highest potential in the channel away from the Si-SiO$_2$ surface (where 1/f noise is known to be caused by lattice defects), and minimizing the potential trapping of carriers. This transistor also has a negative threshold voltage which is known to significantly improve the pixel output swing.

When designing the source-follower, linearity characteristics were analyzed to improve the photoconversion process. To this end, various transistor sizes were swept in a DC analysis. Using the aforementioned equation, the linearity error was calculated using values found by simulation as follows:

$$\begin{aligned} y_{ideal} - y &= m(t_1 - t_0) + y_0 \\ 2.225 - y &= (895.2e^{-3}) \cdot 1 + 1.338 \\ y &= 2.705 - ((895.2e^{-3}) \cdot 1 + 1.338) = 7.775e^{-3} \end{aligned}$$

The gain was found by taking the maximum of the derivative of the output in a DC analysis, and was found to be:

$$\begin{aligned} A_{DC} &= 20log10\left(max\left(\frac{dV_{out}}{dV_{in}}\right)\right) \\ &= -0.962\text{dB} \end{aligned}$$

Abiding by the aforementioned requirements, the sizes in table 4.2 were chosen for the transistors making up the source follower.

Table 4.2: Transistor sizes for the source follower circuit with buried-channel SF transistor using XS018 technology

Transistor	**Width** $[\mu m]$	**Length** $[\mu m]$
M1	0.22	1.4
M2	3	3

4.3.2 Design of Source Follower in XH018 Process

In the final design using XFAB's XH018 process, the values in 4.3 were chosen. Due to the very prompt, last minute decision to port the existing design from XS018 to XH018, re-doing the simulations was not a priority.

Table 4.3: Transistor sizes for the source follower circuit with regular N-MOSFET SF transistor using XH018 technology

Transistor	**Width** $[\mu m]$	**Length** $[\mu m]$
M1	1u	1u
M2	3u	3u

4.4 Pixel Design

Since 4T pixels have become a norm, this configuration was chosen for this project. In contrary to the 3T pixel configuration, there is one extra transistor occupying the pixel area. The disadvantage is lower fill rate. For this project, chip area was not an issue, and the extra benefit of eliminating kTC-noise was heartily welcomed.

Another important reason for choosing the 4T pixel configuration was the fact that X-FAB's XS018 provided CIS specific transistors for in-pixel use, as outlined previously. One of the goals that were kept in mind was the testing of these new components.

The simulated result can be seen in figure 4.3. At 0 μs, TX, RST, and SEL starts in an OFF state. Pulsing TX and RST simultaneously depletes the photodiode PD by removing any photon-charge from the previous capture. At this point, a photogenerated charge starts being collected in PD. SEL is then asserted to enable the pixel signal to propagate to the column bus, after which RST will be pulsed to sample the FD reset level. Subsequently, TX is pulsed, and the photo-electrons on the PD are transferred to FD, resulting in a pixel signal voltage.

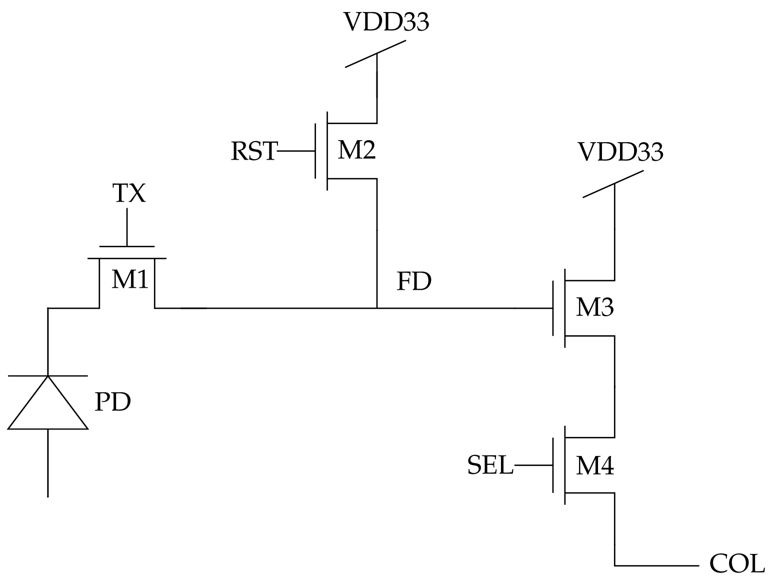


Figure 4.2: Schematic diagram of the 4T Pixel

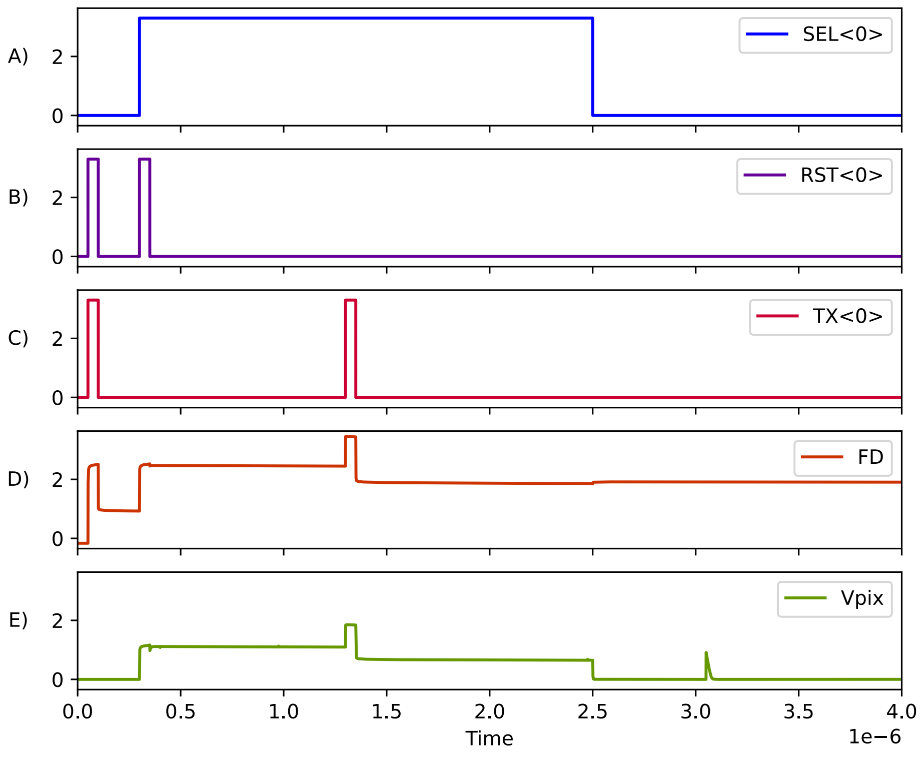


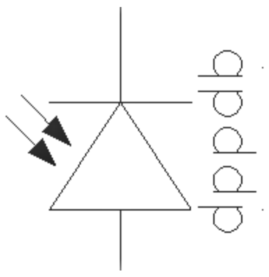
Figure 4.3: Timing diagram of the 4T pixel with foundry photodiode.

4.4.1 Design of Pixel in XS018 Process

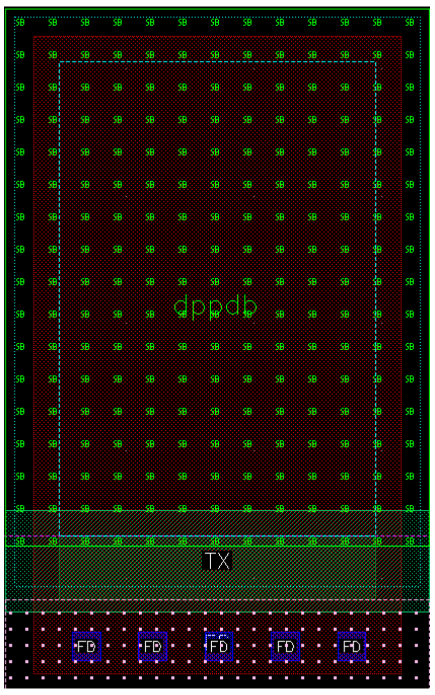
The pixel design was the crux to choosing the XS018 process. This process provides pinned photodiodes, low threshold NMOS, and buried channel NMOS which would be interesting to fabricate and eventually test and characterize.

XS018 Pinned Photodiode

In principle, every diode fabricated in CMOS process flow can be applied as a photodiode with large spectral sensitivity variations. The pinned photodiodes provided by X-FAB are optimized for maximum sensitivity over wide spectral range. The original pixel design was done in the XS018 process, using the photodiode depicted in figure 4.4, where figure a) is the schematic symbol, whilst figure b) shows its layout design.



(a)



(b)

Figure 4.4: (a) Schematic symbol of foundry photodiode. (b) Layout design.

The resulting transistor sizes can be seen in table 4.4.

Table 4.4: Transistor sizes for the 4T pixel in XS018 process

Transistor	**Type**	**Width [µm]**	**Length [µm]**
M1	ne3tx	2.885	0.8
M2	NMOS	0.22	0.35
M3	NMOS	1	0.5
M4	NMOS	0.22	0.35

4.4.2 Design of Pixel in XH018 Process

In the end, the entire design was ported to the XH018 process. The photodiode, buried channel transistor, and low threshold NMOS found in the XS018 process were scrapped and replaced by a custom-made photodiode and two regular NMOSFETs.

The regular diodes provided through the XH018 process were way too large in size to fit into the pixel layout, and would require a huge revamp of the practical floorplanning. This is the reason a custom diode was made.

As a result, no diode was implemented in the pixel schematic, but a diode consisting of mainly an NWELL layer and substrate connection was implemented in the layout. Of course, without a pinned photodiode, full charge depletion is not possible, and the performance is highly reduced.

The final transistor sizes are shown in table 4.5.

Table 4.5: Transistor sizes for the 4T pixel in XH018 process

Transistor	**Type**	**Width [µm]**	**Length [µm]**
M1	NMOS	0.22	0.35
M2	NMOS	0.22	0.35
M3	NMOS	1	1
M4	NMOS	0.22	0.35

4.4.3 Pixel Layout

Figure 4.5 shows the final pixel layout with the custom diode. Note that some layers are removed for higher visibility. A photodiode is created by placing an NWELL on top of a diffusion layer.

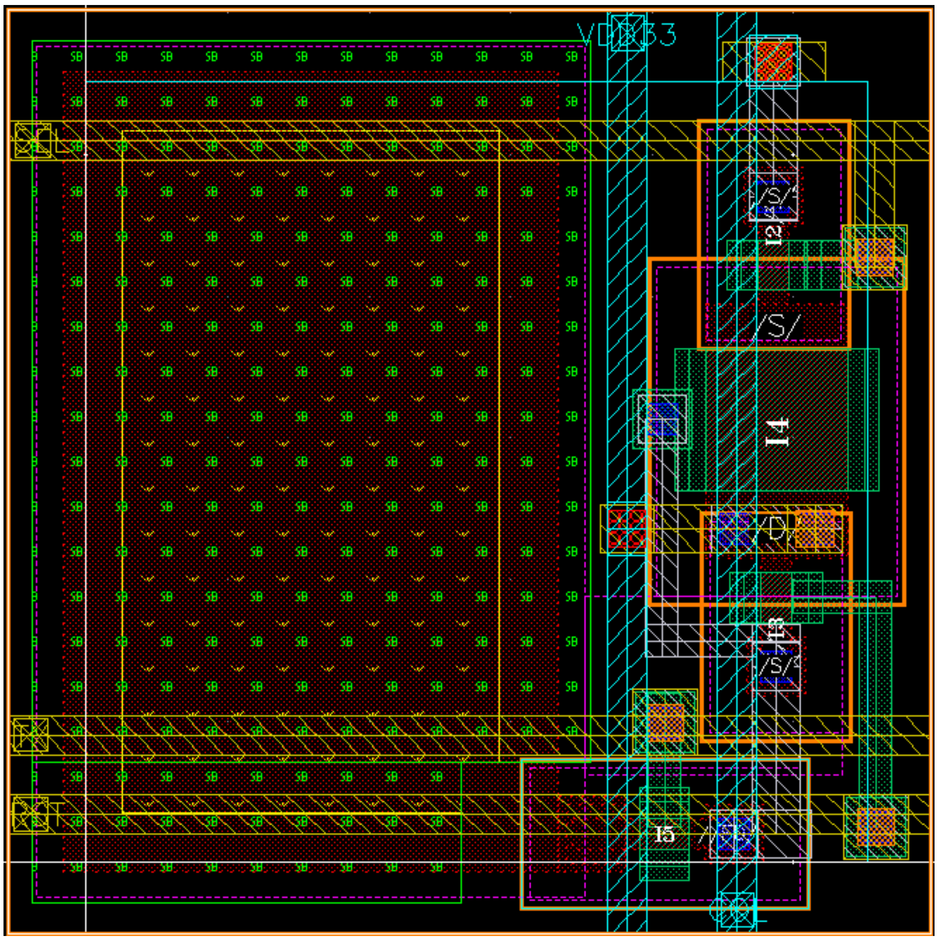


Figure 4.5: Layout of 4T pixel with some layers removed for improved visibility

4.5 Pixel Array Design

To simplify the ASIC design process as well as the signal controlling process, the pixel array was kept to the bare minimum. Conventional CIS consists of millions of pixels, and for this project, the number of pixels was not the imperative.

The pixel count was originally planned to be 320 x 240 pixels, or QVGA resolution, as the original project plan was to reuse as much as possible from previous master's projects over the years. However, as they were using a different ASIC technology and had a different goal for their project, the reuse of their design proved to be more complicated than useful. In the end, with the whole project scope in mind, the pixel array was decided to be comprised of 16 x 16 pixels.

Additionally, in conventional CIS there are, amongst others things, dummy pixels and optical black pixels included along the border of the array. The former ensures a uniform environment for the outermost pixels by minimizing the impact of surrounding circuits, and the latter prevents light from entering the pixels. However, optimization is not a priority, and therefore, such implementation was decided against.

Figure 4.6 and 4.7 respectively show a snippet of the pixel array and a the full sized schematic of the pixel array.

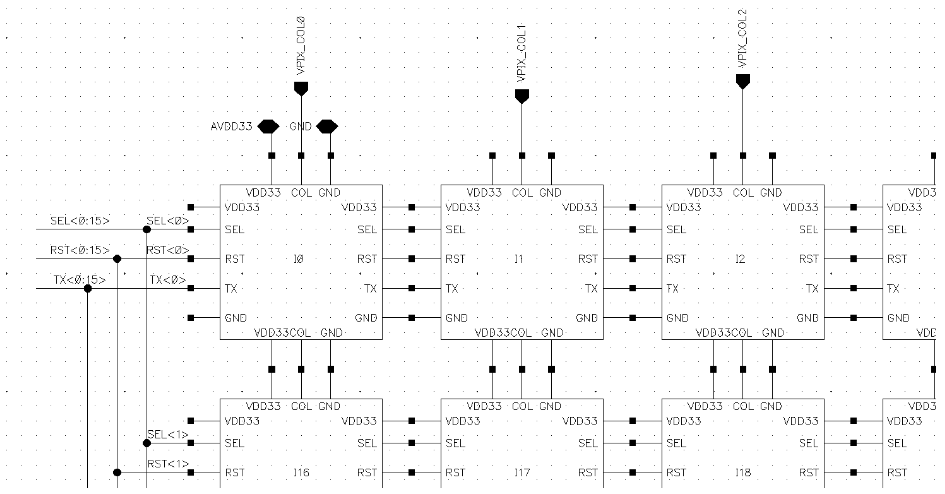


Figure 4.6: Schematic diagram of a selected area of the pixel array.

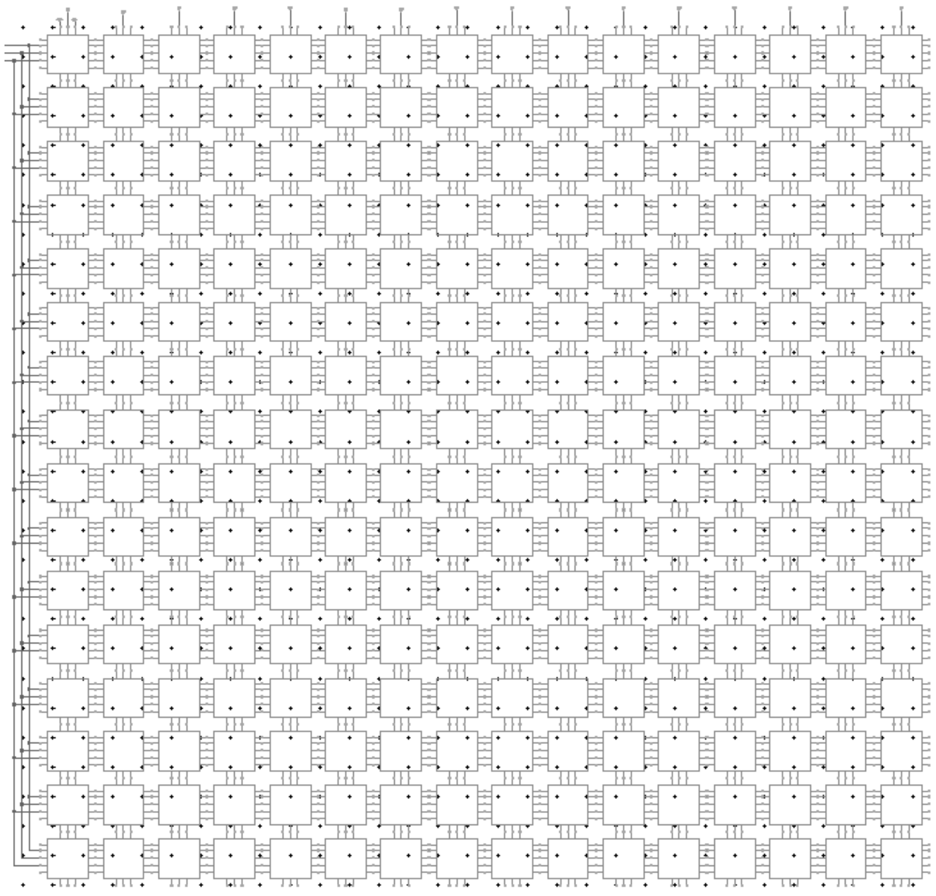


Figure 4.7: Full schematic diagram of the pixel array comprised of 16 x 16 pixels.

4.5.1 Layout of Pixel Array

The layout of the pixel array is shown below, and features 16 x 16 pixels. The outermost border is a guard ring that isolates the array against noise by preventing spurious electrons and holes from entering the array.

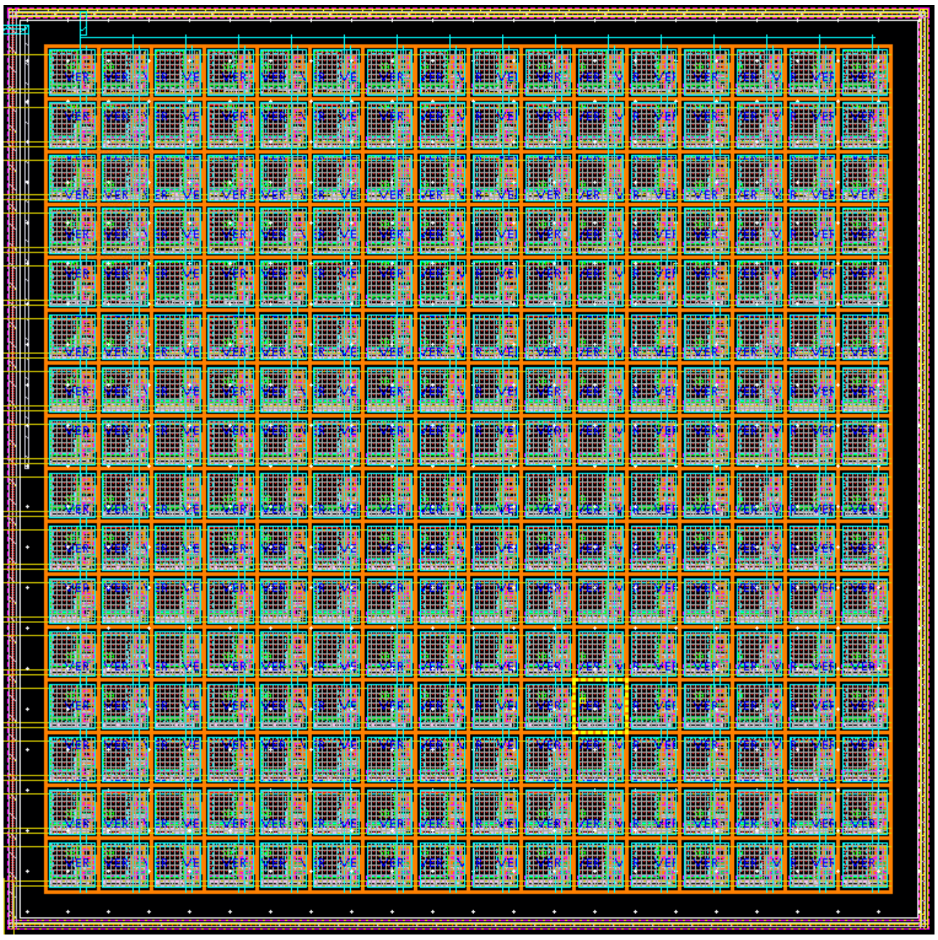


Figure 4.8: Pixel array layout

4.6 Analog Preprocessors

As demonstrated earlier, the pixel output signal is analog, which must be processed by analog circuits. Such circuits normally include sample-and-hold, color separation, automatic gain control (AGC), level clamp, tone adjustment, and much more prior to being converted to a digital signal [9]. This project is not focused on signal processing, so naturally, most of the possible circuits are omitted.

Sample-and-hold circuits are normally a must, as it performs the necessary operations of sampling the reset and pixel signals. Different topologies can be used for improving consistency of the pixel to pixel output and reducing fixed pattern noise, like designs found in [10].

With the 4T pixel topology, the switching is done by the in-pixel transfer gate where opening the gate would lead to changes in the output node. For this reason, only a level clamp is incorporated as an analog preprocessor as depicted in figure 4.9. The level clamper follows a switch-cap configuration, where the goal is to shift the input waveform either above or below a DC reference level without altering the shape of the waveform.

Furthermore, this structure eliminates reset voltage variations and outputs the voltage difference between the reset and the pixel levels as they change, effectively achieving CDS operation.

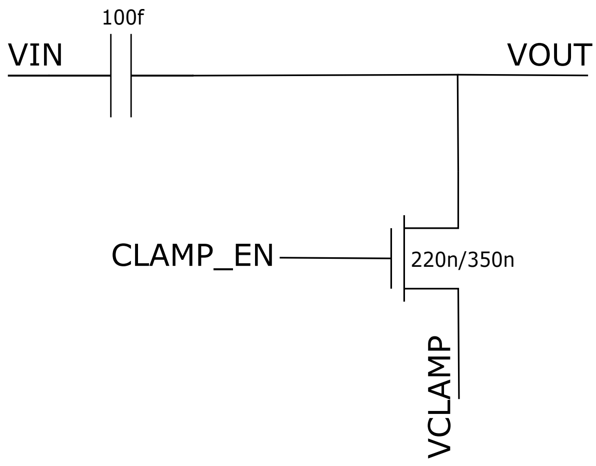


Figure 4.9: Schematic of clamper circuit

The timing digram in figure 4.10 shows that as V_{in} goes from 3 V to approximately 2.5 V, V_{out} gets clamped to 1.7 V and 1.2 V respectively. The clamper detects the difference between the reset signal of a unit pixel and the data signal by the optical input, where the reset signal is clamped to a predetermined voltage level V_{offset}.

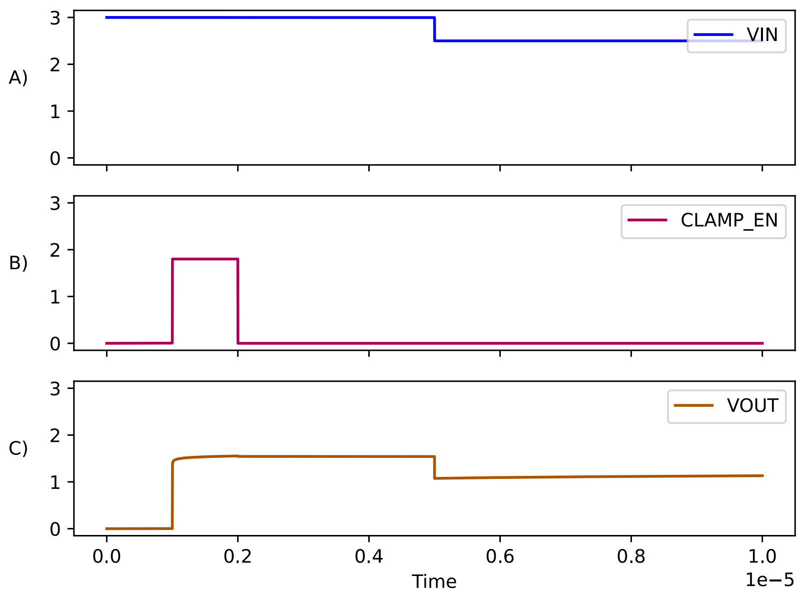


Figure 4.10: Timing diagram of clamper circuit, showing the input voltage from a pixel V_{IN}, the clamp enable switch CLAMP_EN being asserted, and the output voltage V_{OUT} being clamped to the clamping voltage of 1.7 V.

4.6.1 Level Clamper Layout

The clamper layout is shown below, where the capacitor is occupying most of the space, and the entire layout width is the pixel width.

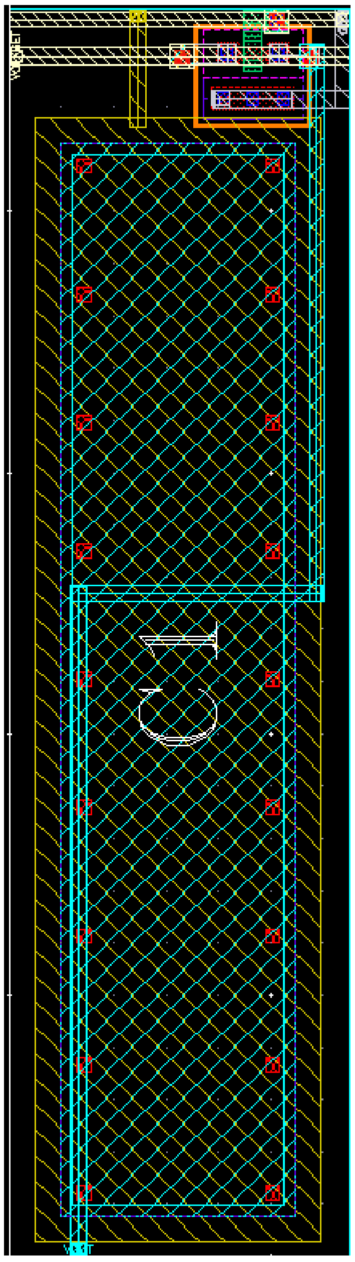


Figure 4.11: Layout of clamper circuit

4.7 Row Decoder Design

The chosen row decoder was originally designed by a previous master's student, Mathias H. F. Wilhelmsen, found in [16]. Literature reports designs using digital peripheries consisting of column and row shift registers or so-called scanners with a clock trigger [14]. However, knowing that the chosen decoder was tested and verified previously, makes it a safer option to implement. Furthermore, scanners are simpler in design, but harder to control by external means. A decoder gives more freedom in accessing specific rows or columns simply by providing a bit address externally, which can be useful in the testing process.

The row decoder is implemented with Gray coding which prevents multibit count error and consumes less power compared to the natural binary code [11]. With Gray Code, only one bit changes state from one position to another. The row decoder is formed by alternating layers of NAND and NOR logic. The basic components are 1-2 decoders, which are implemented in four different ways; by NAND and NOR gates in both "normal" configuration, as well as an equivalent mirrored configuration. The additional mirrored configuration enables gray encoding. The decoders are depicted in the figures 4.12a - 4.15b.

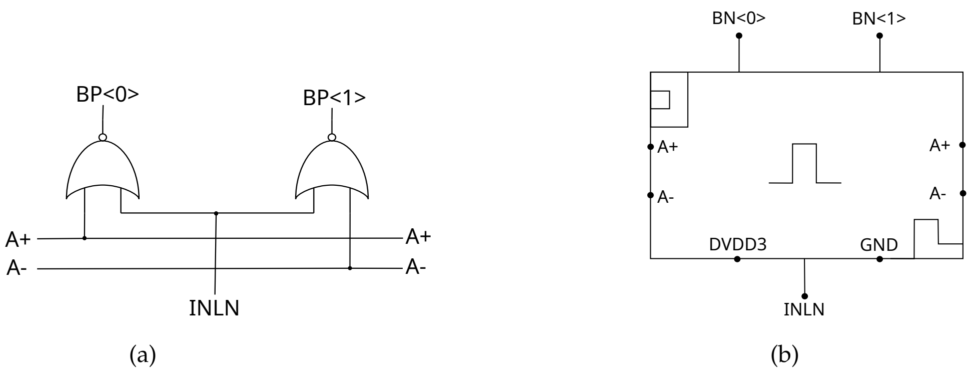


Figure 4.12: (a) Schematic of 1-2 demux implemented with NOR gates and (b) Symbol equivalent.

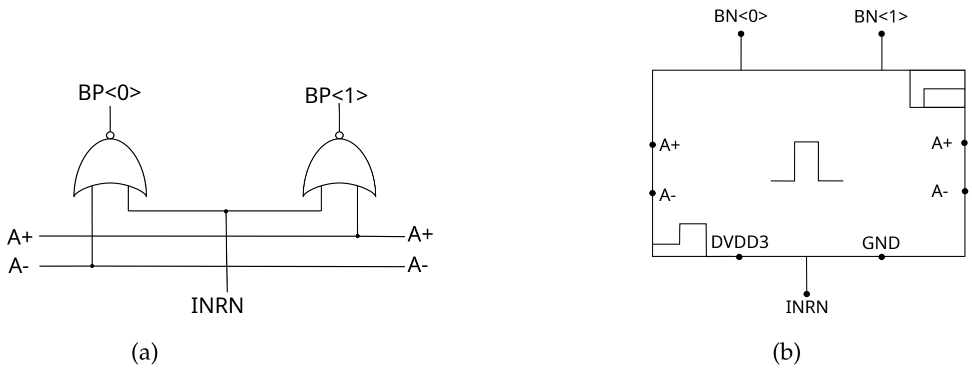


Figure 4.13: (a) Mirrored variant of a 1-2 demux implemented with NOR gates and (b) Symbol equivalent.

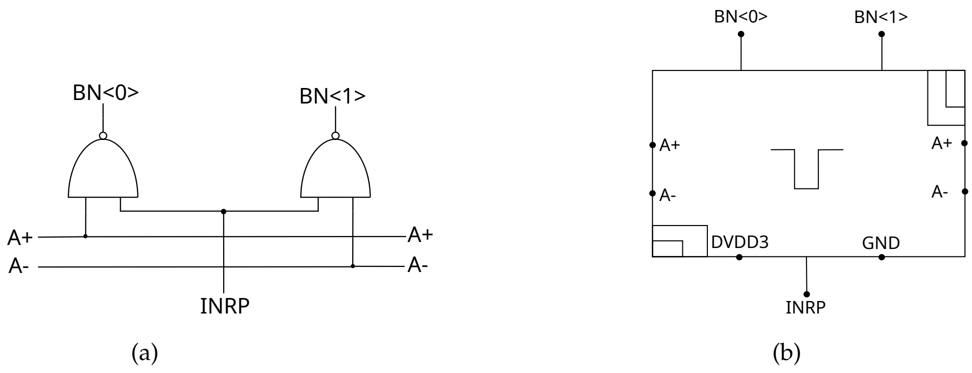


Figure 4.14: (a) Schematic of 1-2 demux implemented with NAND gates and (b) Symbol equivalent.

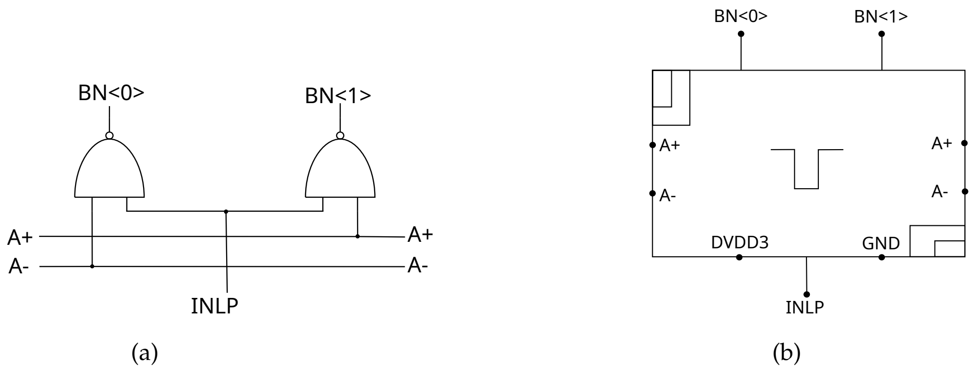


Figure 4.15: (a) Mirrored variant of a 1-2 demux implemented with NAND gates and (b) Symbol equivalent.

Since the pixel array contains 16 rows, a 4-16 decoder is the final product, created from the 1-2 decoders. An address of four bits are needed to select between the 16 rows. In addition to row selection as the output, there are additional control blocks to enable either the transfer gates, reset gates, or select gates of the selected row. Figure 4.16 shows the final 4-16 decoder.

The build of the decoder is more or less identical to [16], but generally simpler due to less rows that need to be addressed. However, the original design did not include a transfer gate selection in the control block. The circuit is designed hierachically, forming a symmetrical binary tree that allows a symmetrical layout structure. The gates are from the foundry.

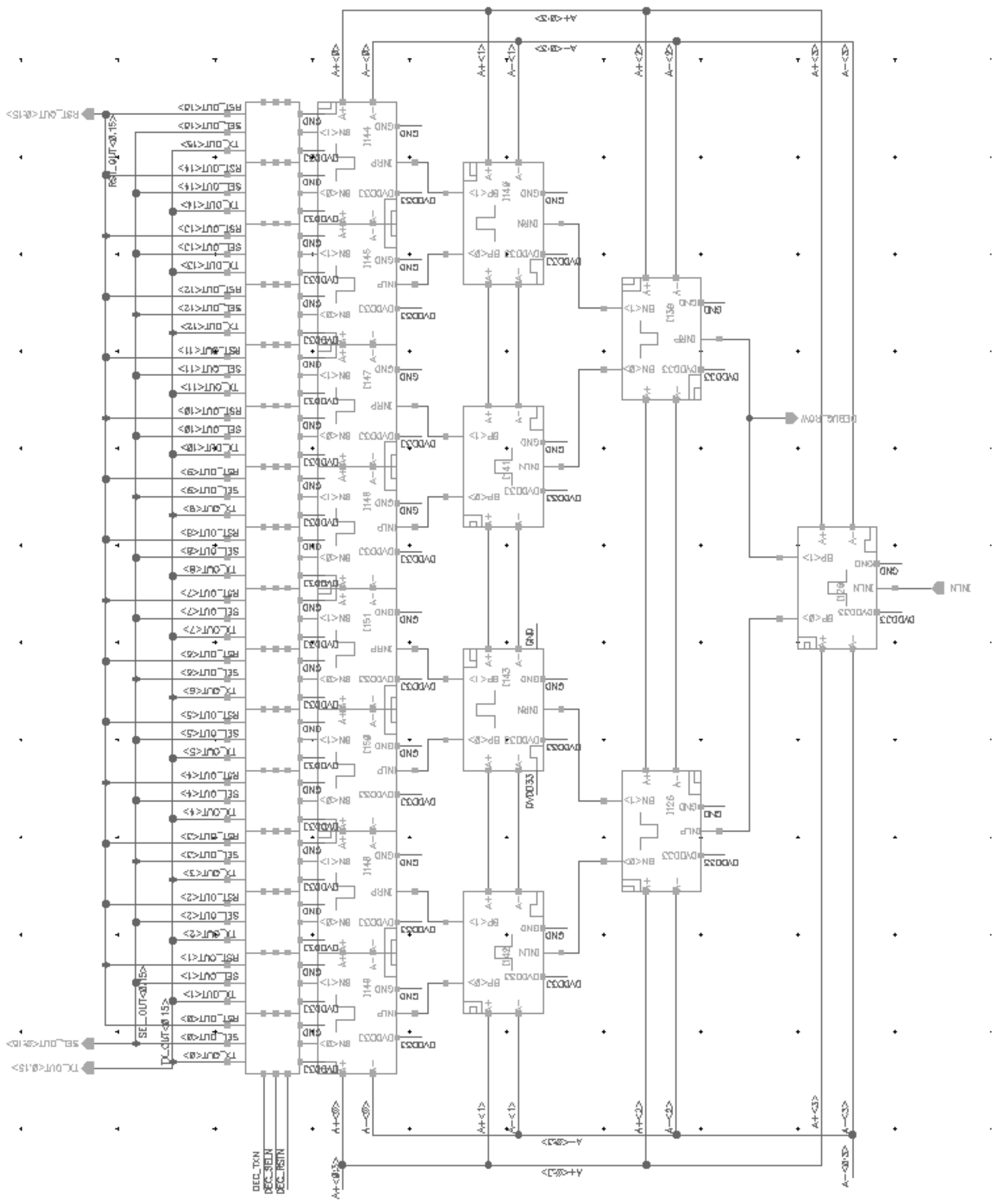


Figure 4.16: Schematic of 4-16 row decoder with additional control blocks for enabling TX, RST, and SEL gates.

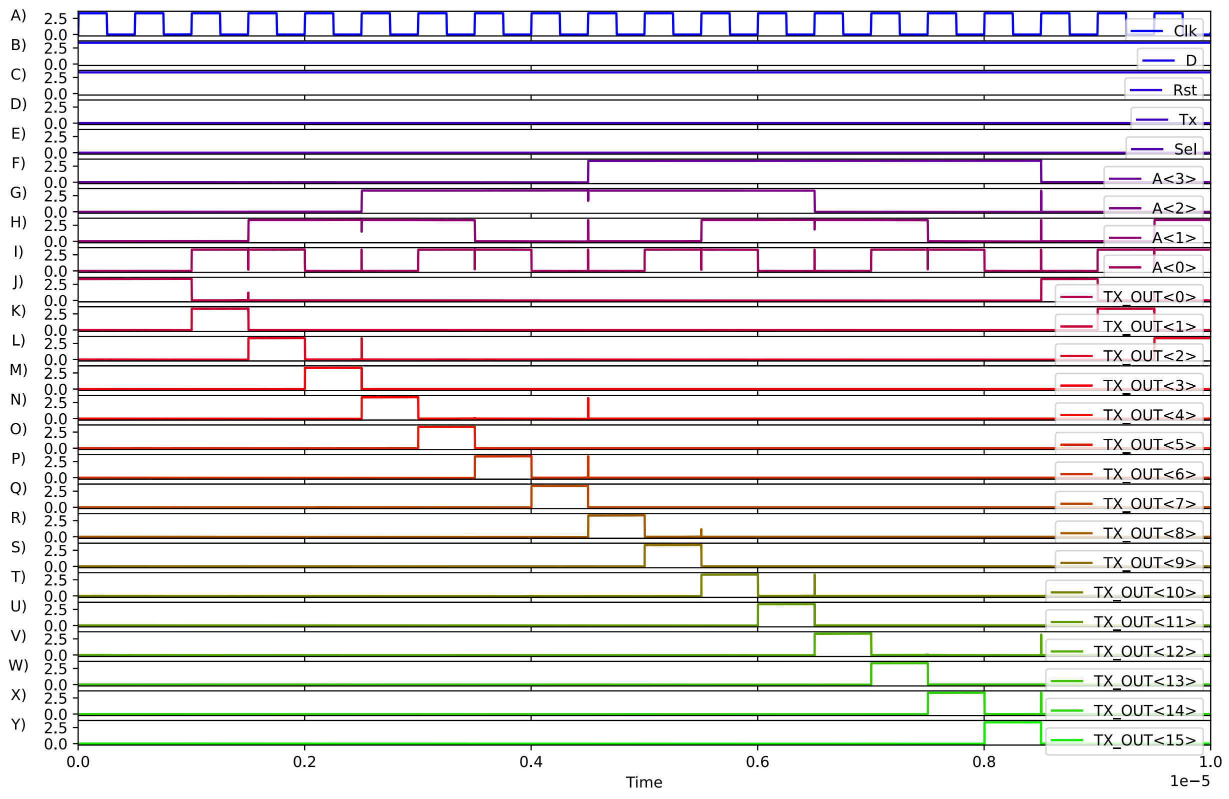


Figure 4.17: Timing diagram of 4-16 decoder.

4.7.1 Layout of 4-16 Row Decoder

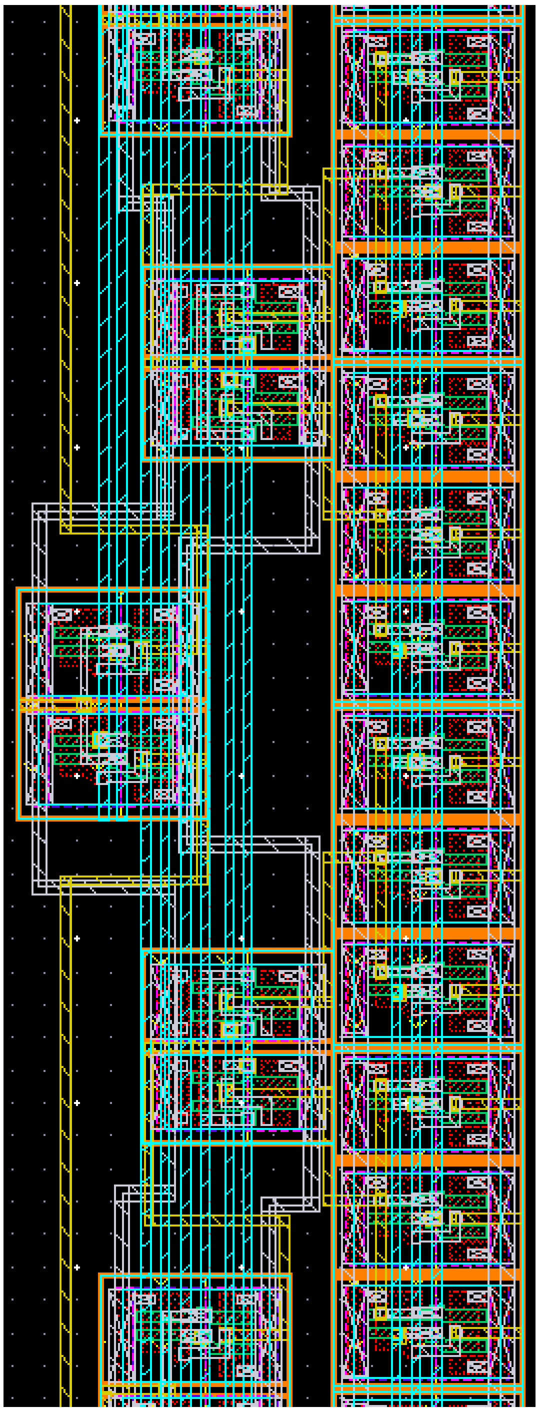


Figure 4.18: Small snippet of the layout of the 4-16 row decoder.

4.8 Column Decoder Design

A basic multiplexer was implemented in the column decoder, as seen in figure 4.19. The diagram shows a combinatorial logic with 16 input pins and 4 select lines with one output. Four 4-1 MUXs are used to obtain the 16 inputs, and the fifth MUX connected to their outputs multiplexes only one of the four prior MUX outputs at a time.

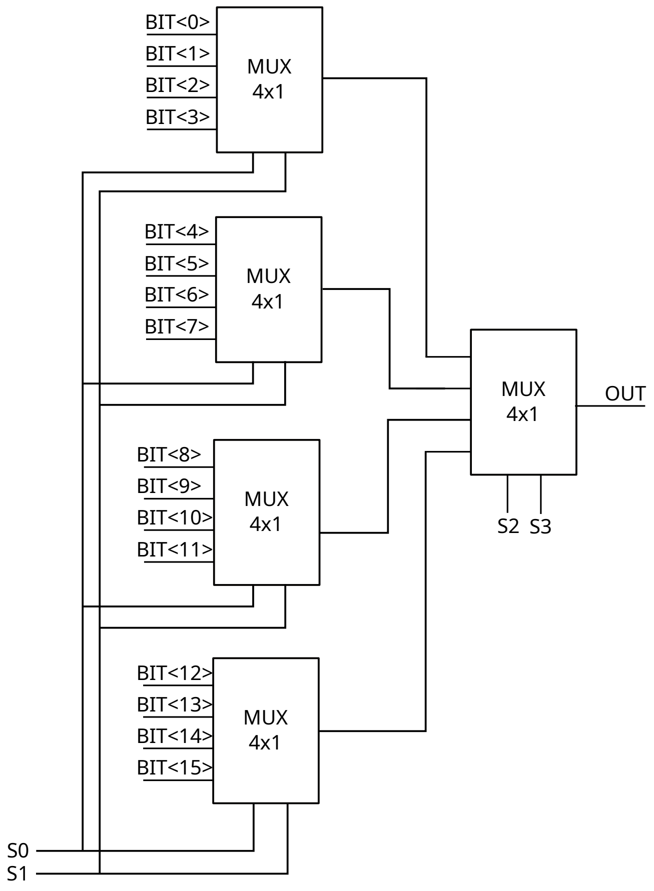


Figure 4.19: Schematic of column decoder

In the top layer, 16 units of 16-1 MUXs were implemented to multiplex between th 16 different columns in order to read out the digitized signal levels. Every MUX output is connected to one ADC outputting 16 binary numbers.

4.8.1 Column Decoder Layout

The column decoder layout is shown below, using digital components provided by the foundry. The layout width is identical to the pixel width.

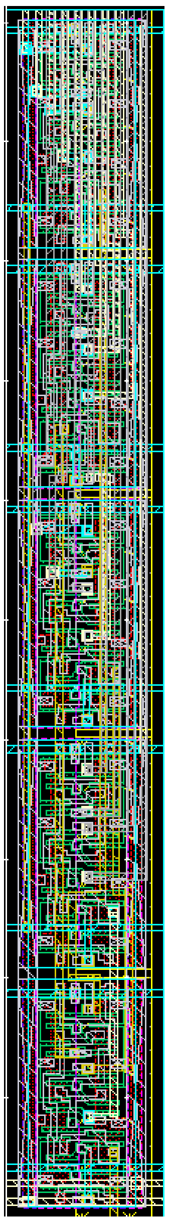


Figure 4.20: Layout of the column decoder

4.9 Single-Slope ADC Design

Figure 4.21 shows a system block of the SS-ADC consisting of an analog comparator and a digital counter with latching. There are in total 16 ADCs on the ASIC – one in each column below the pixel array.

The column comparators are driven by a ramp generator (DAC) and the pixel signal output. The column counters perform the A/D conversion by counting the number of clocks until the comparator output switches, and digital CDS is possible to do by alternating between up counting mode and down counting mode with a control signal.

The number of comparators in a typical CIS is usually very high - perhaps hundreds of thousands. As the SS-ADC is merely composed of a comparator and a counter, the main advantage is the minimal amount of analog components involved. The area consumption is small, and it is relatively easier to design and implement – especially compared the more complex architectures involving numerous more components. E.g. is the SAR ADC that requires both a comparator and a DAC, or the cyclic ADC requiring amplifier, comparator, and S/H circuitries.

Compared to architectures such as the cyclic ADC, it has very good linearity and accuracy. In contrary to the cyclic ADC that needs exactly 2x gain, a specific amount of gain is not needed for the SS-ADC.

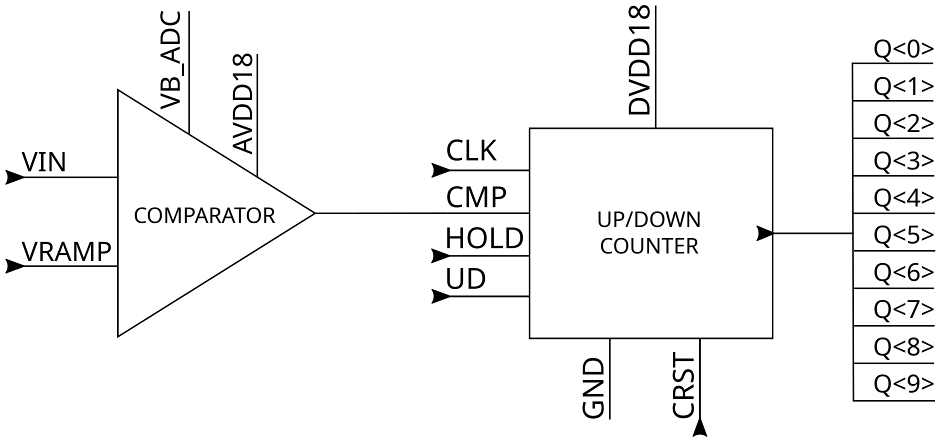


Figure 4.21: Block diagram of single-slope ADC

It goes without saying that minimizing the comparator size is desirable in order to reduce silicon chip area, as the column ADC should have the same width as a single pixel on the chip. A simpler architecture helps achieving this. Furthermore, the simplicity makes it easier to ensure uniformity between columns, since any difference in the columns lead to vertical stripes in the final image [13]. A central ramp generator can be shared amongst the ADCs, and therefore reduce the overall power consumption. Another important advantage of a central ramp generator is that it is easier to achieve higher uniformity between each ADC, since any non-ideal behaviors in the ramp generator will affect all ADCs in an equal manner.

There are of course, weaknesses with the SS-ADC as well. The main issue is that the ramp generator introduces noise to the images and the generator is known to consume a lot of power. Another weakness is that it has low conversion rate [12]. For this project however, power consumption and conversion rate are not the focal point as a 16 x 16 pixel array has lenient speed, power consumption and noise requirements.

Conventional image sensors possess dynamic ranges of around 60 dB [5], and as described in chapter 2, the dynamic range is determined by the ratio between the largest and smallest values that the ADC can measure. According to equation 2.8, 10 bits are required to achieve $\sim$ 60 dB in dynamic range.

The final specification of the ADC can be seen in table 4.6.

Table 4.6: Design specifications of the single-slope ADC

Specification	**Value**	**Unit**
Level	Column	
Resolution	10	bits
Width	6.52	µm

4.9.1 Continuous-Time Comparator Design

The classic continuous-time comparator with a differential pair was implemented in the analog part of the SS-ADC.

At a basic level, all circuits must somehow be operable within the intended voltage range, which is characterized by its ICMR. For comparators and general op-amps, static gain is of utmost important because it describes the minimum amount of input change necessary to make the output swing between two binary states. After all, the two binary states must be able to drive the digital circuit connected to the comparator's output. The offset voltage is also crucial and often problematic as it cannot be predicted, and varies randomly from circuit to circuit for a given design [2].

Speed (time response) and noise were of low priority due to the low pixel and pixel array specifications which already set a limit to the overall achievable image quality, and hence the ADC and comparator. In other words, aiming for high speed is not necessary when dealing with low pixel counts, and a low pixel count clearly limits the achievable image resolution. Of course, it is argueable whether one parameter should be prioritized over another, as for instance noise leads to uncertainty of the transistion region of the comparator, which again affects other parameters, creating a sort of cascading effect of non-idealities. However, following the good old trade-off principle, it was decided that only a few select parameters would be used as an anchor to the design itself, whilst most had to be omitted due to time restrictions.

Input Common Mode Range

Since the comparator input is differential, the input-common-mode range is important. ICMR is usually determined by the range where all transistors of the comparator remain in saturation [2]. Several simulations were done to ensure a decent ICMR, as seen in figure 4.23 and 4.24.

The former shows a transient plot where the triangular line represents the ramp, the solid lines represent the reference voltage, while the dotted lines are the comparator output swings. The comparator switches at the expected transistion points as the reference voltage intersects with the ramp, as seen in figure 4.23. It is important to note that these simulations are made using components

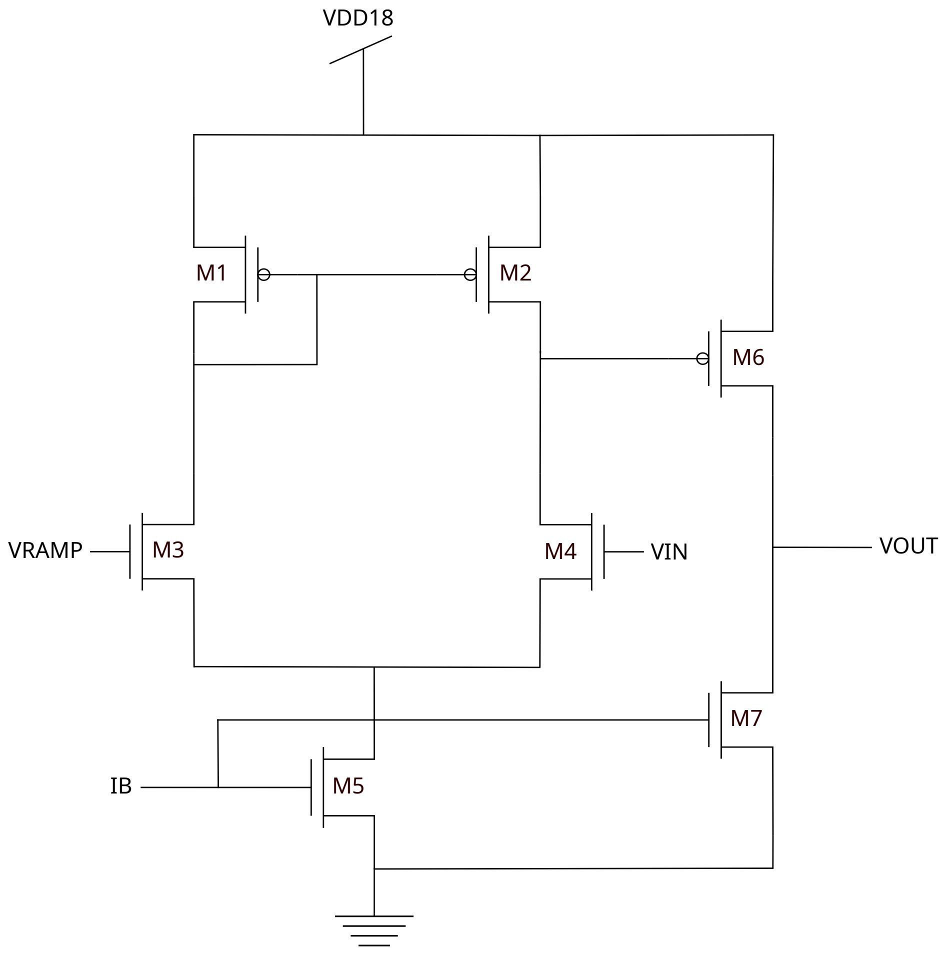


Figure 4.22: Schematic diagram of the continous-time comparator

of typical corner models, and no parasitic extractions were done. Hence, it is expected that the real product might deviate a lot from these simulations.

Figure 4.24 shows a DC analysis where the differential input pair is fed with V_{in}, which is swept across a range of values between 0.6 V and 1.8 V, and plotted against V_{out}. Evidently, the differential pairs are on and operative from about 0.8 V, and this is an indicator of the threshold voltage being at this voltage. It also seems that the linear range starts from 0.9 V input. Based on this knowledge, the expected input range should range from 0.9-1.6 V.

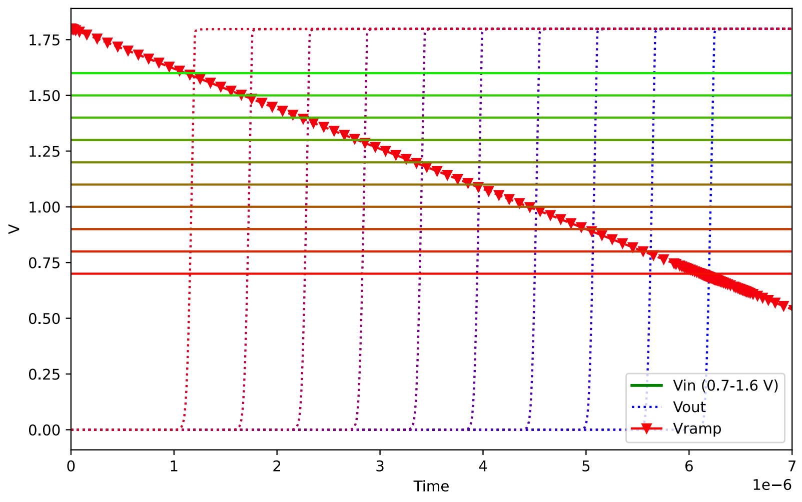


Figure 4.23: Transient analysis: V_{in} (solid lines) swept from 0.7-1.6 V, V_{ramp} (triangle markers), and V_{out} (dotted lines)

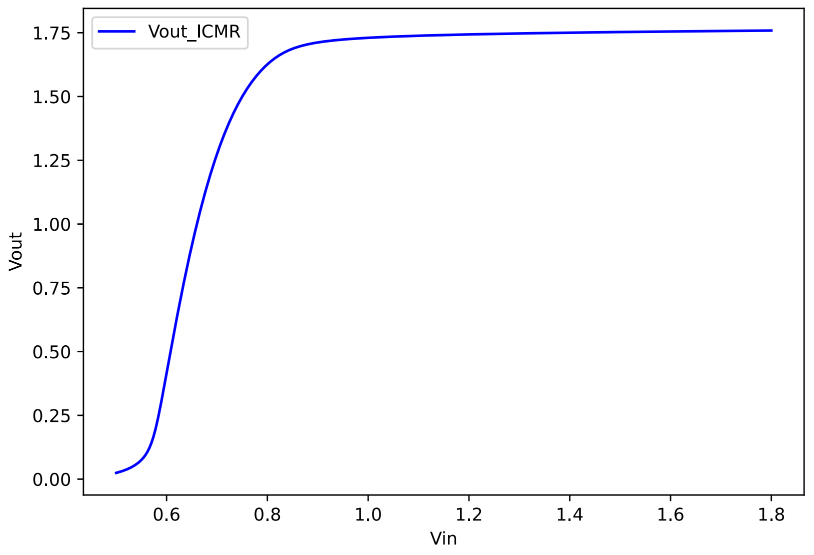


Figure 4.24: Input Common Mode Range simulation with a DC sweep

DC Gain

The DC characteristics of the comparator were found through simulations, and the gain is approximately 53 dB. The input node V_{ramp} was held at a constant, and the voltage on V_{in} was swept. This method of simulating the DC gain can be expressed in mathematical terms as follows:

$$\begin{aligned} A_{DC} &= 20\text{log10}\left(max\left(\frac{dV_{out}}{dV_{in}} \right) \right) \\ &= 53.2 \text{ dB} \end{aligned}$$

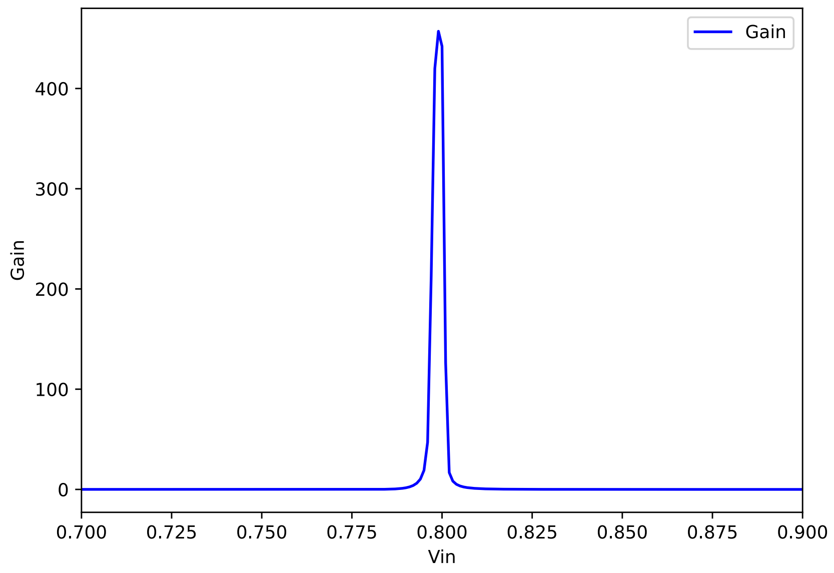


Figure 4.25: Simulation of the comparator DC gain

Input Offset

Figure 4.26 is the result of a simulation where V_{in} was swept from 0.7 V to 1.6 V with steps of 0.1. The plot shows V_{out} as a function of $V_{in} - V_{ref}$. Under ideal circumstances, every V_{out} graph should switch from a low state to a high state at $V_{in} - V_{ref} = 0$, but it is clear that this is not the case. As a matter of fact, every V_{out} that corresponds to a $V_{in} - V_{ref}$ step switches state with certain offsets. The lower the input voltage, the higher the offset.

In the figure, the leftmost green graph represents the output signal when $V_{in} = 1.6$V, whilst the rightmost blue graph is the resulting output when $V_{in} = 0.7$V. Hence, it is possible to conclude that there is a non-linear input offset, and that the comparator shows a worse performance at lower input voltages.

This behavior is expected when dealing with a differential pair consisting of N-MOSFETs, as there is a certain threshold voltage required to turn on the transistors.

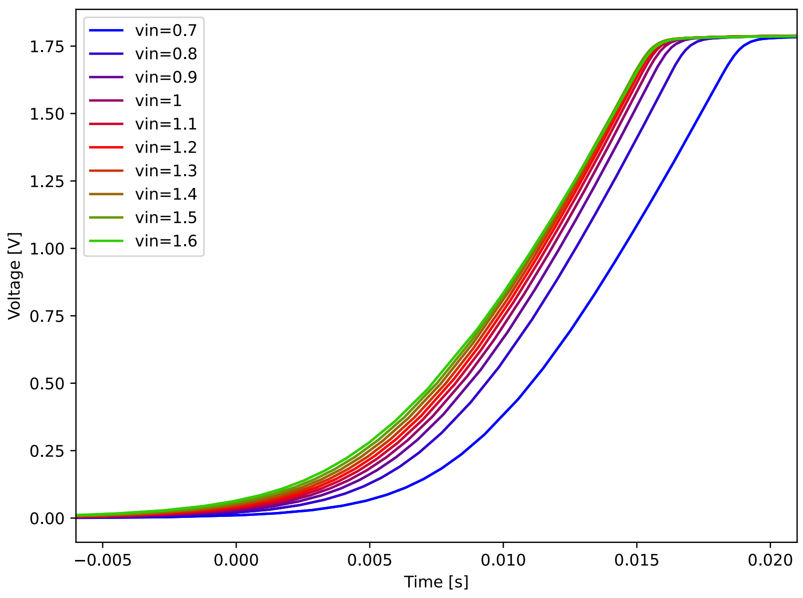


Figure 4.26: Simulation of the comparator input offset, $V_{in} - V_{ref}$ vs. V_{out} (transient analysis).

This non-linear characteristic can be further inspected in figure 4.27, where V_{offset} was plotted against a range of V_{in} values.

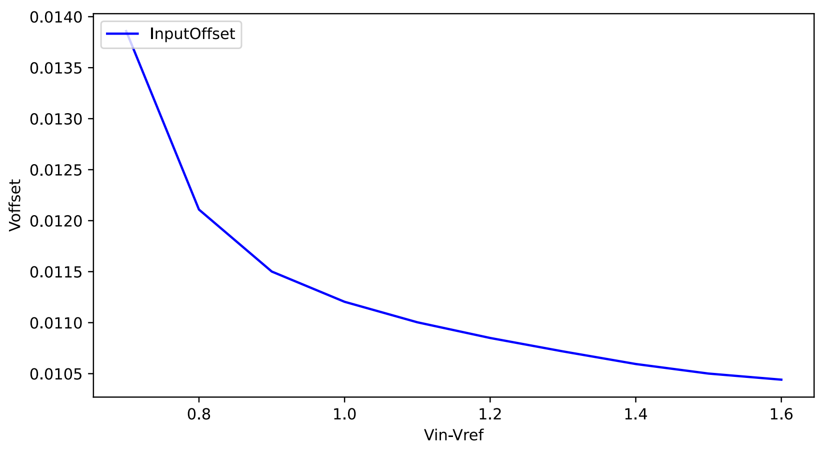


Figure 4.27: Simulation of the comparator input offset using transient analysis: $V_{in} - V_{ref}$ vs. V_{offset}.

Current Consumption

The current consumption was found through simulations by probing the current drawn by the power supply, and is depicted in figure 4.28. When the comparator is fully on and operating, the total current consumption is approximately 22 µA.

This figure is another indicator of the threshold voltage being at around 0.8 V, as it does not start consuming power before hitting this mark. The device parameter documented for this device states that the threshold voltage upper limit for short channel lengths should be at 0.73 V.

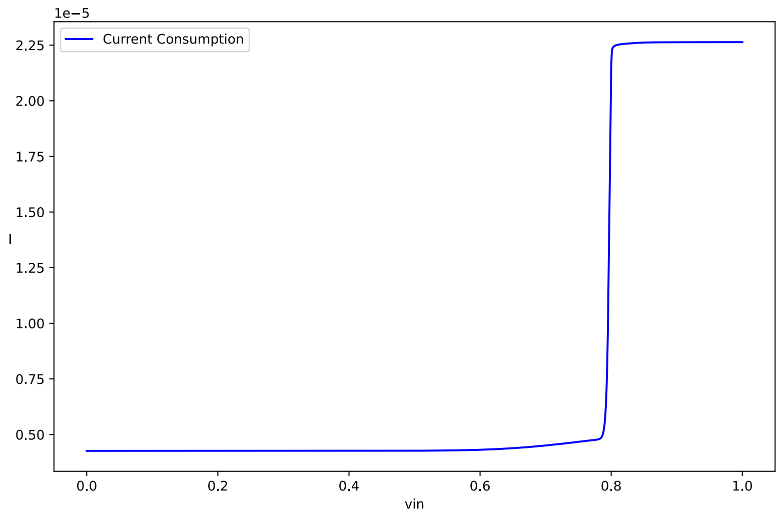


Figure 4.28: Simulation of the comparator current consumption

Continous-Time Comparator Design

Taking the aforementioned parameters into consideration, the final transistor sizes were chosen and shown table 4.7.

Table 4.7: Transistor sizes for the Continous-Time Comparator in XH018 process

Transistor	**Type**	**Width [µm]**	**Length [µm]**
M1	PMOS	0.5	0.18
M2	PMOS	0.5	0.18
M3	NMOS	0.5	0.18
M4	NMOS	0.5	0.18
M5	NMOS	5	0.18
M6	PMOS	4	0.18
M7	NMOS	2	0.18

4.9.2 Continuous-Time Comparator Layout

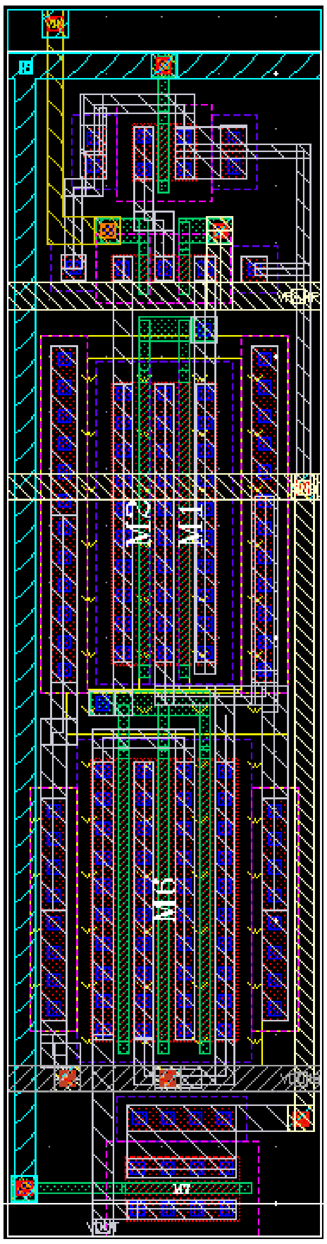


Figure 4.29: Layout of the Countinous-Time Comparator

4.9.3 Up/Down Counter Design

Figure 4.30 shows a traditional up/down counter based on the topology mentioned in [7]. This topology consists of DFFs and MUXs, where the former help realizing an edge triggered operation, whilst the latter provide the up/down counting operation.

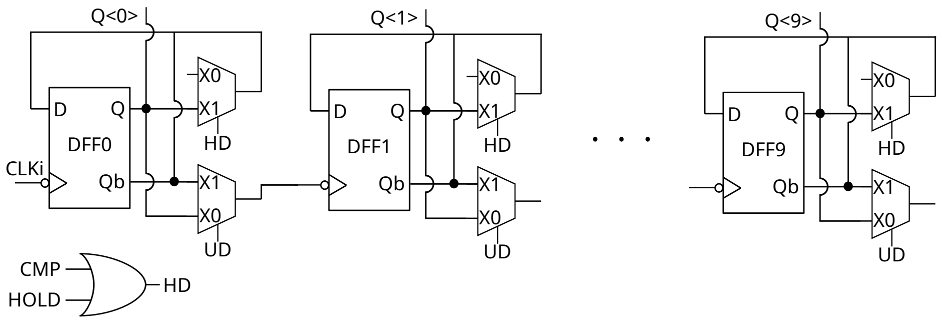


Figure 4.30: Schematic of a traditional up/down counter

The simulated waveforms are shown in figure 4.31, where the resulting outputs are denoted by Q<0> to Q<9> - a total of 10 output bits.

Upper most graph (A) depicts the clock signal. First and foremost, the UD signal specified whether the counter should count up (LOW) or down (HIGH). At first, down counting is desired. When CEN goes HIGH, the down counting is enabled.

As the input signal is being compared to the ramp signal during comparator operation, the CMP signal will be triggered once the signals cross. More specifically, once the ramp signal is lower than the pixel voltage, CMP goes HIGH. This marks the end of the down counting, and the resulting bits are stored in the DFFs. Subsequently, HOLD signal becomes HIGH, and UD goes LOW to prepare for up counting. The counter will start its up counting when CEN once again becomes HIGH, and stop when CMP is HIGH. Then, the up counting results are stored in the DFFs, and DCDS can be achieved.

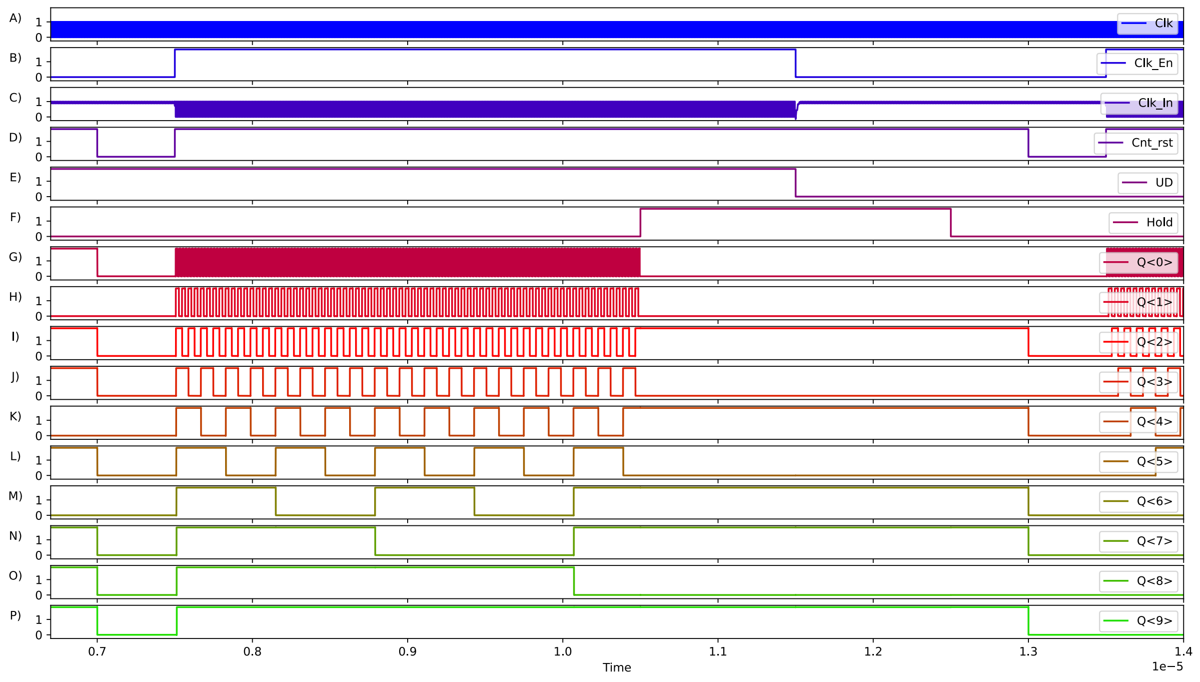


Figure 4.31: Timing diagram of up/down counter

4.9.4 Up/Down Counter Layout

Figure 4.32 is the layout design of the counter. The entire layout is too long to fit within one page and simultanously be visible, so only a small snippet is shown. The components within the design are provided by the foundry.

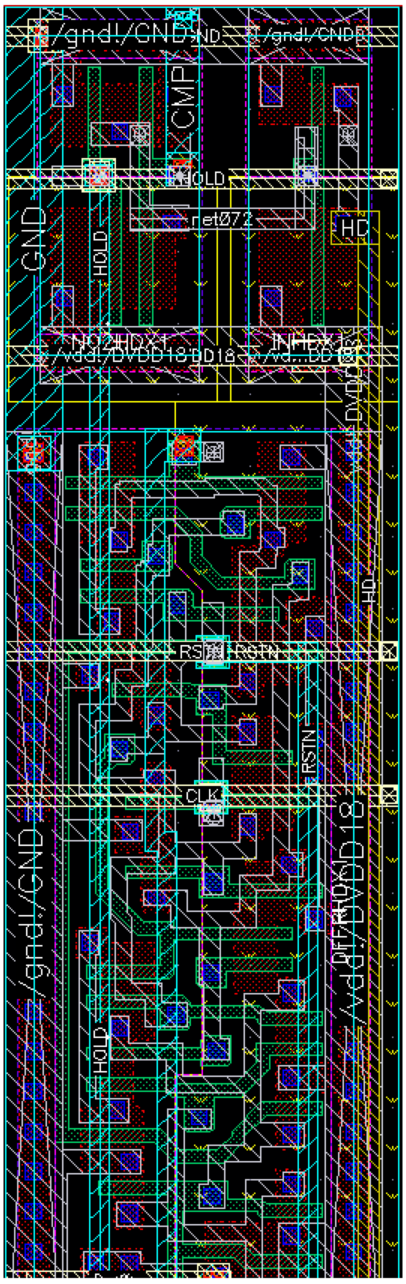


Figure 4.32: Snippet of the counter layout

4.10 Top Level Design

4.10.1 Pad Frame

The final component required for the ASIC design before embarking on the top level is the pad frame. The pad frame in this project conists of 65 pads that can be divided into the different categories shown in table 4.8 and 4.9.

Table 4.8: Pad types for the ASIC core module

Signal	**Voltage [V]**	**Pad type**
Analog Input	3.3	APR01DPC
Analog Input	1.8	APR01DP
Analog Output	1.8	APR00DP
Digital Input	3.3	ICPC
Digital Output	3.3	BT4PC
Digital Input	3.3/1.8	ICP
Digital Output	3.3/1.8	BT4P

Different pads are necessary since they are, first of all, differentiated by analog and digital signals. Secondly, they are dedicated to different voltage levels: 3.3 V and 1.8 V. Lastly, they are differentiated by being dedicated to input vs. output signals.

Table 4.9: Pad types for the pad frame

Signal	Voltage [V]	Pad type
Analog supply	3.3	VDDPADPC
ESD-supply for AVDD33	3.3	VDDORPADPC
Analog supply	1.8/3.3	VDDPADP
ESD-supply for AVDD33	1.8/3.3	VDDORPADP
Digital supply	3.3	VDDPADPC
ESD-supply for DVDD33	3.3	VDDORPADPC
Digital supply	1.8/3.3	VDDPADP
ESD-supply for DVDD18	1.8/3.3	VDDORPADP
Ground	3.3	GNDORPADPC
Ground	1.8	GNDORPADP

4.10.2 Top Level Simulation

Integrating most of the aforementioned sub-blocks into the top layer and running a simulation results in the timing diagram in figure 4.33.

Plot A to D depicts the signal chain for one single pixel, showing **SEL**, **RST**, **TX** and V_{pix} signals.

E to G shows the two comparator inputs Vin_{ADC} and V_{ramp}, and its resulting output, **Cmp**.

The rest of the signals shows the counter signals, where H to M are the control inputs, whilst N to W are the resulting bit outputs representing the digitized signal level.

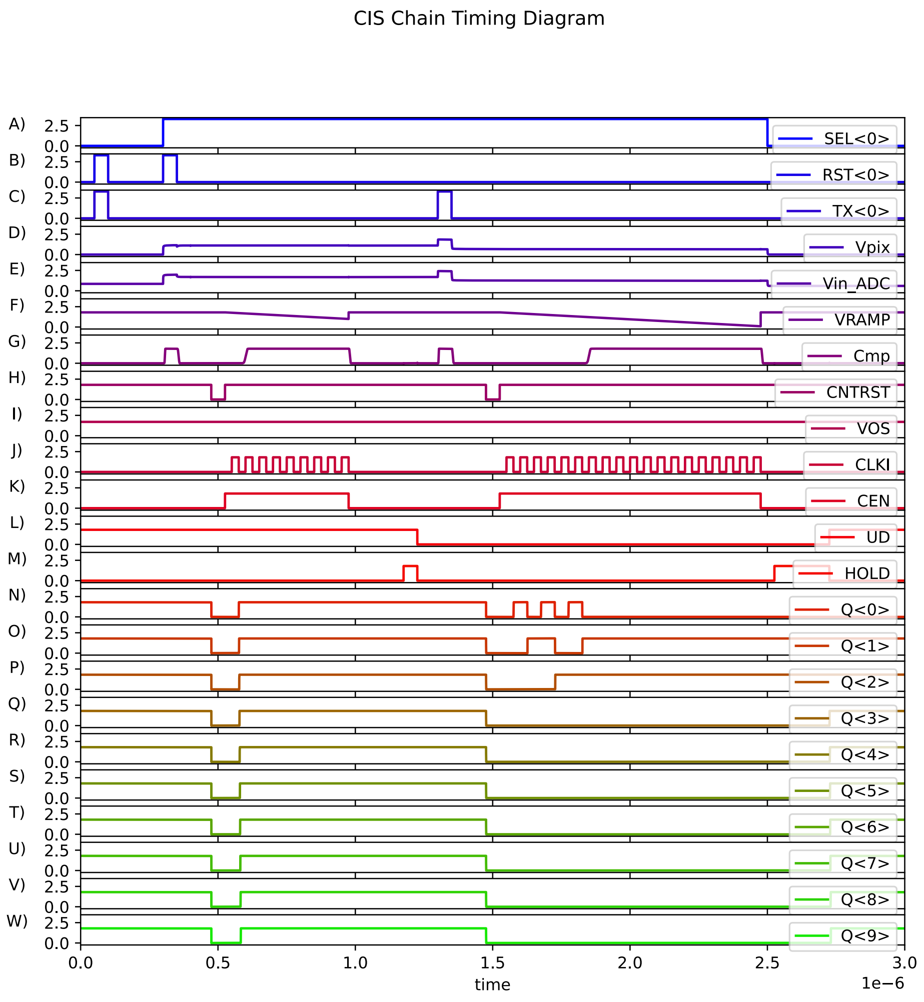


Figure 4.33: Timing diagram of the CIS signal chain, demonstrating the basic timing for reset and pixel voltage sampling for one pixel.

4.10.3 Top Level Layout

Finally, the top level layout is shown in figure 4.34 integrating all the sub-blocks with a pad-frame. Furthermore, some stand-alone circuits for debugging as well as large decoupling capacitors were placed in the remaining available space.

A total of 65 pads were placed around the chip, and 54 were routed to various analog and digital I/O signals on the chip and subsequently out to a multitude of components on the PCB.

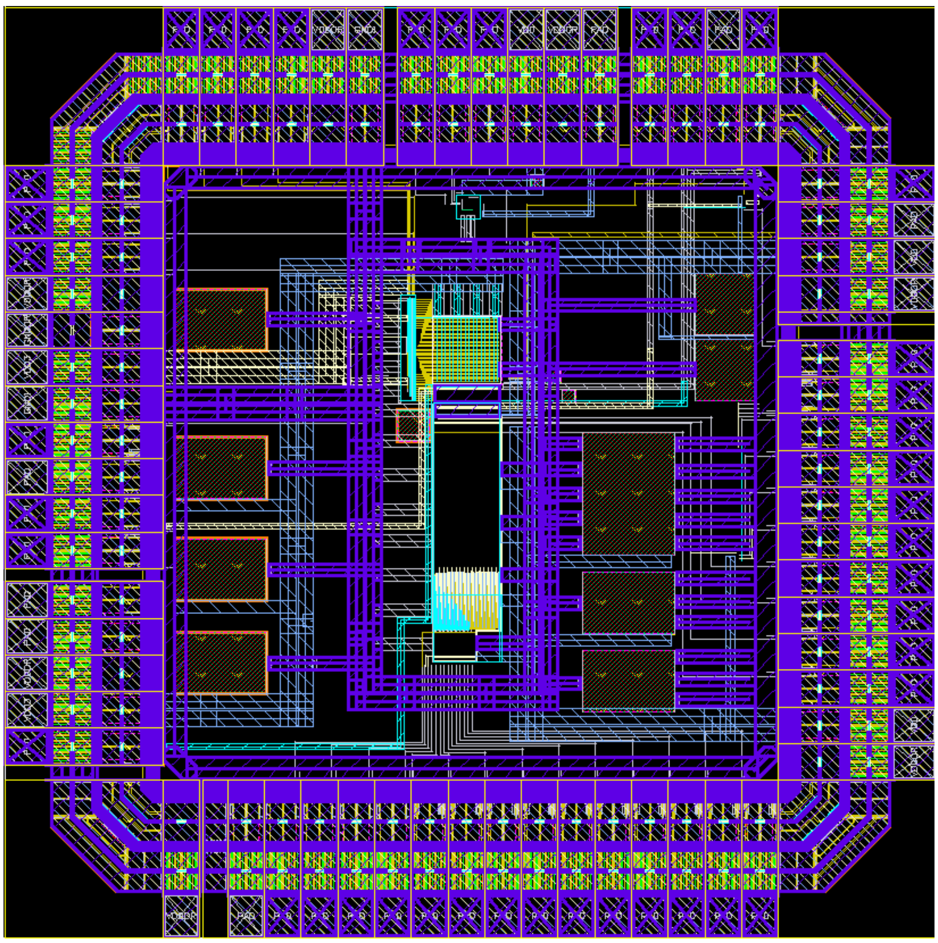


Figure 4.34: Layout of the entire ASIC

4.11 Testability

Debug pins were placed and routed to the pads in various spots for eventual testing, verification and debugging. The output test pins are listed in table 4.10.

Several stand-alone components (e.g. comparator, counter) were placed on the chip in order to do testing and eventual debugging in case the chip would not work as intended.

Note that plenty of extra input pins were needed as well to control the stand-alone components, but listing these were not deemed necessary.

Table 4.10: Debugging pins on the ASIC

Pin	**Description**
DBGP_ROW	Digital output pad for debugging the row decoder.
DBGP_CNT	Digital output pad for debugging the counter.
DBGP_OUT_VPIX_COL	Analog output pad displaying the pixel voltage in col. 15.
DBGP_OUT_CMP	Analog output pad for debugging a stand-alone comparator component.
DBGP_OUT_MUX	Digital output pin for debugging the MUX output.

Chapter 5

Testing and Verification

The test and verification system is comprised of a custom-made PCB, an FPGA board, and a range of laboratory instruments. The test setup was planned to be made by another student who, in the end, decided to not finish the project. In the remaining time, a minimal set-up was attempted to be made.

However, this meant that a full ASIC test would not be possible with the entire pixel array and the readout circuitry for a proper image capture. Throughout the development, it was decided that small functionality tests would be done to verify whether a couple of designs would operate as intended. In addition, the debug pins placed around the chip were measured and used to verify whether certain areas within the CIS were functional as well.

Table 5.1: Instruments used for test and verification of the ASIC.

Instrument	**Model**
DC Power Supply 1	Agilent E361A
DC Power Supply 2	Agilent E361A
DC Power Supply 3	Agilent E361A
Waveform Generator	Agilent 33120A
Digital Multimeter	Keysight 34470A
Oscilloscope	Keysight DSOX1202G

It should be noted that for industrial sensors, a complete characterization of the ASIC is critical for determining if the CIS performance parameters meet the final product specifications. Evaluating resolution, spectral response, characteristics under illumination and in the dark, defects, photo-conversion characteristics and much more are common procedures that cannot be covered fully in this project.

5.1 Printed Circuit Board

The PCB was designed in KiCad 6 with a range of SMD and through-hole components. A stencil came with the order, making most of the assembly possible by the application of solder paste, component placement, and baking in a reflow oven.

The PCB is built of 4 layers: two signal layers, one power plane, as well as a ground plane.

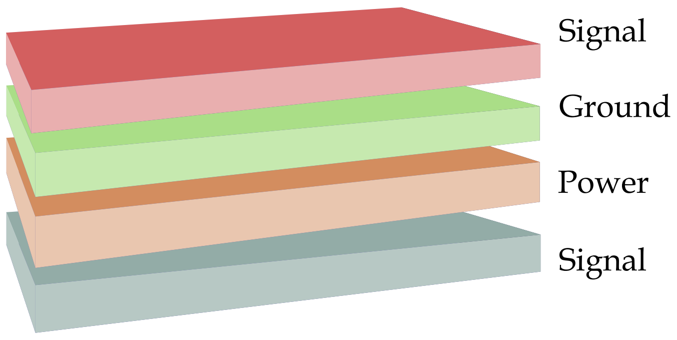


Figure 5.1: Illustration of the PCB layer division

The power domain separation is demonstrated in figure 5.2. Four of the domains serve as core supplies, while the rest are connected to various ESD protection lines in the pad frame.

The components implemented on the PCB are listed below:

A total of 38 digital GPIO pins are connected between the ASIC and the board connector, interfaced by an extended Arduino shield (using a custom footprint that resembles the stanard shield, but with additional pins), four analog voltages connected between the ASIC and potentiometers, and finally, eight jumpers that provide the ASIC's power supply needs.

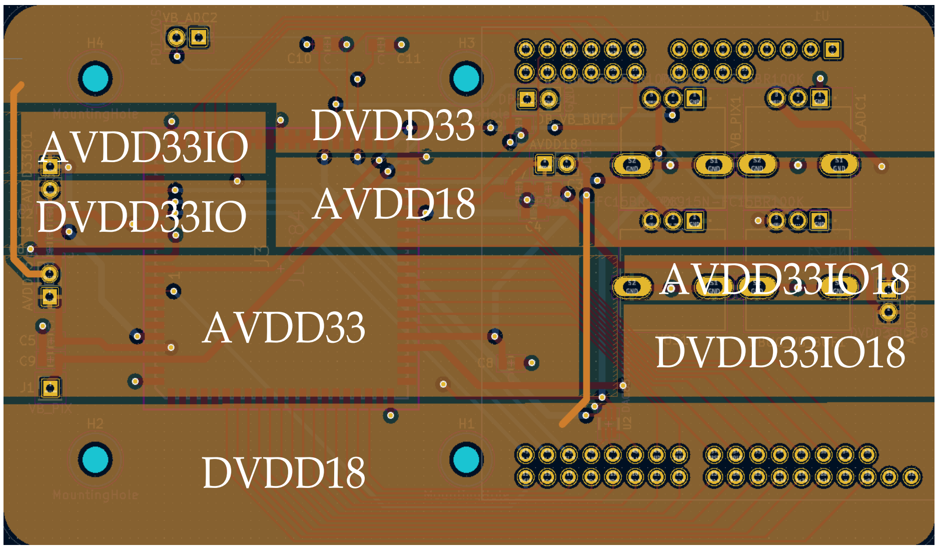


Figure 5.2: Power plane division in the PCB layout

Table 5.2: Table of PCB components

Component	Type
IC Socket	MC-84PLCC-SMT
DAC	DAC8311IDCKT
Potentiometer	P0915N-FC15BR100K
Capacitors	Ceramic SMD
Arduino shield	Custom-made

5.1.1 PCB Layout

The layout is shown below, and only two out of four layers are visible: the top electric layer and the power plane. It was important to make space for the lens mount, and therefore plenty of room is made between the IC socket and the rest of the components. To account for this, approximately 43 cm. was measured between the middle of each mounting hole. This creates a limit to how close components such as decoupling capacitors can be placed within chip proximity. A workaround could be to place the capacitors on the backside, however, this would make the baking process more difficult.

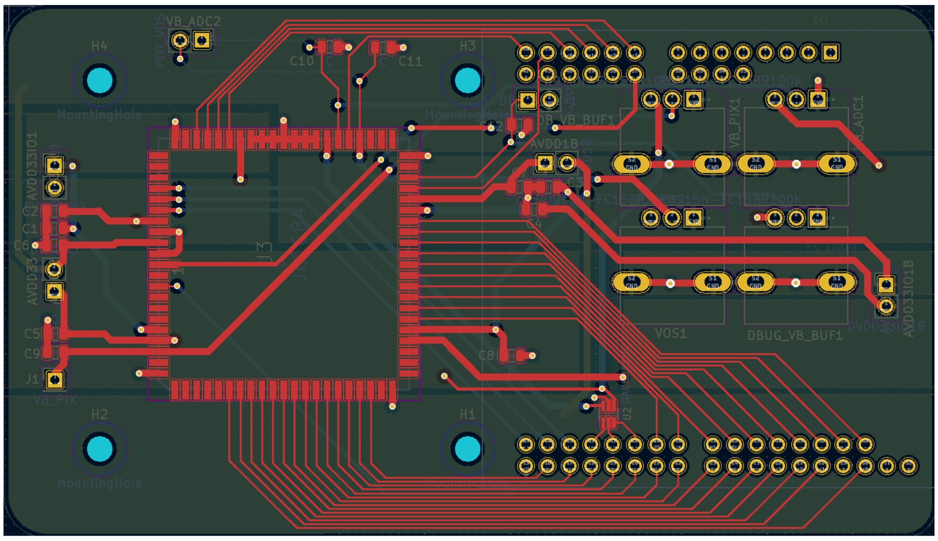


Figure 5.3: Layout of the PCB, showing the top electric layer and traces of the power plane. The bottom electric layer and ground plane are not visible here.

Finally, the following table shows all the signals routed to the Arduino shield on the PCB. Additionally, information about the package position, signal name, and descriptions are provided.

Table 5.3: Package position, shield pin, signal name, and description of each signal.

Package Position	Shield Pin	Signal Name	Description
1	IO14	DBGP_ROW	Debugging of the row decoder
3	IO0	VPC	Clamp enable
7	N/A	AVDD33IO	ESD supply
8	N/A	AVDD33	Analog core supply
9	N/A	VB_PIX	Source follower bias for pixels
17	IO42	$S< 0 >$	MUX address
18	IO13	$S< 1 >$	MUX address
19	IO12	$S< 2 >$	MUX address
20	IO11	$S< 3 >$	MUX address
21	IO10	$DOUT< 0 >$	Digital output of pixel level
22	IO9	$DOUT< 1 >$	Digital output of pixel level
23	IO8	$DOUT< 2 >$	Digital output of pixel level
24	IO7	$DOUT< 3 >$	Digital output of pixel level
25	IO6	$DOUT< 4 >$	Digital output of pixel level
26	IO5	$DOUT< 5 >$	Digital output of pixel level
27	IO4	$DOUT< 6 >$	Digital output of pixel level
28	IO3	$DOUT< 7 >$	Digital output of pixel level
29	IO2	$DOUT< 8 >$	Digital output of pixel level
30	IO1	$DOUT< 9 >$	Digital output of pixel level
36	N/A	DVDD33IO18	ESD supply
37	N/A	DVDD18	Digital core supply
38	IO32	UD	Counter control
39	IO33	CLKI	Counter clock
40	IO34	CRST	Counter reset
41	IO35	HOLD	Counter control

42	IO36	DBGP_VIN_HOLD	Counter control (debug)
43	IO37	DBGP_VIN_CRST	Counter reset (debug)
44	IO38	DBGP_VIN_UD	Counter control (debug)
45	IO39	DBGP_VIN_CNT	Counter input (debug)
46	IO40	DBGP_CNT	Counter control (debug)
47	IO41	DBGP_VIN_CMP	Counter input (debug)
48	N/A	AVDD33IO18	ESD supply
49	N/A	AVDD18	Analog core supply
50	N/A	DBUG_OUT_VPIX_COL	Debug of column readout
51	N/A	DBGP_VB_BUF	SF bias (debug)
57	N/A	VOS	Clamp voltage
58	N/A	DEBUG_OUT_CMP	Comparator output (debug)
59	N/A	VB_ADC	ADC bias
60	N/A	VRAMP	Comparator input ramp
67	N/A	GND	Ground
68	N/A	DVDD33IO	ESD supply
69	A3	$A < 0 >$	Row decoder address
70	A2	$A < 1 >$	Row decoder address
71	A1	$A < 2 >$	Row decoder address
72	A0	$A < 3 >$	Row decoder address
78	IO28	DEC_TX	Row decoder control
79	IO27	DEC_SEL	Row decoder control
80	IO26	DEC_RST	Row decoder control
81	N/A	DVDD33IO	ESD supply
82	N/A	GND	Ground
83	N/A	DVDD33	Digital core supply
84	N/A	GND	Ground
N/A	IO29	DAC_SYNC	DAC control for ramp generation
N/A	IO30	DAC_SCLK	DAC clock for ramp generation
N/A	IO31	DAC_DIN	DAC data for ramp generation

5.2 PYNQ-Z1

Using the built-in PYNQ platform described in chapter 3 was intially desired. This was decided against due to the base overlay provided by Xilinx not meeting the needs of the chip control. Instead of changing the overlay, which would have required changes at FPGA level anyhow, it was decided that HDL programming would be the easiest.

The Vivado 2022.1 program was used for synthesis of hardware description language design, whilst the Vitis IDE 2022.1 platform was used for the embedded application.

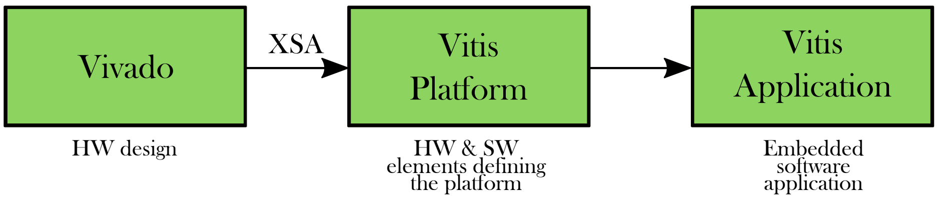

Vivado
XSA
Vitis Platform
Vitis Application
HW design
HW & SW elements defining the platform
Embedded software application

Figure 5.4: Diagram illustrating the design flow of the chip control using an on-board FPGA on the PYNQ-Z1 board.

To achieve this, the following was done:

- Creation of a hardware module in Vivado 2022.1 that includes a Zynq-7000 SoC processor and several hardware elements for functionality and interfacing peripherals.
- Synthesized, implemented, and generated the bitstream before exporting the hardware definition to Vitis IDE.
- Created a C-based application in Vitis IDE 2022.1 and flashed it on the FPGA on the PYNQ board.

The data transfer goes through UART serial communication.

5.2.1 Logic Analyzer

The Digilent Digital Discovery was used for capturing and displaying the digital signals generated by the PYNQ board, as it provides a whole suit of features to allow visualization and simulation of digital signals for embedded projects. The built-in logic analyzer was utilized for this purpose, and the pattern generator was useful to generate signals to perform smaller, isolated tests.

5.3 Assembly

The final assembly is demonstrated in figure 5.5 where the ASIC has been mounted on the socket placed on the PCB, and the PCB is mounted on top of the PYNQ-Z1 board by the means of an Arduino shield. Several banana cables are attached for supplying the PCB with the necessary power and ground signals.

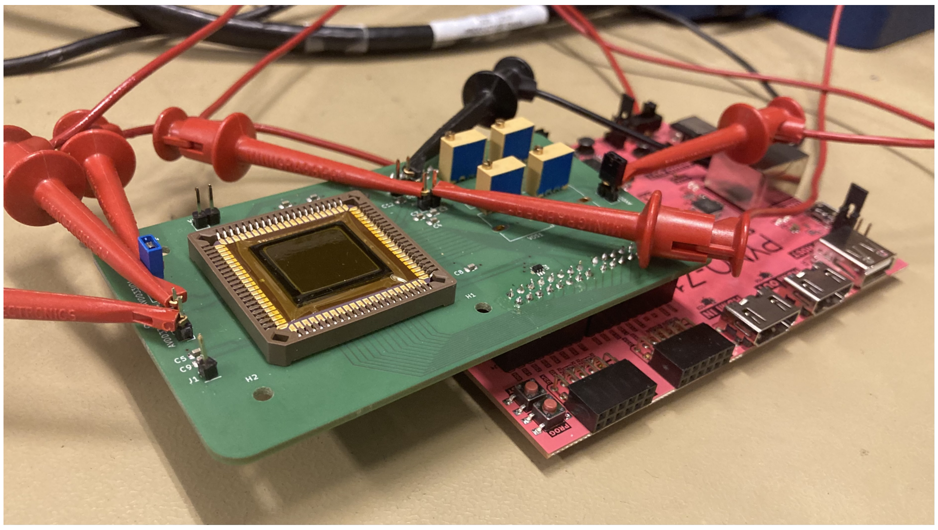


Figure 5.5: The test setup with the ASIC mounted on the PCB, and the PCB attached to the PYNQ board.

5.4 Comparator Verification

To test and verify the comparator, several measurement probes were connected to the comparator pins and observed using an oscilloscope. A waveform generator provides a ramp, and a constant voltage was set as the threshold voltage, V_{in}.

By stepping V_{in} and measuring with an oscillopscope, the resulting comparator output can be seen in figure 5.6. Evidently, the increasing V_{in} voltage causes a shift in the comparator input, as expected. Hence, it is be possible to say that the comparator behaves at a basic level, as a comparator should – by comparing a threshold voltage with a ramp, and switching from a low state to a high state at the transition point.

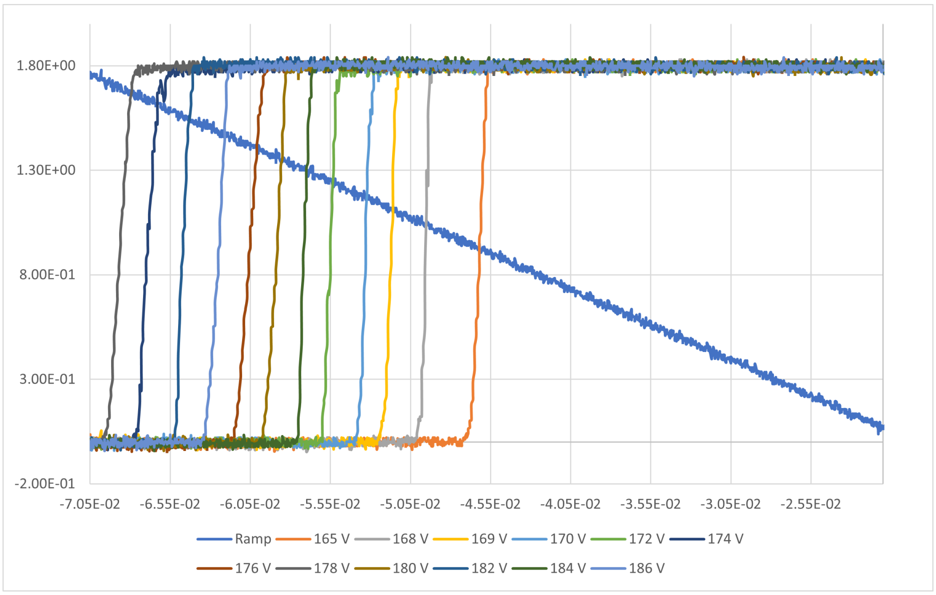


Figure 5.6: Comparator verification displaying measurements done with an oscilloscope.

Unfortunately, it was discovered that the pad (in the ASIC pad frame) had errors. For the comparator circuit, the pad providing an input voltage to the threshold V_{in} was mistakenly made as a digital pad – not analog, as it should be. This digital pad is gated, and is supposed to level shift an input of 3.3 V down to 1.8 V. Somehow, feeding the digital pad with in-between voltages (approximately 1.65 V to 2.2 V), seemed to result in some level-shifting down to values other than the logic levels 0 and 1.8. Presumably, there is a switching-point in the buffer that provides a small working linear range where the comparator is able to operate as it should. Therefore, it was still possible to see some basic comparator behavior by stepping the voltage with fine steps.

However, since it is not possible to know the V_{in} voltage for sure, characterization cannot really be done for this component.

5.5 Row Decoder Verification

For the digital parts, some debug probes were placed inside the main CIS system in case digital errors would occur, and the need for debugging would arise.

The row decoder has such a probe dedicated to debugging, which was probed whilst providing the decoder with its Gray code input addresses. This proved to be useful just to validate the functionality during the testing process. The result can be seen in figure 5.8.

DBGP_ROW is the output of the first NOR gate in the decoder hierarchy, and should oscillate depending on the input address. The resulting pulse wave matches the simulated pattern.

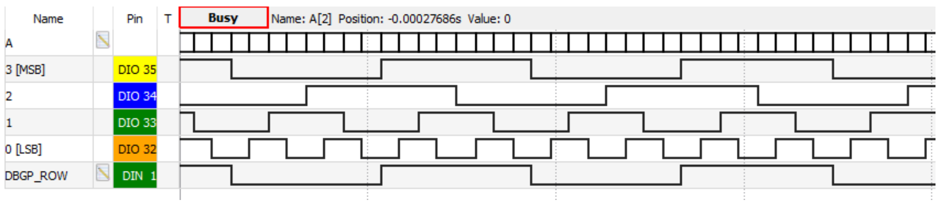


Figure 5.7: Row decoder verification showing four input bit addresses (A<0>-A<3>), and the resulting output DBGP_ROW toggling as expected. This capture was done using a digital logic analyzer.

5.6 Counter Verification

A stand-alone counter circuit was placed outside of the main system, leaving the possibility for an isolated test in order to prove its functionality. One of its outputs is connected to a pad and can subsequently be probed.

In figure 5.8, DBGP_CNT signal is seen. This signal represents the LSB of the resulting 10-bit counter output, and is expected to oscillate with a pulse period that is the double of the clock pulse period. The figure shows that this is the case, and that the counter works as expected.

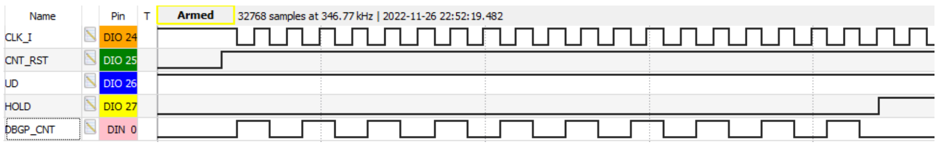


Figure 5.8: Counter verification showing the input clock CLK_I, control signals CNT_RST, UD, HOLD, and an output bit $DBGP\_CNT$. The capture was done using a digital logic analyzer.

To conclude, only a couple of components were testable, since the amount of stand-alone components dedicated to testing was lacking. Ideally, an entire ADC (and not just comparator and counter separately) should have been added outside of the main system. Regrettably, a test pixel was not placed outside of the pixel array either, making the verification of this part difficult.

However, two stand-alone circuits were tested and validated – an analog circuit (comparator) and a digital circuit (counter). The tests show that these are functional. Additionally, the row decoder which is not a stand-alone circuit, but rather a part of the entire system, is validated as well. Hence, nothing is pointing to a non-functional sensor, and further development on the test and characterization system might show promising results.

Chapter 6

Conclusion

The objective of this project was to produce an image sensor resembling a typical industrial sensor; albeit very limited in every aspect, considering the fact that the scope of such a project would under normal circumstances be intended for a collaboration between two students. If such an arrangement would be possible, the workload would ideally be shared (e.g. one making the schematics and conducting simulations, while the other would be designing the layout and creating a testing rig).

Resultantly, a complete system was made without optimization of any subcircuits. The main deviation of this project from previous ones is the on-chip ADC, and therefore there is a larger focus on this part.

Some hiccups occurred along the way, and every part was dealt with and adapted to in the best, and usually simplest possible manner. Firstly, the chip was intended to be manufactured in a technology specific to image sensor applications, namely XFAB's XS018. One of the goals of using this process was to test their image sensor specific photodiodes and look at the performance.

The initial schematic design, simulation and layout design was done using the XS018 technology and sent for tapeout. Unfortunately, the fab cancelled this tapeout a couple of weeks after the final deadline, leaning on the fact that there were too few customers during this particular run. However, they proposed to port the design to another, more commonly used technology – XH018.

A lot of time was spent on the development of a test system, but only a fraction could be dedicated to the actual testing. Further developments of the test and characterization system is therefore needed to verify the performance of the entire ASIC, but these initial tests demonstrate that individual circuits (both analog and digital) are functional. Errors were discovered in the pad frame, but deemed as non-critical. Given enough time, a thorough demonstration of full system functionality and ADC characterization is ideal.

6.1 Future Work

For next iterations of this chip, small improvements can be made to each sub-component, depending on the needs. For instance for the pixel and pixel array, dummy pixels and optical black pixels can be added to improve the image quality. The chosen counter is known to consume more power than desired, and a low-power counter can be implemented instead. Additional sub-systems can be added (e.g. to RGB matrix for colored images), but this would depend entirely on application needs, and only imagination (and available resources) will set the limit.

All of the components within the sub-blocks were simulated using typical corner models. However, an important concern is the mismatches in the MOSFETs. Even though simulations using typical corner models are better than nothing, it is important to run parasitic extractions (PEX), typically by incorporating statistical varations in each device to create a more precise analog model of the circuits.

If a larger pixel array is relevant, the SS-ADC might be limited by long conversion time which is limited by the maximum clock frequency. Improved performance in this regard is possible by selecting other architectures, such as the SAR ADC. Otherwise, improvements can be added to the SS-ADC, like improvements in offset and linearity. E.g. autozeroing technique can be implemented as an offset-cancellation technique. However, these ADC improvements would be more applicable to projects focused on component design specifically.

The most important part is that a proper characterization system should be developed to do a full system verification, but also to characterize sub-blocks such as the comparator and ADC. More stand-alone components for testing and debugging is recommended on the chip, as this seemed to be lacking – most notably, a pixel should have been added for characterization. In addition, placing an entire ADC block outside of the main system would be ideal, rather than only comparator and counter components separately.

Ultimately, development of an application or algorithm for basic image-processing would tie everything together, as it would allow a more seamless construction of an image after acquiring the binary data of each pixel.

Bibliography

[1] P. E Allen. *CMOS analog circuit design*. eng. New York, 2012.

[2] R. Baker. *CMOS: Circuit Design, Layout, and Simulation, Third Edition*. Vol. 18. Sept. 2010. ISBN: 978-0470881323. DOI: `10.1002/9780470891179`.

[3] E. W. Boogart et al. 'Very Low Dark Current CCD Image Sensor'. In: *IEEE Transactions on Electron Devices* (Sept. 2009), pp. 2462–2467.

[4] W.S. Boyle and G.E. Smith. 'The inception of charge-coupled devices'. In: *IEEE Transactions on Electron Devices* 23.7 (1976), pp. 661–663. DOI: `10.1109/T-ED.1976.18470`.

[5] Fernando Campos, Bruno de Castro and J.W. Swart. 'A Tunable CMOS Image Sensor with High Fill-Factor for High Dynamic Range Applications'. In: *Engineering Proceedings* 2 (Nov. 2020), p. 79. DOI: `10.3390/ecsa-7-08235`.

[6] Saemin Im and Sang-Gyu Park. 'Thermal noise analysis of switched-capacitor integrators with correlated double sampling: Thermal Noise of SC Integrators with Correlated Double Sampling'. eng. In: *International journal of circuit theory and applications* 44.12 (2016), pp. 2101–2113. ISSN: 0098-9886.

[7] Jong-Seok Kim, Jin-O Yoon and Byong-Deok Choi. 'Low-power counter for column-parallel CMOS image sensors'. In: *2016 IEEE Asia Pacific Conference on Circuits and Systems (APCCAS)*. 2016, pp. 554–556. DOI: `10.1109/APCCAS.2016.7804028`.

[8] Seunghyun Lim et al. 'A High-Speed CMOS Image Sensor With Column-Parallel Two-Step Single-Slope ADCs'. In: *IEEE TRANSACTIONS ON ELECTRON DEVICES* 56.3 (Mar. 2009), pp. 393–398.

[9] J. Nakamura. *Image Sensors and Signal Processing for Digital Still Cameras*. Optical Science and Engineering. CRC Press, 2017. ISBN: 9781420026856. URL: `https://books.google.fr/books?id=UY6QzgzgieYC`.

[10] Giuseppe Rossi. *Design of a sample-and-hold circuit for high speed CMOS image sensor*. 2008.

[11] Chen Shoushun, F Boussaid and A Bermak. 'Robust Intermediate Read-Out for Deep Submicron Technology CMOS Image Sensors'. eng. In: *IEEE sensors journal* 8.3 (2008), pp. 286–294. ISSN: 1530-437X.

[12] Martijn Snoeij. *Analog Signal Processing for CMOS Image Sensors*. 2007. ISBN: 978-90-9022129-8.

[13] Martijn Snoeij et al. 'COLUMN-PARALLEL SINGLE-SLOPE ADCS FOR CMOS IMAGE SENSORS'. In: (2006).

[14] Jinn-Shyan Wang et al. 'Low power shift registers for megabits CMOS image sensors'. In: *2011 9th IEEE International Conference on ASIC*. 2011, pp. 17–20. DOI: `10.1109/ASICON.2011.6157111`.

[15] Xinyang Wang et al. 'A CMOS Image Sensor with a Buried-Channel Source Follower'. In: *2008 IEEE International Solid-State Circuits Conference - Digest of Technical Papers*. 2008, pp. 62–595. DOI: `10.1109/ISSCC.2008.4523057`.

[16] Mathias H. F. Wilhelmsen. 'Design of CMOS Image Sensor with Linear Readout Amplifier'. In: (2016).

[17] Orly Yadid-Pecht et al. 'Optimization of noise and responsivity in CMOS active pixel sensors for detection of ultralow-light levels'. In: *Proceedings of the SPIE* (Apr. 1997), pp. 125–136.

[18] Timothy York. *Fundamentals of Image Sensor Performance*. Accessed: 25-07-2022. URL: `https://www.cse.wustl.edu/~jain/cse567-11/ftp/imgsens/`.

Appendices

Appendix A

ASIC

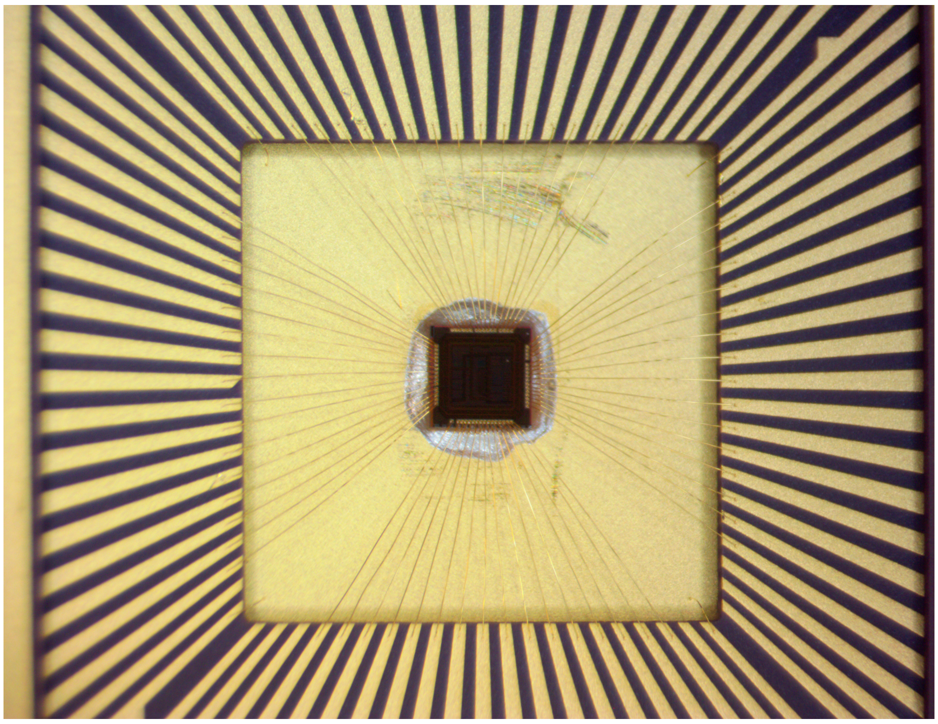


Figure A.1: Microscopy image of the ASIC showing the chip and the bonding to package pins.

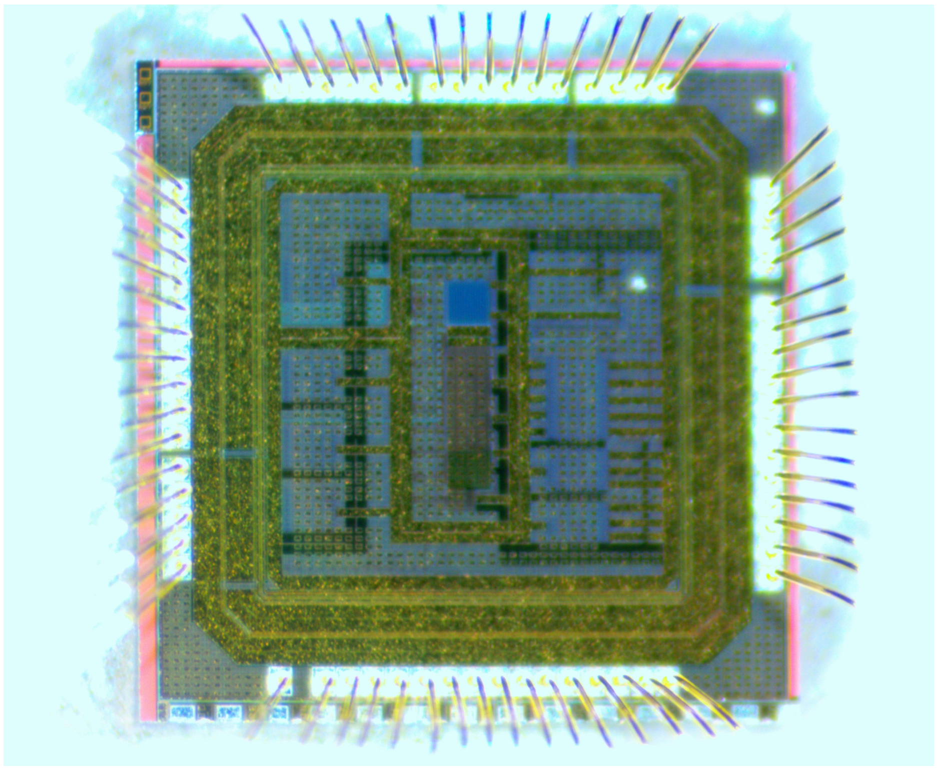


Figure A.2: Microscopy image of the ASIC where the pixel array (blue square) and readout circuits (below the pixel array) are visible.

Appendix B

PCB

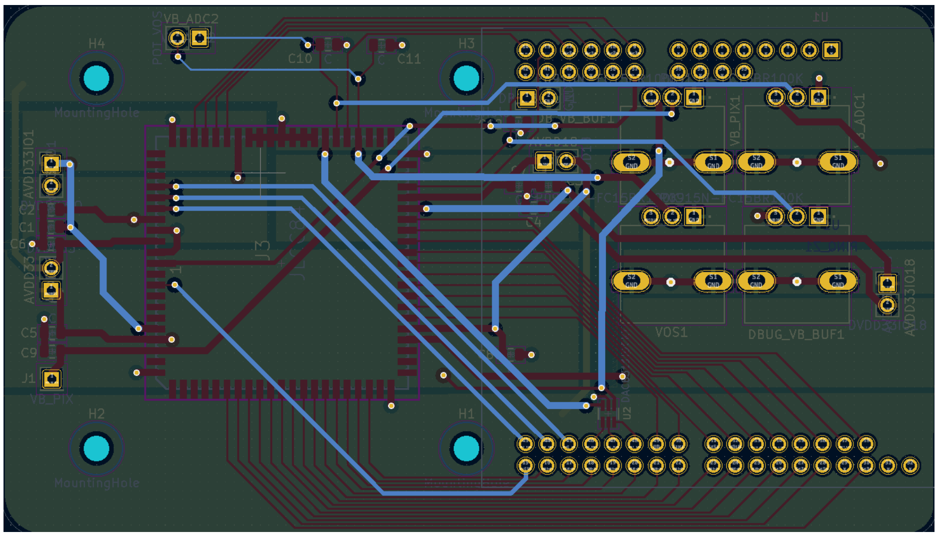


Figure B.1: PCB layout showing the bottom electric layer.

Appendix C

Code

```
#include <stdio.h>
#include <stdint.h>
#include <stdbool.h>
#include "platform.h"
#include "xil_printf.h"
#include "xparameters.h"
#include "xgpio.h"

#define GPIO_DEVICE_0 XPAR_AXI_GPIO_DEVICE_ID
#define GPIO_DEVICE_1 XPAR_AXI_GPIO_1_DEVICE_ID
#define GPIO_INTERNAL XPAR_AXI_INTERNAL_DEVICE_ID
#define GPIO_DAC XPAR_AXI_DAC_DEVICE_ID

// Inputs
#define DBGP_ROW_BP 0
#define DBGP_CNT_BP 1
#define DOUT0_BP 2
#define DOUT1_BP 3
#define DOUT2_BP 4
#define DOUT3_BP 5
#define DOUT4_BP 6
#define DOUT5_BP 7
#define DOUT6_BP 8
```

```
#define DOUT7_BP 9
#define DOUT6_BP 10
#define DOUT9_BP 11

#define DOUT_GP 2
#define DOUT_GM 0x000003FF << DOUT_GP

// Output
#define DBGP_VIN_CMP 0
#define A0 1
#define A1 2
#define A2 3
#define A3 4
#define S0 5
#define S1 6
#define S2 7
#define S3 8
#define DBGP_VIN_CNT 9
#define DBGP_VIN_UD 10
#define DBGP_VIN_CRST 11
#define DBGP_VIN_HOLD 12
#define HOLD 13
#define CRST 14
#define UD 15
#define DEC_TX_BP 16
#define DEC_SEL_BP 17
#define DEC_RST_BP 18
#define VPC33 19

#define CEN 0
#define RESET_CLK 1

#define DAC_DIN 0
#define DAC_SCLK 1
#define DAC_SYNC 2
```

```
#define S_GP 5 //bit shift of row select adr S<3:0>
#define S_GM 0x0F<<S_GP //bit shift of row select adr S<3:0>

#define A_GP 1 //bit shift of col select adr A<3:0>
#define A_GM 0x0F<<A_GP //bit mask of col select adr A<3:0>

XGpio Gpio0; //input
XGpio Gpio1; //output
XGpio Gpio2; //output
XGpio Gpio3; //input

uint16_t grayCode[] = {0b0000, 0b0001, 0b0011, 0b0010, 0b0110, 0b0111,
    0b0101, 0b0100, 0b1100, 0b1101, 0b1111, 0b1110, 0b1010, 0b1011,
    0b1001, 0b1000};
uint16_t bCode[] = {0b0000, 0b0001, 0b0010, 0b0011, 0b0100, 0b0101,
    0b0110, 0b0111, 0b1000, 0b1001, 0b1010, 0b1011, 0b1100, 0b1101,
    0b1110, 0b1111};

int delay_us(int us){
   // TODO function content temporarily empty.need to check cpu frequency

}



int write_pin_value(XGpio * InstancePtr, unsigned Channel, u32 bit, u32
    value){
   u32 temp = XGpio_DiscreteRead(InstancePtr, Channel);
   temp = temp & !(1<<bit); // set everything to 1 except one bit
   temp = temp | (value<<bit);
   printf("temp is %d\r\n", temp);
   XGpio_DiscreteWrite(InstancePtr, Channel, temp);

}

int read_pin_value(XGpio * InstancePtr, unsigned Channel, u32 bit){
   u32 temp = XGpio_DiscreteRead(InstancePtr, Channel);
   if(temp & (1<<bit)) return 1;
   else return 0;
}


int read_DOUT_pins(int value){
   u32 temp_reg = XGpio_DiscreteRead(&Gpio0, 1);
   temp_reg = (temp_reg & DOUT_GM) >> DOUT_GP;
   return temp_reg;
}


int select_row(int adr){
   printf("adr: %d\r\n", adr );
   int temp_reg = XGpio_DiscreteRead(&Gpio1, 2);
   printf("a: %d\r\n", temp_reg );
   temp_reg = temp_reg & !(S_GM);
   printf("b: %d\r\n", temp_reg );
   temp_reg = temp_reg | grayCode[adr]<<S_GP;
   printf("c: %d\r\n", temp_reg );
   XGpio_DiscreteWrite(&Gpio1, 2,temp_reg);
   printf("Write to row sel done");
   return 0;
}

int select_col(int adr){
   printf("adr: %d\r\n", adr );
   int temp_reg = XGpio_DiscreteRead(&Gpio1, 2);
   printf("a: %d\r\n", temp_reg );
   temp_reg = temp_reg & !(A_GM);
   printf("b: %d\r\n", temp_reg );
   temp_reg = temp_reg | bCode[adr]<<A_GP;
   printf("c: %d\r\n", temp_reg );
   XGpio_DiscreteWrite(&Gpio1, 2,temp_reg);
   return 0;
}
```

```
int decoder_rst(int val){
   write_pin_value(&Gpio1, 2, DEC_RST_BP, val);
}



int decoder_tx(int val){
   write_pin_value(&Gpio1, 2, DEC_TX_BP, val);
}

int decoder_sel(int val){
   write_pin_value(&Gpio1, 2, DEC_SEL_BP, val);

}

int vpc_toggle(int val){
   write_pin_value(&Gpio1, 2, VPC33, val);
}

int crst_write(int val){
   write_pin_value(&Gpio1, 2, CRST, 1);
}

int ud_write(int val){
   write_pin_value(&Gpio1, 2, UD, 1);
}

int cen_write(int val){
   write_pin_value(&Gpio1, 2, CEN, 1);
}

int pixel_control_depletion(){
   decoder_sel(0);
   int i=0;
   for(i=0;i<=1;i++){
      select_row(i);
```

```
        decoder_rst(1);
        decoder_tx(1);
        delay_us(1);
        decoder_rst(0);
        decoder_tx(0);
    }
    return 0;
}

int counter_reset(){
    ud_write(1);
    crst_write(1);
    delay_us(1); //delay
    ud_write(1);
}

int counter_countdown(){
    ud_write(1);
    crst_write(1);
    delay_us(1); //delay
    ud_write(1);
}

int pixel_control_sample_reset(){
    decoder_sel(1);
    int i=0;
    for(i=0;i<=1;i++){
        select_row(i);
        decoder_rst(1);
        delay_us(1);
        decoder_rst(0);

        vpc_toggle(1); // start reset signal sampling
        delay_us(0.5);
        vpc_toggle(0.5);
        /*
```

```
      for (j=0;j<=15;j++){
         column_operation();
      }
      */
   }
   return 0;
}

int pixel_control_sample_tx(){
   int i=0;
   int integration_time=5;
   for(i=0;i<=1;i++){
      select_row(i);
      decoder_tx(1);
      delay_us(1);
      decoder_tx(0);
   }
   delay_us(integration_time);
   decoder_sel(0);
   return 0;
}

int pixel_control(){
   pixel_control_depletion();
   pixel_control_sample_reset();
   pixel_control_sample_tx();
   pixel_control_depletion();
}

void initialize(){

   int status = XGpio_Initialize(&Gpio0, GPIO_DEVICE_0);
          if (status != XST_SUCCESS) {
            print("Err: Gpio initialization for gpio0 failed\n\r");
         } else {
            print("Info: Gpio initialization for gpio0 successful\n\r");
```

```
    }

     XGpio_SetDataDirection(&Gpio0, 1, 0xFFFFFFFF); //channel 1 have
         all inputs

status = XGpio_Initialize(&Gpio1, GPIO_DEVICE_1);
        if (status != XST_SUCCESS) {
          print("Err: Gpio initialization for gpio1 failed\n\r");
        } else {
          print("Info: Gpio initialization for gpio1
              successful\n\r");
        }
        printf("END!!\r\n" );
        XGpio_SetDataDirection(&Gpio1, 1, 0xFFFFFFFF); //channel 1
            have all inputs
        XGpio_SetDataDirection(&Gpio1, 2, 0x00000000); //channel 2
            have all outputs

status = XGpio_Initialize(&Gpio2, GPIO_INTERNAL);
           if (status != XST_SUCCESS) {
             print("Err: Gpio initialization for internals
                 failed\n\r");
           } else {
             print("Info: Gpio initialization for internals
                 successful\n\r");
           }
           printf("2\r\n" );
           XGpio_SetDataDirection(&Gpio2, 1, 0x00000000); //channel 2
               have all outputs
/*
status = XGpio_Initialize(&Gpio3, GPIO_DAC);
              if (status != XST_SUCCESS) {
                print("Err: Gpio initialization for DAC failed\n\r");
              } else {
                print("Info: Gpio initialization for DAC
                    successful\n\r");
                }
                printf("1\r\n" );
                XGpio_SetDataDirection(&Gpio3, 1, 0x00000000);
                    //channel 2 have all outputs */
    int line = 10;
    int linegrayCode = grayCode[line];
    printf("The value for %d in gray is %x\r\n", line,linegrayCode );
    select_row(10);
    printf("select_row executed");
    //select_col(6);
    printf("END!!\r\n" );
    return 0;

    /*
     print("Hello World\n\r");
        print("Successfully ran Hello World application");
        cleanup_platform();*/

}



int main()
{
    init_platform();

    initialize();

    volatile int delay = 0;

/*
    while(1){ // TEST FOR OUTPUTS
      print("Info: OUTPUT HIGH\n\r");
      //XGpio_DiscreteWrite(&Gpio, 2, 0xFFFFFFFF);
      write_pin_value(&Gpio, 2, 6, 1);
      for (delay=0; delay < 100000000; delay++); // wait for 10000000
          clock cycles and do nothing (aka nop)
      print("Info: OUTPUT LOW\n\r");
      //XGpio_DiscreteWrite(&Gpio, 2, 0x00000000);
      write_pin_value(&Gpio, 2, 6, 0);
      for (delay=0; delay < 100000000; delay++); // wait for 10000000
          clock cycles and do nothing (aka nop)

    }
*/

    write_pin_value(&Gpio3, 1, 2, 0); // TEST FOR OUTPUTS - write and
        probe

    /*
    while(1){ // TEST FOR INPUTS - connect to ground to see changes
      if( read_pin_value(&Gpio0, 1, 1) ){
         print("Info: Changes\n\r");
      }else{
         print("Info: 0\n\r");
      }
    }
   */

    return 0;
}
```