Design and test of a CMOS image sensor with global shutter and High Dynamic Range

A camera suitable for capturing scenes with fast moving objects and/or unstable illumination sources

Marius Lunder Lillestøl

Thesis submitted for the degree of Master in Nanoelectronics 60 credits

Department of Informatics Faculty of mathematics and natural sciences

UNIVERSITY OF OSLO

Spring 2017
Design and test of a CMOS image sensor with global shutter and High Dynamic Range

A camera suitable for capturing scenes with fast moving objects and/or unstable illumination sources

Marius Lunder Lillestøl
Design and test of a CMOS image sensor with global shutter and High Dynamic Range

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

Printed: Reprosentralen, University of Oslo
Abstract

A CMOS image sensor with global shutter mechanism, HDR capability and high shutter efficiency has been designed and tested. The image sensor consists of a 128x128 pixel array, readout circuitry and special components for global shutter and skimming operations. The Image sensor has been designed using AMSoptim0.35µm process. Hardware and software has been successfully added for testing and verification purposes. The global shutter functionality has been verified through comparison with a rolling shutter camera under the same conditions. Furthermore the HDR skimming technique has been implemented and verified through comparison with a normal capture operation.
# Contents

I Introduction  
1 Thesis outline  
1.1 Workload  

2 Background  
2.1 Short introduction to the Image Sensor  
2.1.1 The pixel  
2.1.2 CCD and CMOS  
2.2 Image Sensor properties  
2.2.1 Attributes  
2.2.2 Noise in Image sensors  
2.3 Techniques and solutions to inadequate Image sensors  
2.4 Global shutter vs. Rolling shutter in CMOS Image sensors  
2.4.1 Rolling shutter  
2.4.2 Global shutter in CMOS  

3 Motivation  

II Method  

4 Schematics and simulations  
4.1 Overview  
4.1.1 Planning  
4.2 Pixel  
4.3 TX control and M1 buffer  
4.4 Row- and Column decoder  
4.5 Sample & hold  
4.6 Output buffer
5 Layout
  5.1 Pixel ................................................................. 35
  5.2 TX control and M1 buffer ............................................. 38
  5.3 Sample & hold, CDS .................................................. 39
  5.4 Top and pad frame ..................................................... 41
    5.4.1 Pads ............................................................. 41
    5.4.2 Inner circuit ..................................................... 41

6 Image sensor test system .................................................. 43
  6.1 PCB ................................................................. 45
    6.1.1 Analog to digital converter ...................................... 45
    6.1.2 Lens and lens mount ............................................ 47
  6.2 System on chip ....................................................... 47
    6.2.1 FPGA ............................................................ 47
    6.2.2 Qsys generated soc_system ..................................... 50
    6.2.3 C program ....................................................... 52
  6.3 PC ................................................................. 54
    6.3.1 Java ............................................................. 54
  6.4 Test system verification ............................................... 55
    6.4.1 Skimming problems ............................................. 56
    6.4.2 Full well capacity .............................................. 58

III Results ................................................................. 59

7 Results ................................................................. 61
  7.1 Global shutter performance and image quality analysis ............ 61
  7.2 High dynamic range ................................................... 63
  7.3 Measurements ......................................................... 65
    7.3.1 Power consumption .............................................. 65
    7.3.2 Image sensor characteristics ................................... 65
  7.4 Future work ........................................................ 69
    7.4.1 Improvement .................................................... 69
    7.4.2 Future projects ................................................ 70

8 Conclusion ............................................................... 71

Bibliography ................................................................. 71

A Top layout of CMOS image sensor ....................................... 75

B Pixel layout and pixel Array layout .................................... 77
C  Full Image sensor under microscope  79
D  Pixel array under microscope  81
List of Figures

1.1 Thesis outline .................................................. 3
2.1 A reverse biased photodiode .................................. 6
2.2 Shows the decreasing potential in a photodiode for different amounts of light ................................................. 7
2.3 Shows different image capturing schemes in CCD and CMOS[11] . . 8
2.4 3T active pixel .................................................. 11
2.5 a) PN photodiode with source follower, b) Pinned photodiode with source follower [12] ........................................... 12
2.6 4T active pixel with pinned photodiode ....................... 13
2.7 Rolling shutter operation ...................................... 15
2.8 Effects of a rolling shutter capturing a fast moving object[4] .......... 15
4.1 Schematic top view of the image sensor design ................. 22
4.2 Schematic of the suggested 4T pixel ......................... 23
4.3 Schematic of the suggested 5T pixel ......................... 24
4.4 Simulation of the suggested 5T pixel, with three different simulated light intensities. The timing is not to scale in this simulation due to demonstration purposes ............................................. 24
4.5 Simultation setup for different M1-buffers .................... 26
4.6 Schematic of the TX control circuit. The transfer gate pMOS input is inverted, but not shown in this hierarchy .................. 27
4.7 The 1 to 2 decoders ........................................... 29
4.8 A comparison between the 8-256 decoder on the top and the 7-128 decoder on the bottom ........................................... 30
4.9 A comparison between the 4-16 decoder on the top and the 3-8 decoder on the bottom. Notice that A<3> signal is removed as it is not needed anymore. ........................................... 30
4.10 Logic controlling select and reset signal for a given row ........... 31
4.11 Correlated double sampling circuit, one in each pixel column .... 32
4.12 6T pixel allowing true CDS .................................. 33
List of Tables

4.1 Truth table for TX-control unit ........................................... 27
4.2 Natural binary ................................................................. 28
4.3 Gray code ................................................................. 28

6.1 Key parameters of LTC1746 ............................................ 46
6.2 Lens specifications ......................................................... 47
6.3 HPS command menu ..................................................... 53
Preface

This thesis summarizes the authors work and learning outcome through the five years as a student. It has been written to complete the M.Sc Degree in Informatics: Nanoelectronics at the University of Oslo. Through this whole project I have been accompanied by fellow student Soman Cheng. Together we have been challenged in almost everything we have learned up until the project, including chip design, programming in multiple languages, circuit board design, electronics theory and physics, and many more.

After enrolling a great course on Image Sensor Circuits and Systems, held by Adjunct Professor Johannes Sølhusvik, it became clear that this was by field of interest. How things in general work has always fascinated me, but less often is it about a topic that everyone can relate to. When Johannes later presented an available master thesis on Global shutter image sensor I was quick to apply. When Soman also applied it was only seen as a great benefit to the project as it became a realizable goal to develop a fully functional camera.

I would first and foremost like to thank Soman Cheng for being the best companion imaginable during this long and hardworking project. I would like to thank Johannes for being a good supervisor, sharing his great enthusiasm for the project and pushing us forward when dearly needed. Thanks to fellow students Mathias H. Wilhelmsen and Stian Sumstad for sharing their experience on their image sensor project, and for building the foundation needed for this project, and hopefully many to come. A great thanks to Olav Stanly Kyrvestad, Senior Engineer at NANO Resarch Group, for always providing good answers whenever I came running to your office, and for the talks we had after the mystery had been solved. Thanks to Philipp Dominik Häfliger, professor at the NANO Resarch Group, for the exact same reasons with Olav.

A big thanks to my supporting family, especially to my two lovely younger sisters, and thanks to my fellow students for giving me the best 5 years so far in my life.

The biggest of all thanks should none the less go to my dearest Yi-Ven for staying with me through five years of late home from school and all the "no time to make dinner today, we’d better get some sushi" days we have shared.
Part I

Introduction
Chapter 1

Thesis outline

The thesis is structured to represent the timeline of this project. It starts with the theoretical background studied and presenting previous achievements in the field of image sensors. This is followed by the planning phase of the project, where considerations have been made and simulations performed. Next is the design phase of the Image Sensor, where details on the Layouts are described followed up by the creating of a test environment for the finalized image sensor. Finally the results will be presented and discussed.

Figure 1.1: Thesis outline
## 1.1 Workload

It should be noted that both participants have contributed to each others distributed tasks, and that the division line shows who should be credited the most for the work done.

<table>
<thead>
<tr>
<th>Work Distribution. LO=Layout</th>
</tr>
</thead>
<tbody>
<tr>
<td>Marius</td>
</tr>
<tr>
<td>Planning</td>
</tr>
<tr>
<td>Schematics</td>
</tr>
<tr>
<td>Pixel LO</td>
</tr>
<tr>
<td>TX-control LO</td>
</tr>
<tr>
<td></td>
</tr>
<tr>
<td>Sample &amp; Hold LO</td>
</tr>
<tr>
<td>M1 buffer LO</td>
</tr>
<tr>
<td>TOP and Padframe LO</td>
</tr>
<tr>
<td>FPGA and VHDL</td>
</tr>
<tr>
<td>C program</td>
</tr>
<tr>
<td>PC and Java</td>
</tr>
</tbody>
</table>
Chapter 2

Background

Digital image sensors are found everywhere in today’s society. They are not only found in digital SLRs, but also in everything from your car to the Curiosity Mars rover[7]. Over the last couple of decades there has been many revolutions improving on these devices. Major advancements in technology has made engineering an image sensor into a very complex process. However there is still a lot to improve on when it comes to image quality as well as size and power consumption. They have without a doubt become a big part of our everyday society, and the demand for a good image sensor in a small frame has never been more sought after.

This chapter will first give an introduction to the pixel and it’s operation, together with a short introduction to the two main image sensor technologies. Section 2.2 will focus on properties of the image sensor and 2.3 will present some of the previous achievements in the image sensor field. Section 2.4 will compare the more common rolling shutter to the global shutter.

2.1 Short introduction to the Image Sensor

2.1.1 The pixel

The principle of capturing light and convert it into a measurable unit is the same for all image sensors. In the core of an image sensor lies a 2D array of photodiodes (PD). A photodiode at it’s simplest is a regular pn-junction diode, and will have a depletion region between the n-type silicone and the p-type silicone. This depletion region, when reversed biased, expands and becomes electrically floating. If a photon that has more energy $E_{\text{photon}} = \frac{hc}{\lambda}$ than the bandgap in silicon hits the depletion region, it will be absorbed and generate an electron-hole pair through the photoelectric effect.
When the switch in Figure 2.1 is closed the diode is connected to supply voltage. This charges up the intrinsic capacitance in the diode and increases the depletion region between the n-doped and the p-doped silicone, as mentioned. This action is called reset, and it prepares the photodiode for a new measurement. When the switch is again opened the integration\textsuperscript{1} starts. The potential across the photodiode will remain fairly stable and close to VDD if not exposed to any light. If however light is allowed to reach the diode, the photons will start generating electron-hole pairs. The excited electron will start to drift towards the positively charged n-type silicone and the hole in the opposite direction. This again generates a current in the diode. The current will now result in a potential drop across the diode, which can be measured at $V_{out}$. This drop will continue as long as light hits the diode. The end of integration comes whenever the voltage is measured or sampled. The diode is then reset to high potential by closing the switch; ready for a new capture. The amount of photons hitting the diode, for a given time period, is determined by the intensity of the incident light. Because of this the voltage loss in the diode will vary as more or less electron-hole pairs are generated. Figure 2.2 shows how different light intensities give different voltage drops. Voltage, compared to counting electrons, makes for a convenient way of measuring the intensity of the light hitting one given photodiode. A photodiode arranged so that it can be measured is used in image sensors and makes up what is called a pixel.

---

\textsuperscript{1}Integration in this context is the time span in which the photons hitting the diode are accounted for
Figure 2.2: Shows the decreasing potential in a photodiode for different amounts of light

2.1.2 CCD and CMOS

The two main image sensor technologies used today are the semiconductor charge-coupled device (CCD) and the complementary metal oxide semiconductor (CMOS), see Figure 2.3. The CCD transfers the charges from the photodiode, after integration, directly through MOS capacitors close together in a vertical line. Every pixel is connected to its own MOS capacitor, and one line is found in every column. As the image is taken, all the pixel charges are transferred to its MOS capacitor at once. The vertical line, acting as a shift register, moves the charges down the column in parallel. All the columns ends up in a horizontal shift register that serially moves the charges to a charge detector. This detector converts the charge to a voltage and amplifies the signal. The CCD has what is called a global shutter operation where all pixels are captured simultaneously. The standard CMOS image sensor on the other hand, works very differently. Instead of shifting each pixel to a storage node like the MOS capacitors in CCD. The pixel output is selected through a transistor, one by one, and transferred through conducting metal paths. What pixel should be selected at any given time is decided by a horizontal and vertical scanner.

In the early days of digital imaging, the CCD was the leading technology due to its superior image quality. Since the 1990’s however the CMOS has gradually taken over the marked. As the fabrication technology improved, and the demands kept climbing, researchers started again testing image sensor designs using CMOS technology. This led to many revolutions in the 1990s that gradually made CMOS a real competitor to CCD again. It was well known that CMOS technology used less power and that it allows for a high level of integration compared to the CCD.
Figure 2.3: Shows different image capturing schemes in CCD and CMOS[11]

The fact that the pixels in CMOS technology is made in the same process as in logical circuitry, allows for logic to be placed on the chip itself together with the pixel array. This can greatly reduce the size of a fully functional camera, and gives room for complex logical operations. This became a valuable feature as the desire to have cameras in mobile phones and other devices arose. Such devices does not require a very high image quality and were therefore suited for the CMOS image sensor. The already blooming production of processors and memory for computers using CMOS technology, made the CMOS image sensor very cheap to manufacture compared the specialized and expensive processes required to make CCDs. As CMOS technology advanced they started taking over the market and today they dominate the professional DSLR market as well. Although this is the case, the CCD technology lives on and is used for scientific, medical and high performance cameras due to it’s superior image quality and light sensitivity.
2.2 Image Sensor properties

To start of this section, the early problems with the PN photodiode and CMOS technology will be discussed. Following this will be the introduction to some of the revolutions that helped improve on the CMOS image sensor.

2.2.1 Attributes

Full well capacity

The full well capacity is defined as the amount of charges a pixel can store up to the point of saturation.

\[ N_{sat} = \frac{1}{q} \int_{V_{reset}}^{V_{sat}} C_{PD}(V) dV \]

Where \( N_{sat} \) is the number of electrons, \( q \) is elementary charge \( 1.60218 \times 10^{-19} \text{C} \), \( V_{reset} \) is the potential at reset, \( V_{sat} \) is potential at saturation and \( C_{PD} \) is the PD capacitance.

A large area PD will have a higher \( C_{PD} \) can store more charges thus increasing it’s full well capacity.

Conversion gain

Conversion gain is the potential drop, i.e the voltage difference per accumulated electron and has unit \( \mu \text{V/e} \)

\[ CG = \frac{q}{C_{PD}} \]

High conversion gain is proportional to light sensitivity as \( \Delta V \) is higher for each electron accumulated.

Dynamic range

Dynamic range in imaging is the ratio between the highest and the lowest light intensity measurable limited by the noise floor. A camera with high dynamic range will be able to capture more of the bright and dark areas in a scene. For images taken with lower dynamic range it is common to gain up the dark areas and attenuate the bright in post processing in order to achieve this desired effect. This however, has its limitations as it will not add any extra information in the picture than what is already there. This means that if a dark area is to dark, brightening it will only make it uniformly brighter, and not reveal extra detail in that area. There are many different approaches in order to achieve HDR, and it is still a very popular research field today. One of the more common techniques relies on taking multiple images
with different exposure time and combine them into one by selecting the the highest pixel values from each that is not saturated. This gives good results but there are many limitations to it as well. First of all the camera has to be perfectly still, which is not often possible with mobile phone cameras and/or compact cameras. Another problem is that the images still needs to be run through an algorithm. Some of the most high end cameras can do this on chip, but standard commercial cameras often requires software editing tools.

2.2.2 Noise in Image sensors

Limitations regarding the passive pixel

Although the CMOS passive pixel sensor (like the setup in Figure 2.1) was a major breakthrough at it’s time it had many limitations when it came to capturing good quality images. The signal from the photodiode had to be carried all the way out of the chip which lead to a lot of added noise to the final image. This also limited the size of the sensor as the signal could only go so far before it was to weak and unusable due to the dominant noise.

Fixed pattern noise (FPN)

Fixed pattern noise is local offsets in the image present in the image over multiple captures. This is typically seen as vertical or horizontal lines in the image, where one column could be brighter or darker relative to the others.

Dark current

Dark current is the reverse bias leakage current which exists in PN-diodes[2]. Each pixel will have different amounts of this current which will results in a fixed pattern noise which will be the same for every capture made. However on top of this leakage current lies a shot noise which consists of thermally generated electron-hole pairs in the diode that follow the Poisson process, meaning they are completely random.

kTC noise

A problem related to MOS technology is the so called reset noise (kTC noise) which appears as the charge-detecting node is reset[10]. This is the same phenomenon as thermal noise in capacitors which can be calculated as \( V_n^2 = kT/C \) or as the electrical charge standard deviation \( Q_n = \sqrt{kTC} \) or \( Q_n^2 = kTC \). Where k is Boltzmann’s constant, T is the temperature in kelvin and C is the capacitance. Small
differences in transistors and temperatures across the sensor will lead to different reset values for each individual pixels. Because of this the integration in each pixel will start at different values for each capture, resulting in a noisy image.

**Image Lag**

Image lag is seen as a ghost effect where information from a previous frame is present in the current image. This is typically seen in images of moving scenes with a high contrast in light intensity. The effect is usually caused by an incomplete reset of the photodiode between frames leaving residues behind.

### 2.3 Techniques and solutions to inadequate Image sensors

**Active pixel sensor**

One of the major breakthroughs came with the active pixel sensor (APS)[6]. APS is a setup in CMOS that gives each pixel its own amplification stage. This is typically done using a source follower as shown in Figure 2.4. This helped reduce the image noise significantly. It also helped speed up the readout process which now became faster than the opposing CCD. The final big advantage was that the pixel output signal could be carried further internally in the circuit, making it possible to add a lot more pixels to the sensor giving a higher resolution and a better image quality.

![Figure 2.4: 3T active pixel](image)

Figure 2.4: 3T active pixel
Correlated double sampling

In order to reduce the reset noise (kTC noise), correlated double sampling is suggested[8]. This process samples both the pixel value and the reset value. The pixel value is then subtracted from the reset value and it is this difference that gives the final value of the given pixel. When the two are subtracted the result will be the same, regardless of the initial offset, for the same amount of electrons accumulated in the diode. First the reset transistor is pulsed and the reset potential of the PD is sampled immediately, while the potential is still high. From this the integration goes as normal until the PD potential is sampled again, providing the actual image signal. This implementation reduces kTC noise considerably, however it requires some extra sample and hold logic.

Pinned photodiode, PPD

In order to reduce image lag, originally in CCD image sensors, a pinned photodiode\(^2\) structure was invented by Teranishi et al. in 1982 [16]. This was done by creating a sandwich structure consisting of a heavily doped \(p^+\) layer on top of the \(n\)-layer of a normal photodiode. In addition a transfer gate (TX) separating the photodiode from the readout node was added, see Figure 2.5. The readout node is referred to as Floating Diffusion node (FD node)\(^3\). This allows for the entirety of the electrons in the photodiode to be transferred to the readout node, thus removing image lag. Another benefit from the \(p^+\) layer is that it greatly reduces surface generated dark current. This invention became the introduction to the 4T pixel in Figure 2.6, where the transfer gate becomes the 4th gate in addition to the 3T pixel.

Figure 2.5: a) PN photodiode with source follower, b) Pinned photodiode with source follower [12]

\(^2\)It was named pinned photodiode by Burkey et al. [3] in 1984

\(^3\)The floating diffusion node term points to the \(n^+\) region as seen in Figure 2.5. This node is not connected to any source and is therefore floating. The \(n^+\) region used to be made through diffusion, now through implants, but has kept its name since then
The 4T pixel resets the PD by pulsing both \textit{RESET} and \textit{TX}. After integration \textit{TX} is again pulsed to transfer the potential from \textit{PD} to \textit{FD}. The signal is now ready to be selected by pulsing \textit{SELECT}.

![Diagram](image)

\textbf{Figure 2.6: 4T active pixel with pinned photodiode}

\textbf{Skimming}

Skimming\textsuperscript{4} is a technique mentioned in [13] and is used to increase the dynamic range of the Image Sensor. It does so by allowing the pixels experiencing low illumination to accumulate throughout the exposure time while shortening the exposure time of the "closer to saturation" pixels. The simplest implementation of this scheme pulses the TX gate with half potential, leaving it half way open, during exposure. This transfers only the pixels near saturation to the FD node, skimming the top, leaving the low lit pixels to continue accumulation in PD. The transferred electrons are now removed by resetting the FD node, and some exposure time is given the skimmed pixels to accumulate. This removes the need for multiple full frame captures allowing more movement in the scene.

\textsuperscript{4}\text{Also known as Lateral overflow HDR}
2.4 Global shutter vs. Rolling shutter in CMOS Image sensors

2.4.1 Rolling shutter

The most common capture method in CMOS technology is rolling shutter. Rolling shutter is the opposing readout method to global shutter, which is standard for CCD. In a global shutter all pixels across the whole array are integrated then sampled simultaneously, which one would assume is necessary for an image to be taken. However a so called electronic shutter, today commonly called rolling shutter, was suggested by Eric Fossum in 1997 [17]. This scheme rapidly resets and samples pixel row by row\(^5\) instead of the whole array, see Figure 2.7.

In the beginning of a frame, a reset signal traverses down the pixel array row by row, resetting the photodiodes, and starts the integration. Soon following, at the same rate and with a fixed distance is the select signal. The distance between these signals can be used to decide the integration time, as by delaying the select signal will allow the reset row to be exposed light for a longer period. When the reset signal reaches the last row it simply starts from the top again, as do the select signal. One major advantage is that the continuous sampling allows for a continuous output which is suitable for video capture, without requiring extra memory elements. It also means that there is no time lost between captures, as when waiting for all rows to be read out one by one as in global shutter sensors. Producing one full frame in rolling shutter takes approximately the same time as for a global shutter, as all rows needs to be sampled recursively for both. The rolling scheme however, allows for more exposure time per pixel within this frame. This again makes it possible to increase the speed of the traversing reset and select signals\(^6\), which enables very high frame rates compared to the amount of light needed.

The rolling shutter operation, although being the most common today, has some limitations. The fact that there is a delay between each row, makes the rolling shutter weaker when it comes to capturing fast moving objects. As a fast moving object is captured at different points in time for each row it can appear shifted or stretched. Skew and smear in images are well known artifacts caused by a rolling shutter mechanism.

---

\(^5\)One can also do this column by column, though it’s not as common

\(^6\)if the data rate can be handled by external components
2.4.2 Global shutter in CMOS

Most of today’s commercial CMOS image sensors, like the ones used in smartphones, makes use of the rolling shutter mechanism. In the high end marked today there are a few CMOS sensors that has adopted the global shutter mode[14]. Similar to a CCD, when a CMOS sensor is in global shutter mode, each pixel begins and ends its exposures simultaneously, and the signal is stored in each pixels previously mentioned FD-node. The signals are then read row by row as the A/D converters situated in each column clocks through the sensor. Before beginning the next exposure, there is a global reset to eliminate any charge accumulated during readout.
and ensure every pixel achieves an equal and simultaneous exposure. Since the architecture of a CMOS sensor currently requires that each row of pixels is read out individually, the last few pixels has to wait in their FD-node while the others are read out one by one. While it’s waiting there it is very vulnerable to noise thus creating a big challenge in designing global shutter CMOS image sensor with good image quality. With the newest nanometer technology one could theoretically add a lot of in chip memory and A/D converters to the sensor in order to achieve close to instantaneous readout, however this would be pricey and not suitable for commercially available cameras, as of now. Many researchers work today on achieving global shutter CIS witch high enough image quality that can be sold commercially and finally integrated into a smart-phone.
Chapter 3

Motivation

CMOS image sensors have become a far developed technology, yet many challenges remains and improvements can be made. As the title suggest the main theme of this project is designing a global shutter CMOS image sensor. Since there has been so many improvements to the image sensor regarding noise reduction, light sensitivity and resolution, the rolling shutter effects presented earlier are one of the biggest remaining problem in modern cameras. The world is switching to LED illumination due to longer lifetime and lower cost. This introduces a big problem for rolling shutter cameras as the exposure time is asynchronous with the pulsating light source from the LED lamps. This is especially the case of HDR cameras based on multiple exposures combined together. The goal is therefore to design a 128x128 pixel CMOS image sensor with global shutter pixels and special readout timing that supports HDR capture without artifacts from LED flicker. The pixels will have circuitry to temporarily store the captured image, as well as anti-blooming/shutter devices to enable (partial) integration and simultaneous start/stop of (partial) integration across all pixels. The ‘partial’ reference is in case of HDR capture. Without ‘partial’ integration the sensor should give a linear response.
Part II
Method
Chapter 4

Schematics and simulations

This chapter will go through the different components used in the CMOS image sensor integrated circuit. Why and how we chose to implement each component will be discussed, together with an explanation to how they operate. The row- and column-decoders are modified versions of Mathias H. Wilhelmsen and Stian Sumstad’s original design. By reusing already tested circuits, a lot of time is saved and more attention to the new components is allowed.

4.1 Overview

The image sensor consists of a 128x128 pixel array, row and column decoders, sample and hold circuitry, transfer gate control (TX control), output buffer and M1 buffer. All of these can be found in figure 4.1, and will be discussed individually in the following sections.

4.1.1 Planning

Given 5\text{mm}^2 total chip area as a guide\textsuperscript{1}, and knowing the minimum requirements for a global shutter operation, some rough estimates had to be made to make sure everything would fit. It is highly recommended to draw sketches and be as accurate as possible to make sure that the design you choose will fit within the restricted area.

\textsuperscript{1}The 5\text{mm}^2 requirement was early estimate done by group leader at the NANO group. The total area per student will vary as more or less students apply for ASIC production, and no guaranties were given at this stage
4.2 Pixel

Being the most important part of an image sensor, the pixel is without a doubt the center of attention. The pixel design will decide which surrounding components needs to be implemented in order for it to operate properly. It is therefore often the best place to start when planning the chip. Due to the maximum chip size requirement of 5\( \text{mm}^2 \), we early decided to go for the simplest possible solution in order to save space, allowing more pixels. The AMS C350 process provides a p-cell of a standard n-well in substrate diode with anti-reflective coating\(^2\) which is tested by AMS and optimized with regards to dark current and responsivity. As discussed in section 1.2.4, in order to obtain global shutter operation a minimum of 4 in-pixel transistors is needed, and the pinned photo diode is very much preferable due to it’s superior noise reduction and image lag prevention. Contacting Europractice regarding the technology available, we were told that AMS doesn’t offer any process which supports pinned photo diodes. This lead us to use the available p-cell provided together with a standard transistor to separate the diode from the storage node. The first proposed pixel design is shown in Figure 4.2. Although somewhat

\(^2\)p-cell PHDNWA850 found in PRIMLIB
unconventional, it was an attempt to mimic the typical 4T pixel that uses PPD.

The proposed pixel behaved as expected in simulation, despite being a regular PD and not a pinned photodiode. It was however noticed that the TX gate would open after some time, allowing all the charge stored in the FD node to flow back to the now lower potential in the PD. Looking at the potential across the diode we realized that after some time it had become negative. This was enough to "reopen" the TX gate as its gate-source voltage exceeded threshold, even though the gate voltage was zero. The photodiode will act as a current source as long as photons are accumulated, and will eventually hit negative potential if not reset in time. In a global shutter image sensor signals, especially for the last rows, has to be stored for a relatively long time while waiting to be sampled. If not accounted for this loss of signal would at least corrupt the data in the last few rows which are sampled last. A simple solution to this problem would be to move the reset signal over to the PD and activate reset directly after the signal transfer. This would keep the PD at VDD potential, preventing the reopening of the TX gate. Since it is of interest to be able to completely empty the FD-node of charge we added a separate reset transistor to the PD, instead of moving the reset pulse. This also gave us separate control over the PD and the FD-node, allowing us to reset the FD-node without having to open the TX-gate. The resulting 5T pixel schematic can be seen in Figure 4.3, and the PD reset transistor is called M1.

The photodiode used in schematics is provided together with the p-cell, and can be used for simulations. The only parameters that can be changed are width and length which can be between 5µm-150µm. The schematics PD acts as a non-ideal
current source, with current gain matching the real pixel, based on measurements done by AMS themselves. The amount of current produced by the PD can be controlled by adding a voltage to the light terminal as seen in the schematics. This allows for simulations to be done, though not providing any insight towards the light sensitivity of the PD. The plot in Figure 4.4 shows the simulated FD-node response together with the suggested timing diagram.

First both Global reset and M1 gates are opened to reset the photo diode and the FD-node. Notice that as the gates close the FD-node drops slightly. This is
caused by charge injection as discussed in the introduction part, and it implies that correlated double sampling is advisable. Notice that the potential at PD is simulated as well and that it starts to drop as soon as integration starts. This simulation is done for three different simulated light intensities by stepping the input to the provided PD symbol. When TX goes high the FD node voltage follows the PD voltage until closed. The FD node holds the last value stable after TX end. Once the SELECT signal is pulsed the output from the source follower can be seen on V_OUT. A reset of the FD node is then performed and the reset value is sampled. Based on the simulation it is confirmed that the pixel outputs different values for different lights intensities and that the reset signal is more or less the same for all. Another thing to notice is the slow decent of the output signals after SELECT goes low. The simulation uses a capacitive load to simulate the column bus. In addition a biased nMOS connects to ground, behaving like a fixed current source, in order to empty the load after output is deselected. This biased gate is found in the Sample & hold circuit, and is crucial to allow the column bus to be emptied of charge between reads. The slow decent could suggest that the current can be increased in this case. However increasing it to much can decrease the output range and it should therefore be adjusted to account for both.

4.3 TX control and M1 buffer

The TX and M1 signals that toggles their respective transistors are so called global signals. This means that they have to toggle the transistors in each and every pixel simultaneously. Both the TX control circuit and the M1 buffer was added post pixel-array layout. The M1 and TX column and row routes were included in the pixel array in order to extract the parasitic capacitance on these. The extracted loads were used to determine the buffer size/strength needed to drive the signal up and down within a reasonable amount of time compensating for the RC-delay. Considering that these signals has do drive over 16000 gates gate-capacitance each, this step is crucial in order to ensure a good response. The extracted parasitics on each net ended up being as high as 50pF. The resistance of the whole net was measured to be 160kΩ. This is somewhat misleading as this takes into account the whole length of the net and counts it as one. Since the M1 signal is designed to branch out to every column, the resistance of one column would be more accurate to use in a test. The resistance of one column was extracted and found to be 1.1kΩ. A load of 50pF and 1.1kΩ was therefore added when designing the M1 buffer and TX control unit.
M1

The M1, being a digital signal, needs only a large enough buffer in order to drive all the transistors, and this was tested in a simple simulation with different buffer sizes driving the load. The setup can be seen in Figure 4.5. Testing the standard library buffers only, expecting that they will be optimized in both performance and size, we ended up choosing the BUFx32.

![Figure 4.5: Simultation setup for different M1-buffers](image)

TX-control

Since this project includes experimenting with the skimming technique to capture HDR images, it is required that the TX signal should be able to switch between different fixed voltage levels. In order to control these levels we chose to have three external analogue inputs, each with its own potentiometer for adjustability. Logic was implemented in order to quickly switch between the three fixed potentials as seen in Figure 4.6, based on two addressing signals S1 and S2 respectively. In addition to the logic a large transmission gate is added to each signal in order to drive the global TX-signal’s high load. The logic circuit controls the transmission gates by opening one of them, allowing only one of the fixed voltage signals to drive the output to the TX gates at any given time. Table 4.1 shows the truth table for the TX-control circuit, where LVL1, LVL2 and MAX are the adjustable voltage potentials from PCB\(^3\). The output then connects to the TX gate in each pixel. The voltage inputs can be adjusted by pot-meters as mentioned. This together with the ability to select them with S1 and S2 signals externally gives great freedom when it comes to both timing and voltage levels.

\(^3\)Notice that S1=S2=0 simply toggles the nMOS, closest to the output (TX), thus connecting the output to ground.
As mentioned in the introduction to this chapter, this circuit is based on the design by Wilhelmsen M.H. and Sumstad S. The original design consists of a 8-256 decoder using Gray code input addressing. The Gray code is special in that it only toggles one bit for each increment when counting in binary. Table 4.3 and 4.2 shows natural binary and Gray code. Looking at one example from the tables we see that going from 3 to 4 in natural binary counting changes all three bits whilst Gray code only one. This is very beneficial to this project as it eliminates any risk of glitching. For example outputting the wrong address for a brief moment between increments. A glitch could in theory result in the reset of a row of pixels before intended, thus corrupting the data from this whole row.

In order to get an understanding of the row decoder we start by looking at the

---

4Also known as reflected binary code (RBC)
Table 4.2: Natural binary

<table>
<thead>
<tr>
<th>Decimal</th>
<th>b0</th>
<th>b1</th>
<th>b2</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>1</td>
<td>0</td>
<td>0</td>
<td>1</td>
</tr>
<tr>
<td>2</td>
<td>0</td>
<td>1</td>
<td>0</td>
</tr>
<tr>
<td>3</td>
<td>0</td>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td>4</td>
<td>1</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>5</td>
<td>1</td>
<td>0</td>
<td>1</td>
</tr>
<tr>
<td>6</td>
<td>1</td>
<td>1</td>
<td>0</td>
</tr>
<tr>
<td>7</td>
<td>1</td>
<td>1</td>
<td>1</td>
</tr>
</tbody>
</table>

Table 4.3: Gray code

<table>
<thead>
<tr>
<th>Decimal</th>
<th>g0</th>
<th>g1</th>
<th>g2</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>1</td>
<td>0</td>
<td>0</td>
<td>1</td>
</tr>
<tr>
<td>2</td>
<td>0</td>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td>3</td>
<td>0</td>
<td>1</td>
<td>0</td>
</tr>
<tr>
<td>4</td>
<td>1</td>
<td>1</td>
<td>0</td>
</tr>
<tr>
<td>5</td>
<td>1</td>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td>6</td>
<td>1</td>
<td>0</td>
<td>1</td>
</tr>
<tr>
<td>7</td>
<td>1</td>
<td>0</td>
<td>0</td>
</tr>
</tbody>
</table>

lowest level in the hierarchy, where we find the 1-2 decoder as in Figure 4.7. They consist of either two NAND gates or two NOR gates. Looking at the NAND gate first, we see that one of the logic gates take the input from the address A+ and the other gate has input from the same signal inverted A-. The remaining inputs on each gate are connected to the same signal that acts as an enable. The NAND gate will only give an output if the enable signal is high, and the opposite is true for the NOR version. Since the NAND configuration output is active low it cannot be connected to another NAND and expect it to give the same operation. This is where the NOR configuration comes in, and will take the active low input from a NAND and generate a active high output. Alternating these two configuration for each address bit input and creating a tree construction with a depth of N-bits allows for any \( N - 2^N \) decoder to be made. To preserve the symmetry of the Gray code every other gate in the tree column is mirrored with regards to it’s outputs. The tree structure can be seen in Figure 4.9. Since an equal amount of NAND and NOR gates are found in each branch, and the branches are of the same length, the delay from input to output is expected to be the same for all inputs.
The original design was of an 8 to 256 decoder, meaning that 8 address bits provides 256 different outputs. This project however, needs to address only 128 rows. The choice of having 128 rows and columns falls on the fact that \(2^7 = 128\) makes a realizable \(N - 2^N\) decoder, providing no unused address space. A simple dissection of the original design is needed in order to obtain the desired 7-128 decoder. The 8-256 decoder consist of one 4-16 decoder, the top square in each of the schematics, where each of it’s 16 outputs branches out to it’s own 4-16 decoder. The outputs consist, in other words, of 16 decoders with 16 outputs each, giving a total of \(16 \times 16 = 256\). It is now clear that since exactly half of the outputs are needed for this project, simply dividing the original design in half will do the job. A comparison between the row decoders is found in Figure 4.8.

The top 4-16 decoder has to be changed into a 3-8 decoder. There is now 8 4-16 decoders that takes care of the 4LSB’s\(^6\) and the remaining 3MSB’s\(^7\), out of 7 total, should be taken care of by a 3-8 decoder. Again it is just a matter of dividing the original 4-16 decoder as shown in Figure 4.9.

---

\(^5\)This 4-16 decoder, takes care of the 4MSB from the row address, whilst the other 16 4-16 decoders takes care of the 4LSB

\(^6\)Least Significant Bit

\(^7\)Most Significant Bit
Figure 4.8: A comparison between the 8-256 decoder on the top and the 7-128 decoder on the bottom

Figure 4.9: A comparison between the 4-16 decoder on the top and the 3-8 decoder on the bottom. Notice that A<3> signal is removed as it is not needed anymore.

This decoder can now be used both as Row and Column select, since there is an equal amount of pixels in each direction, 128x128. The column decoder is simply a copy of the decoder at this stage, and need therefore no further explanation.
Row decoder with multiple outputs

The row decoder is extended to choose whether the row-select signal or the row-reset signal for each row should be turned on. This is done by adding the logic in Figure 4.10 to each row-decoder output. The circuit will now output select and/or reset, depending on their value, to one row at a time. The global reset signal is also implemented here. By adding an OR operation\(^9\) between the original row reset signal and the global signal, the global signal simply bypasses the decoder and overrides the row-reset when needed. This allows for either the entire pixel array to be reset for global shutter operation. It also allows for one row only to be reset, needed for the correlated double sampling scheme.

4.5 Sample & hold

As addressed earlier, correlated double sampling should be implemented to compensate for the reset (kTC) noise. The suggested CDS in[1] has been simulated together with pixel schematics to verify that it works as intended. Figure 4.11 shows the circuit implemented. As a pixel row is selected, the output of the APS source follower from each pixel in this row is allowed out to the column lines. The signal is held here until row select goes low. There is one CDS for each column, and the signal from the pixels go directly to these. The CDS samples a signal on the capacitor based on a pulse to either the SAMPLE SIGNAL or the SAMPLE RESET gates shown in the figure. Once both the signal and reset are sampled the SELECT OUTPUTS signal toggles to output them in parallel. The SELECT OUTPUT signal comes from

---

\(^8\) RGLOB in Figure 4.10

\(^9\) In order to stick with NAND/NOR logic, NOR inverted = OR was chosen
the column decoder, which scans through all the CDS’s one by one and outputs the stored signals to a the shared buses SIGNAL OUT and RESET SIGNAL OUT which goes to the output buffer. The COLUMN BIAS is there to ensure that the column bus is emptied of charge between reads as mentioned in the pixel section.

**non true CDS**

The global shutter requires simultaneous reset of all pixels. Since the signals has to be stored row by row, it not possible to sample the reset signal before the image signal in our 5T pixel implementation. This is a problem since the image signal has an offset from the previous reset and not the sampled reset. This is still a decent way of reducing kTC noise, but not at efficient as if the reset and the signal belonging to it were sampled together. In order to obtain true CDS in a global shutter a 6T pixel, seen in Figure 4.12 can be used.

This pixel features an additional sample signal (SS) gate. The PD signal can be stored in between this and TX while the FD node is reset and sampled. A pulse to SS will transfer the signal to the FD node, and can from here be sampled as well.

**nMOS as capacitor**

It was decided to test whether a nMOS could replace the standard capacitor in order to save space in layout. The suggested standard capacitor was of type poly1-poly2 separated by an oxide layer, whilst an nMOS cap utilizes the capacitance between
poly1 and diffusion separated by thin oxide. We have that

\[ C = \varepsilon_r \varepsilon_0 \frac{A}{d} \]

Where \( C \) is the capacitance, \( \varepsilon_r = \varepsilon_{oxide} \) (in our case) is the permittivity in oxide relative to the permittivity in vacuum, \( \varepsilon_0 \) is the electric constant \( \varepsilon_0 \approx 8.854 \times 10^{-12} F/m \), \( A \) is the area overlap between two conducting sheets and \( d \) is the distance between them, thickness of oxide in our case. We see that reducing the distance \( d \) through the means of choosing a thinner oxide layer will increase the capacitance per area given the same oxide material.

In order for the nMOS cap to work, it’s source, drain and bulk are all connected to ground, and the gate to the sample signal. With a capacitance requirement of \( C = 600 \, fF \)\(^{10} \) the required area of the nMOS were calculated as:

The process parameter provided by AMS as a typical value.

\[ C_{ox, poly1-diff} = 4.54 \frac{fF}{\mu m^2} \]

Since

\[ C = C_{ox} A \Rightarrow A = \frac{C}{C_{ox}} \]

we get

\[ A = \frac{600 \, fF}{4.54 \frac{fF}{\mu m^2}} = 132.16 \mu m^2 \]

\(^{10}\)Empirical value
Choosing nMOS Length \( L_n = 9 \mu m \) to make sure it is less or equal to the width of a pixel

\[
W_n = \frac{A}{L_n} = \frac{132.16}{9 \mu m} = 14.68 \mu m
\]

Rounding up to \( W_n = 15 \mu m \) gives a capacitance of

\[
C = Cox W_n L_n = 4.54 \frac{fF}{\mu m^2} \times 135 \mu m^2 = 612.9 fF
\]

Figure 4.13: Comparison between poly1-poly2 cap (Left) and nMOS-cap (right), both have \( C = 612.9 fF \)

### 4.6 Output buffer

The output buffer is actually just two pMOS biases with source connected to VDD, one for signal and one for reset signal, used to counter the pMOS source follower in the CDS. This is usually found within the correlated double sampling schematic. Since only one CDS output is active at any given time in our implementation, they can share the same bias, and save a lot of space.
Chapter 5

Layout

This chapter will discuss the layouts, and choices made before and during the process. It should be noted that this section only covers the authors layout designs in depth. It is therefore referred to Soman Cheng’s thesis for further detail on the remaining layouts.

<table>
<thead>
<tr>
<th>Layout workload</th>
</tr>
</thead>
<tbody>
<tr>
<td>Marius</td>
</tr>
<tr>
<td>Pixel</td>
</tr>
<tr>
<td>TX-control</td>
</tr>
<tr>
<td></td>
</tr>
<tr>
<td>Sample &amp; Hold</td>
</tr>
<tr>
<td>M1</td>
</tr>
</tbody>
</table>

5.1 Pixel

By using measurements from Mathias and Stian’s design together with the known size of the pads, the pixel area was estimated to be:

\[ \text{Pix}_{\text{grid}} = 1846 \mu m \times 1479 \mu m \]

Taking the shortest of the two sides gives

\[ \frac{1479 \mu m}{128 \text{ pixels}} = 11.5546875 \frac{\mu m}{\text{pixel}} \]
meaning that the pixel must be

\[ \text{Pix}_\text{area} \leq 11.55 \mu m^2 \]

Keeping in mind that even 0.1µm adds up to 0.1µm × 128 = 12.8µm, the first pixel design was made to see how small it could be. The pn-diode provided by AMS-opto was chosen to have the smallest possible size of 5µm × 5µm. The p-cell comes with a guard ring around the whole pixel giving it a total area of 7µm × 7µm, and can be seen in Figure 5.1a. AMS also warned us that no changes to this design could be made without losing the guarantee that it would work. The decision was to use the p-cell by AMS-opto, provided it would fit within the size requirements. The first design is found in Figure 5.1b

![Figure 5.1: The AMS-opto pixel (a), and the 5T-pixel layout v1 (b)](image)

Seeing from layout version 1 that it was possible to stay within the given area requirement, it was decided to stick with the AMS-opto pixel. Version 1 of the layout had to be discarded since the routing connecting to all the signals has to stay within the area, which would be impossible in this iteration without crossing another signal on the same layer.

**Version 3**

Version 3, Figure 5.2, of the layout managed to route all the wires through barely within the required area, with its size of 11.5µm × 11.5µm. A layer of metal 4 is also added, covering all the logic except the sensitive area of the pixel. Besides shielding the logic from light hitting the pn-junctions in the transistors, thus preventing current to occur and generate noise, the layer is connected to VDD. This provides easy access to the two transistors that requires VDD. Another great advantage is
that extra VDD signal stability is added when connected to a large surface with high capacitance. This is highly recommended as all pixels have two transistors each that will draw current from VDD simultaneously during global reset. This version is not

![Figure 5.2: Pixel layout v3, 11.5µm x 11.5µm](image)

without flaws either. The problem here is that the two bottom signals are in the same metal layer and that they both are horizontal. The initial plan for these signals was that one would come from the left and the other from the right. These two signals have to occupy one whole side of the pixel array each as they are the global signals M1 and TX, and they have to fork out to all pixels row wise or column wise. This should not be a problem as they run parallel to each other and never crosses, but as the row and column decoder was finished it was discovered that these two signals could not come from the bottom or the left, as they would intersect with the row and the column decoder signals. Having them come from the same side but in different layers wouldn’t work either as they would intersect with the one vertical signal.

**Version 6**

Several more attempts was made to meet all the criteria that now had entered the scene, and version 6 became the final layout for the pixel. Studying Figure 5.3 we see that we have 3 horizontal and 2 vertical signals in separate layers. The figure also gives a clear view of how the pixels behaves when side by side. The final layout is slightly smaller than version 3 ending up at 11.25µm x 11.25µm. This gives a fill factor of

\[
FF = \frac{5\mu m \times 5\mu m}{11.25\mu m \times 11.25\mu m} \times 100% = 19.75\
\]
5.2 TX control and M1 buffer

As mentioned, the global signals TX control and M1 has to come from the top and the right sides of the pixel array. From here they fork out to every column/row as illustrated in Figure 5.4.

The M1 buffer is placed right in the middle on the pixel arrays top edge to minimize the travel length of the signal. This was the intention for the TX-control as well except on the right edge. With Nilsen’s design occupying this space it was moved to the top right corner in the end.

The sizes of the transistors were chosen so that $W_{pmos} = 2 \times W_{nmos}$, in order to compensate for the weaker mobility in the pMOS transistors. Ideally the pMOS could have been even wider, but with twice the width it was easy to match the height of the nMOS in layout by adding twice as many gates. Adding gates to a transistor allows it to become more compact, and closer to a square, rather then a very elongated rectangle. The amount of gates added were chosen so that both transistors would fit within the GND and VDD rails of the standard library logic gates. See figure Figure 5.5 for the TX-control layout.
5.3 Sample & hold, CDS

Designing the CDS is similar to designing the pixel in that everything needs to be confined within a limited area and should be placed close to its neighbour. Since the size of the two nMOS capacitors already has been determined, the rest of the circuitry has to fit within these limits. It was decided to increase the length of the nMOS caps to match the pitch of the pixel and let the grounded source and drains on these be shared with it’s neighbours. The main problem with the rest of the logic lies with the combination of n and pMOS transistors as they require separate bulk connections. On top of this it was chosen to sandwich all the logic between the capacitors. This was to provide identical travel length of the two signals and to reduce crosstalk between the large capacitors. The final layout of one CDS can be seen in Figure 5.6

Coming from the left is the column bus, and from the right is sample, select and bias signals. The two large and elongated transistors in the middle are each a combination of the pMOS source follower and the Select pMOS, thereby the two gates. Notice that one gate connects to the nearby nMOS capacitor and the other to the Select signal entering from the bottom right in the figure. This
made a compact solution where all 4 pMOS transistors share the same small VDD connection situated between them. The remaining nMOS transistors are small thus more easily placed while still conserving the symmetry. The top wire, with respect to the image, overlaps with the neighbouring CDS. This is not a problem as long as the neighbour has room for it. The bottom of the CDS in the figure is therefore free of wires. Figure 5.7 shows the CDS when placed together with it’s neighbours.

Figure 5.6

Figure 5.7: Three CDS placed together, rotated by 90 degrees with respect to the actual layout
5.4 Top and pad frame

A pad frame was first built to meet the original size requirement of 5mm². The first attempt to fit the completed layout inside this failed by one pad width only, to our great disappointment. Many suggestions were made in order to make it fit without sacrificing resolution and functionality, including removing pads from one of the edges. After consulting the Group leader at NANO it was decided that by adding another student’s design to the chip the total area could be increased. This lead to a collaboration with Espen K. Nilsen, and his design is found in the final layout where all the pads on the right edge are reserved for his use.

For a complete view of the final top layout it is referred to Appendix A. In the top layout phase all the components are placed together and connected. By designing everything based on the pixel pitch a lot of space is saved and time spent on connecting the components is reduced to almost zero. When connected, the complete circuit is placed within the pad frame and the pads are connected to their respective signals.

5.4.1 Pads

The final layout comes with two VDD and ground pads on each of our three sides to ensure good connectivity. One of these supplies VDD to the pads themselves, and the other to the circuit. The decision to separate these was made to ensure a stable supply to the circuit. All digital signals are connected to pads with large built in buffers. The analogue inputs, that is bias signals, are connected to analog pads with built in 1kΩ resistors for maximum ESD\(^1\) protection. To minimize the resistance in the output signals, whilst still maintaining some ESD protection, analog pads with 200Ω were chosen.

5.4.2 Inner circuit

The metal 4 layer shielding the pixels and supplying VDD was extended to supply all components, and connected to the VDD pads. The metal 4 layer was made as big as possible to provide as much capacitance to this layer as possible, giving a stable VDD signal. The area of the metal 4 layer is limited to a certain degree and to avoid design rule violations evenly spaced slots had to be inserted. This is to reduce stress on the layer during fabrication. A guard ring\(^2\), connected to ground, around the pixels is added to ensure a good ground signal connection to the substrate. This also shields the pixels from noise by the surrounding circuits.

\(^{1}\)Electrostatic discharge

\(^{2}\)This can be seen in Appendix B

41
Chapter 6

Image sensor test system

In order to test and verify the Image sensor IC, it was decided to mount the image sensor on a purpose designed PCB, and to control this using the terasIC development kit, DE0-Nano-SoC kit. The development kit has an Altera Cyclone V System-on-Chip (SoC) combining both FPGA and a Hard processor system (HPS) with ARM dual-core Coretex-A9, allowing flexibility and reprogrammability. Quartus Prime Lite Edition is used to compile the VHDL code, assign signals to the output pins and program the FPGA. The subprogram in Quartus Prime called Qsys is used to implement standard library logic and add connections between FPGA and HPS. The C-code has to be compiled using Altera’s embedded command shell. How to obtain and install all the required software is described in the manual that comes with the dev.kit[15]. In addition there are some beginner exercises on the use of the programs, and a guide to help you get everything running the first time. Finally some MATLAB code was implemented to perform image analysis.

In this chapter each component and program written to test the Image sensor will be described. Figure 6.1 shows a block diagram of the main blocks in the test system, and this chapter will starts from the top (PCB) and work it’s way downwards.

It should be noted that the entirety of the test system, except from the PCB, is based on previous work done by Wilhelmsen M. including the choice of dev. kit and the lens mount. The VHDL code, C program and Java program is therefore version of his original work, modified to be compatible with the image sensor for this project. The on chip memory was added by the author to increase over all performance.

---

1For the dev. board user manual download DE0-Nano-SoC User Manual. For complete guides on how to obtain and use the software, DE0-Nano-SoC CD-ROM can be downloaded as a .zip file
Test system workload

<table>
<thead>
<tr>
<th>Marius</th>
<th>Soman</th>
</tr>
</thead>
<tbody>
<tr>
<td>VHDL</td>
<td>PCB</td>
</tr>
<tr>
<td>C program</td>
<td>Lens mount</td>
</tr>
<tr>
<td>Java</td>
<td>MATLAB code</td>
</tr>
</tbody>
</table>

Figure 6.1: Testbench block diagram
6.1 PCB

The printed circuit board allows for the Image sensor, in its JLCC84 package, to be placed in a surface mounted socket. Signals from the chip goes through the socket and via conductive tracks, called traces. Most of the IS-signals connects to two pin rows $2 \times 20$ pins, 2.25mm pitch placed apart so they match the GPIO\textsuperscript{2} headers of the development board. This allows for the PCB to be stacked directly on top of the dev. board, thus allowing the controlling signals from the dev.kit connect to the chip. The two signal outputs goes from image sensor to an 14-bit ACD, discussed in the next section. The PCB has 4-layers where the top and bottom layers are for traces and the two in the middle for power and ground. Compared to boards with fewer layers, this allows for versatility when designing, large ground and power planes for power stability and in general shorter traces reducing noise and RC-delay. The board is designed to be powered with 5.0V directly from the dev. board through GPIO, via USBmini A/B connected to any 5V USB port, or from external power supply via pin connector and ground pin. It comes with an on board voltage regulator that supplies 3.3V required for the image sensor and the pot-meters used to adjust bias voltages and the TX-control analogue input signals. The two test points from the image sensor chip each have their own pin on PCB for easy to reach measurements with oscilloscope. For more detail on the design of the PCB see Soman Cheng’s thesis.

6.1.1 Analog to digital converter

The analog to digital converter was chosen to meet some initial requirements.

Sample Rate

A rough estimate to the required sample rate can be calculated as

$$\text{Sample rate} = \frac{S}{F_f} \times \text{desired framerate} \left( \frac{F}{s} \right)$$

With a desire of minimum 15 frames per second

$$128^2 \text{pixels} \times 15 \text{FPS} \approx 0.25 \frac{MS}{s}$$

Differential input

The ADC has to support differential input due to the differential output from the image sensor.

\textsuperscript{2}General Purpose Input/Output
Parallel output

A parallel output of the digital data is preferable. Seen from the dev. boards perspective it is a lot faster to read the data in parallel rather than serial as no buffering is required. Making sure that the dev. board has enough inputs available to support all parallel outputs from the ACD.

Input range

The input range of an ADC is the signal range it can interpret and convert into a digital signal. Based on simulations the output image signal, relative to the reset signal, has a range of about 1.1V. The ADC should be able to at least convert this full range, and not have a input range so large that it will struggle to interpret the signal.

Resolution

The resolution\(^3\) of an ADC is the number of levels the analog input can be quantized in to, within it’s input range. The resolution is often given in terms of bits, where an 8-bit resolution has \(2^8 = 256\) quantization steps. The higher the resolution the better, and normal cameras use a minimum of 8-bits, typically 12-14 bits, and all the way up to 16-bits resolution[5].

The chosen ADC is LTC1746. It not only meets all the requirements but was also used by Stian and Mathias with a fortunate outcome. The VHDL code for controlling the ADC is therefore also implemented, saving time in this project. Some ADC parameters is shown in Table 6.1, refer to datasheet[9] for more detail.

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Min</th>
<th>Typ</th>
<th>Max</th>
<th>Unit</th>
</tr>
</thead>
<tbody>
<tr>
<td>Sample Rate</td>
<td></td>
<td>25</td>
<td></td>
<td>MSps</td>
</tr>
<tr>
<td>Resolution</td>
<td>14</td>
<td></td>
<td></td>
<td>bits</td>
</tr>
<tr>
<td>Analog Input Voltage</td>
<td>±1.1</td>
<td>±1.6</td>
<td>V</td>
<td></td>
</tr>
<tr>
<td>Analog VDD</td>
<td>4.45</td>
<td>5</td>
<td>V</td>
<td></td>
</tr>
<tr>
<td>Digital output</td>
<td>0.5</td>
<td>3.3</td>
<td>V</td>
<td></td>
</tr>
</tbody>
</table>

\(^3\)Also called bit depth
6.1.2 Lens and lens mount

The lens described in Table 6.2 is used to focus light onto the image sensor.

Table 6.2: Lens specifications

<table>
<thead>
<tr>
<th>Brand</th>
<th>Computar</th>
</tr>
</thead>
<tbody>
<tr>
<td>Model</td>
<td>T0412FICS</td>
</tr>
<tr>
<td>Focal length</td>
<td>4mm</td>
</tr>
<tr>
<td>f-number</td>
<td>f1.2</td>
</tr>
<tr>
<td>Mount</td>
<td>CS</td>
</tr>
<tr>
<td>Focus</td>
<td>0.2m - Inf</td>
</tr>
<tr>
<td>Max image format</td>
<td>4.8mm x 3.6mm</td>
</tr>
</tbody>
</table>

In addition, a lens mount has been 3D-printed to hold the lens and fit on the chip socket. The lens mount is a modified version of the mount used in M. Wilhelmsen’s project. It has been modified with some indents in the bottom, so surface mounted capacitors can fit underneath. For more detail see Soman Cheng’s thesis.

6.2 System on chip

The system on chip consists of a FPGA which runs VHDL code and a HPS that boots in a lightweight Linux environment and runs a C-program. The FPGA handles all the signals going to and from the PCB, which includes controlling the image sensor and controlling/receiving data from the ADC. The HPS communicates with the FPGA through the AXI-bridge, and can send images to a computer via Ethernet. Details on each of these will be described in this section.

6.2.1 FPGA

The VHDL code running on the FPGA consists of a TOP entity which connects the Image sensor control, the ADC control and the Qsys generated soc_system components.

Image sensor control

The image sensor control\(^4\) consists of two finite state machines (FSM) named row timing and column timing, both with synchronous reset. State diagrams for both FSM’s are found in Figure 6.3. The row timing state machine will be referred to as

\(^4\)Filename CMOSIS_CTRL.vhd
ROW and column timing as COL in this section. To begin, the init state resets all signals used to default values in each of the state machines separately. These signals includes row- and column addresses which both are set to address zero. If reset it not ROW jumps to Global reset while COL waits. In Global reset state the M1 and Global reset signals are toggled on and a counter starts. The counter increments it’s value for each clock cycle until it reaches a certain value. This value is a constant that can be edited to liking and decides the on time for the M1 and global reset pulses in this case. Multiple values exist for different on times and delays used by different signals and states. When the value has been reached the M1 and Global reset signals are set to zero and the hdr_en=1 check brings ROW to either sTX or sHDR. The counter is also reset, bringing it back to zero, making it ready for the next timing event.

The sTX and sHDR takes care of the exposure time and the TX control signals. sTX follows a normal in pixel signal transfer and sHDR implements the skimming technique.

Next state is sSEL_SIG which controls the SAMPLE SIGNAL in the sample and hold circuit the pixel signals for the current row onto the CDS, starting at zero. sRST state resets the FD node and sSEL_REF state samples the reset signal.

Now that both image signal and reset signals are stored in S&H on the image sensor, ROW waits while COL jumps to sSEL state. This state starts at column one and toggles the SELECT OUTPUT signal in the S&H circuit and sends a do sample signal to ADC control telling it that a signal is ready to be converted. It also sends the current row- and column address to ADC control. When this is done it starts on the next column until it reaches the end. As soon as all columns have been selected it goes back into wait state and ROW jumps to sNEXT state. Here it increments the row address by one and starts at sSEL_SIG again. When the last row has been read out sNEXT state jumps to sFIRST instead where row is set to zero and a new frame can begin with the Global reset state.
All output signals from image sensor control, except those going to ADC control, are connected to its own physical pin one the dev. board and routes to the PCB. What signal good where is decided in Quartus pin planner.

ADC control

The ADC control sends the necessary stimuli to the ADC, ties the ADC output to the correct pixel address and stores the data in the on chip memory.

ADC control tells the ADC when to encode a signal by command from the image sensor control as mentioned. It does so by toggling the ENC (encode) and ENC signals on the ADC. When the ADC is done with encoding the signal it enables its CLKOUT signal which is interpreted by the ADC control and the data on the ADC’s 14 output bits are read. The ADC uses 5 clock cycles to convert the data. This means that the row and -column addresses sent to the ADC control from image sensor system are delayed by a FIFO with 5 steps before it is tied to the data belonging to this pixel.
ADC control then writes the data to the ON CHIP RAM using the row and column addresses to determine the location in memory. This ensures that pixels are placed in the correct location in the final image.

6.2.2 Qsys generated soc_system

Qsys automatically generates interconnect logic for communication between the FPGA and HPS. Components can be selected from a list and connected graphically, and based on this the program generates a VHDL system and a component named soc_system.vhd. The Soc_system component contains the inputs and outputs to the system and can be included in the users VHDL code to connect to desired components. The system used for this project consists of several memory mapped Parallel input outputs registers and on chip memory. In addition to this the Cyclone V Hard Processor System is added in order to access the memory mapped registers through the AXI bridge. The HPS is also needed to run the C-program, and the memory spaces are defined in a header file that is generated based on the entire system.

Parallel input output, PIO

Parallel input outputs are used to configure the VHDL code while it’s running. In other words a signal in image sensor control can be accessed and changed by the HPS through a PIO. Each PIO has a clock and reset input, a slave port and an external connection port. The clock and reset is connected to the main system clock and the memory mapped slave port to any AXI master. The external connection port is exported, and will show up in the generated VHDL code soc_system. Table 6.4 shows the PIOs used in this system and their corresponding signals.

On chip memory

The original ADC control by Wilhelmsen used a single PIO to serially transfer the image data and the location of the pixel it belongs to. This is sufficient for their rolling shutter operation, but gives a low frame rate. It is also very susceptible to frame tearing, i.e., where an image is split in half containing information from two different frames. This occurred in the system due to the lack of timing control between the FPGA and the HPS. The data was sent continuously from the FPGA to the HPS, where one would end up reading/sending faster then the other. It is crucial to avoid these artifacts when trying to capture a fast moving object, and the on chop memory has implemented for this reason. This allows for faster access, lower
The on chip memory is defined to have a data width of 16-bits to be able to fit the 14-bit data from the ADC. The total size of the memory is 32768 bytes, two bytes per pixel and $128 \times 128 = 16384$ pixels. It comes with dual port access in the form of two slave ports where one is exported to FPGA and the other connected to an AXI master. On the FPGA side ADC control writes the data, and the HPS only reads. It is configured to “don’t care” if there is a read access during write. Another option would be to strictly read old data. The decision of don’t care was made since the HPS gets told when to perform a read by ADC control. The on chip memory has 6 input ports: clock, chip select, write enable, address and write data. It also have one output port for data read. Figure 6.5 shows the waveform required to perform either read or write operation.

---

5The pixel location does not need to be transferred as the memory addresses reflects the pixel position in the image
AXI bridge

When a HPS is added in Qsys it includes a lightweight AXI master bridge by default which can connect to any Avalon memory mapped slave port. Each of the PIOs and the on chip memory act as slaves. Each of the PIOs has their own allocated address space defined by the user, and they cannot overlap. For example, pio_use_hdr has the reserved memory allocation 0x0000_0040-0x0000_004F whilst pio_config uses 0x0000_0060-0x0000_006F. It should be noted that the PIOs used in this project have a maximum size of one byte and together they are a total of 27-bits. This means that they could be combined into only one PIO since it allows up to 32-bits data. The PIOs in this project were added as they were needed at different stages in the project. It turned out to be easier and faster to just add an extra PIO rather than combining two. This wastes a lot of memory space, but as long as there is space available this solution is acceptable. One of the on chip memory slave is also addressed by the same lightweight AXI master with address space 0x0002_0000-0x0002_7FFF.

6.2.3 C program

The embedded C program acts as a link between the user and the FPGA by accessing the PIOs and on chip memory in one end, and sends/receives data through Ethernet in the other. The C program is made to act as a host while the other end of the connection act as client. The compiled program is called arm2fpga and can be run from the Linux terminal using the command ./arm2fpga, for reference.

Before the C program can access the PIOs and the on chip memory, a header file is generated containing definitions of these. This is done by placing a provided shell script "generate_hps_qsys_header.sh" in the project folder. Altera’s embedded command shell is now used to run the script. This generates the header file with name hps_0.h which is included in the C project.

The C project contains in general the files fpga.c, network.c and main.c. This section will give a brief explanation to what each of these files do, thus providing a detailed overview of the different tasks of the C program.

fpga.c

fpga.c maps the address space for each PIO and the on chip memory into registers. It also adds pointers to these for ease of use in later code. This is done by including the hps_0.h file which contains a list of all base addresses to the PIOs and the on chip memory.
Table 6.3: HPS command menu

<table>
<thead>
<tr>
<th>Command</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>exp_time</td>
<td>Exposure time 0-255.</td>
</tr>
<tr>
<td>use_hdr</td>
<td>[0</td>
</tr>
<tr>
<td>tx_full</td>
<td>0-255, pulse width of TX, not used when HDR=1</td>
</tr>
<tr>
<td>time_mult</td>
<td>0-3, multiplies exp_time with 2^time_mult</td>
</tr>
<tr>
<td>test</td>
<td>Frame stream test (soft)</td>
</tr>
<tr>
<td>echo</td>
<td>Simple echo test</td>
</tr>
<tr>
<td>frame</td>
<td>Read a frame</td>
</tr>
<tr>
<td>exit</td>
<td>Exit camera control interface</td>
</tr>
<tr>
<td>info</td>
<td>Show info about stream hardware</td>
</tr>
<tr>
<td>help</td>
<td>Show help</td>
</tr>
</tbody>
</table>

**network.c**

This file initializes the TCP/IP protocol used to communicate with the PC through the Ethernet connection. The connection port is defined here and is set to be "56119" in this case. It also contains the functions for sending and receiving data between host and client.

**main.c**

In the top of this file the size of the image it is expected to read is defined as of number of rows and number of columns. This has to be changed if this program is to be used on a different image sensor format. An image read is done by reading the *on chip memory* sequentially. It then builds a frame and sends it to the PC via Ethernet, all on the users command.

The file starts with calling the functions mentioned above in order to establish connection with the FPGA registers and prepare the Ethernet protocol. After initialization it starts a loop that continuously waits for a connection. Once a connection is established it waits for a command from the client. If a command is followed by a value it updates the value in the FPGA through the corresponding PIO. If a command is sent without a value it returns the current value in the register to the user. By sending the command help it returns the command menu found in *Table 6.3*
6.3 PC

The PC communicates with the HPS using a USB cable connected to a USB to UART interface on the dev. kit\(^6\). PuTTY\(^7\) is used to emulate a terminal which connects to the HPS through serial connection. Once connected the device should be given an IP address by running the `ifconfig eth0 x.x.x.x` command. For first time use the `arm2fpga` program can no be transferred using Windows PowerShell or similar using the command `scp arm2fpga root@x.x.x.x:/home/root arm2fpga`. The program can then be executed through the PuTTY terminal using the command `./arm2fpga`. The executed program will now give notice to the user to program the FPGA, and then to connect through Ethernet cable.

To program the FPGA a USB is connected from the PC to the USB Blaster II port\(^8\). The programming is then done through Quartus. To establish a connection and provide a user interface for viewing images sent from the camera a Java program is written.

6.3.1 Java

The Java program provides a graphical user interface GUI for ease of use. The main function of the GIO is to enable the user to capture images. Once captured the image is stored as a .tiff file in a folder in the same directory as the program. To run the program a Java installation is required.

In the bottom left corner of the interface, see Figure 6.6, is a terminal for the user to input commands to the HPS. Next is the Device interface which allows the user to enter the IP address and port, together with a Connect button to start connecting. Whenever streaming a live histogram of the image is shown in the Histogram section. The Frame Preview section allows the user to start a video stream by ticking the Stream box. The slider next to it decides the streams update frequency, by telling the program how long it should wait between each frame request. A single capture can be made and stored in a file using the Capture button, with the option of deciding a file name. The big slider adjust the signal range visible on the stream, this can be used to gain up the image in preview by narrowing the range. An auto range box can also be ticked which adjusts the range to go from the lowest to the highest detected value. The Correction section allows for a Black image to be stored in the program, and if the Sub.Black box is ticked the preview shows the subtracted result.

---

\(^6\)A driver installation is necessary for the PC to recognize both UART and USB Blaster II connectors. This is described in the previously mentioned getting started guide provided with the kit[15]

\(^7\)a free and open-source terminal emulator

\(^8\)A driver installation is needed, see 5
This subtraction does not affect a captured image and is only for use in streaming mode.

![Figure 6.6: The Java programmed user interface](image)

### 6.4 Test system verification

Before attaching the chip to the PCB, and running the first test, some verification of the FPGA and PCB were performed to prevent unnecessary damage to the chip.

To begin the signals from FPGA were assigned to the pins matching the PCB. It turned out that the signal belonging to column address 3 were routed to a reserved 3.3V pin on the FPGA. The signal trace was cut off under a microscope using a scalpel and a wire was soldered from the chip package to the newly assigned pin. The signals from the FPGA pins were measured on oscilloscope before the PCB was attached. This was done to verify that the signals were assigned to the correct pins and that their timing relative to each other worked as expected.

The PCB was then connected and signals were measured again with oscilloscope to see if there could be any grounded signals or other discrepancies. It was discovered that the moved signal, column address 3, was not properly soldered and gave a weak response. This was fixed and a final verification with the oscilloscope was performed and Figure 6.7 shows some of the verified signals.
6.4.1 Skimming problems

The first implementation of the HDR skimming scheme gave no results, meaning it was no noticeable difference in the image when running the normal and HDR mode. It turned out that the reset between levels had been left out so the HDR scheme acted just like a normal full transfer. The next implementation made use of only two of the three levels to reduce the variables. Only having to adjust and concentrate on one pot-meter helped speed up the process. Figure 6.8 shows the timing diagram for the second attempted implementation. Notice from the plot that s1 and s2 are used separately and not together. Usually both are on at the same time to get a full 3.3V pulse to TX. For this implementation it was chosen to adjust the s2 triggered signals pot-meter to give 3.3V instead.\footnote{Notice also the very narrow s2 pulse. At this point the minimum TX pulse width required for a full transfer from PD to FD had been determined, and this pulse is well within this requirement. The plot is zoomed to show the whole exposure time, so even though s2 looks narrow it is in reality 75 clock periods whilst the exposure time is 4096.}

This implementation gave some results, as the saturated areas in the image were dimmed in comparison with normal capture. The problem now was that the saturated areas were dimmed only and revealed no more information than when saturated. It became clear that this was due to the short exposure time between skim...
and read. This lead us to the final implementation where all levels were included.

It was decided to divide the exposure time into three, where the middle exposure time is half as long as the first, and the short exposure time is a quarter. A way to express the different points in time based on the total exposure time and the relation between the different exposures has been found. By giving the short exposure time a value 1 the medium has to be 2 and the long 4. This gives us that the total exposure time can be divided into $1 + 2 + 4 = 7$ partitions. The implemented timing can be explained as:

- Exposure starts at time zero after reset
- The first skim occurs after $(\text{Exp\_time} \times \frac{4}{7}) - \text{rst\_time}$
- The first transition occurs after $\text{Exp\_time} \times \frac{4}{7}$
- The second skim occurs after $(\text{Exp\_time} \times \frac{6}{7}) - \text{rst\_time}$
- The second transition occurs after $\text{Exp\_time} \times \frac{6}{7}$
- The full transfer is performed after $\text{Exp\_time}$

The implemented timing diagram can be found in Figure 6.9.

The TX voltage levels were found by adjusting the pot-meters while looking at the live histogram. It was noticed that as the levels increased some portion of the saturated area could be moved towards the middle values. This resulted in the skim levels 1.5V and 0.9V. Figure 6.10 shows the histogram with and without HDR for images captured of the same scene.
6.4.2 Full well capacity

Parasitic capacitance extractions were done in layout and measured to be 1.97fF for the FD node and 11.0fF for the PD. The accuracy of these numbers is unknown, however if this is the case then a larger source follower could increase the FD nodes capacitance.
Part III

Results
Chapter 7

Results

This chapter begins with presenting the results in the form of images taken. Global shutter images are presented in section 5.1 and HDR in section 5.2. These sections will discuss the overall image quality from a visual point of view. Section 5.3 will present some measurements taken where general performance is the focus. Finally, some future work suggestions are listed and explained.

7.1 Global shutter performance and image quality analysis

In order to verify the global shutter mechanism, an image has been captured of a fast rotating fan. An image of the same fan is captured by a Samsung Galaxy S7 edge using a rolling shutter for comparison. A shutter speed of 1/750s was used for both cameras.

The images clearly show the efficiency of the global shutter when capturing fast moving objects. It should be noted that the global shutter image was under bright illumination and the image has been gained up in an image editing program. By analyzing the global shutter image, we see that it has several artifacts.

Dark column

The dark column on the left is the first column which is connected to a test point routing to the PCB. Due to this, it is assumed that there is some capacitance added to the column thus making it darker than the rest.
Saturated pixels

A clear line of saturated pixels can be found in the first few rows and in the rightmost column. It has been determined that this is due to the lack of vertical structures in the layout. Since there is no structure between substrate and metal 4 in these areas some light is allowed to enter in this gap. This could have been avoided by adding some vertical dummy structure in order to close the gap. Comparing the image to the top layout found in Appendix A we see that there is no saturation in the exact locations where TX-control and M1 buffer is situated. These have vertical structures all the way up to metal 4 and will therefore prevent light from entering.

It can be argued that the metal 4 layer greatly affects the light sensitivity of the image sensor. In this the case the saturated area represents the conceivable sensitivity which could have resulted in a contrast much closer to the Samsung image. This would have made a big difference as extremely short exposure times would have been possible and even faster moving objects could have been captured. On the other hand if the FD node is fully exposed the parasitic light sensitivity could become too high and artifacts would arise. The fact that the shielding metal layer is elevated with respect to the PD can be seen under a microscope as in Appendix D. It is noticed that if light is not aimed perpendicular to the pixel, some shadow occurs on the PD. This could explain the slight shading in the upper right and left corners of the image as light here hits at an angle. The best solution to this would be to increase the size of the metal 4 opening while still ensuring good coverage of the FD node.
Fixed pattern noise

There is a noticeable amount of vertical fixed pattern noise in the image. It should be noted that these are extra visible due to high gain in post processing. Gain is necessary in this case due to the short exposure time required, regardless of the object being under bright illumination. Figure 7.2 shows the image before and after dark frame subtraction.

Figure 7.2: Shows the original image (a) and the FPN reduced image (b)

7.2 High dynamic range

This section will present images taken with and without HDR enabled, at different exposure times, for comparison. All HDR images are taken with the same skimming levels to see if one level is adequate for all scenes to be captured. The first images can be seen in Figure 7.3 We see from the images that the sky is fully saturated in the normal image while some light and dark areas in the clouds can be seen in the HDR image. On the other hand we see less contrast in the building and the structure in the bottom left from the HDR image.

Analyzing the images in Figure 7.4 we see that with further increased exposure the dimming in the bright areas are not as pronounced. The HDR image still has some effect as the silhouette for the far away building is complete. This image further demonstrates that there is no loss of data in the darker areas. With long exposure, as seen in Figure 7.5, the bright areas becomes more or less indistinguishable in the two images. The edge of the window frame clearly shows the lower contrast achieved with HDR enabled but the darker area is identical.
Figure 7.3: Normal (a) vs. HDR (b) image, exposure time: 2.46ms, weather cloudy

Figure 7.4: Normal (a) vs. HDR (b) image, exposure time: 8.19ms

Figure 7.5: Normal (a) vs. HDR (b) image, exposure time: 12.29ms
7.3 Measurements

7.3.1 Power consumption

In order to measure the power consumption, the current through the voltage regulator supplying the image sensor with 3.3V was measured using a multimeter. The current was measured to 3.41 mA giving a power consumption of \( P = 3.41 \text{mA} \times 3.3\text{V} = 11.5\text{mW} \). This result is probably slightly higher than the actual power consumption of the image sensor due to some consumption by the pot-meters in the same 3.3V power net.

7.3.2 Image sensor characteristics

Linearity

The linearity of the image sensor was measured by increasing the exposure time while capturing an image for each step. The test was performed in a completely dark room with only one fixed constant light source aimed directly at the sensor. A paper sheet was mounted to the lens opening to get an even light distribution throughout the image. For each exposure time a dark image has also been captured with a closed lens cap and in complete darkness\(^1\). The linearity is calculated as the mean signal value in a group of 10x10 pixels in the middle of the image.

![Figure 7.6: Linearity plot, shows signal output from chip to ACD over light intensity](image)

The maximum signal range is \( 1.48\text{V} - 0.805\text{V} = 0.675\text{V} \), and the linear region is approximated to be \( 1.40\text{V} - 0.805\text{V} = 0.595\text{V} \). It is assumed that the signal range is limited by the full well capacity of the FD node. A larger source follower will add

---

\(^1\)Some light from the PC display was present during measurements. The screen however was dimmed to max and the camera was shielded by a cardboard box. No difference in the live histogram could be noticed even if the PC display was fully lit.
extra capacitance to the FD node and could therefore contribute to a bigger signal range.

**HDR vs. no HDR linearity**

A linearity plot with HDR enabled is shown together with the normal capture linearity in Figure 7.7. The plot shows that the HDR scheme has successfully increased the optical dynamic range by increasing the exposure range. The two dents in the HDR graph represents the skimming levels.

**Figure 7.7: Linearity plot for normal and HDR operation compared**

TX transfer characteristics

A test has been performed in order to assure that the pixel is capable to perform complete transfer of signal from the PD to the FD node. This was done by fully saturating the pixels using a light source whilst gradually increasing the width of the TX pulse.

From the plot in Figure 7.8 we see that a full transfer is accomplished using a pulse width of at least 80ns. This is not the equivalence of completely emptying the PD of charge but rather filling up the FD node to full potential, limited by it’s full well capacity.

**Parasitic light sensitivity**

Parasitic light sensitivity was measured by turning the TX pulse completely off to only measure the parasitic light and not from the PD. The M1 transistor was also open to keep PD at reset level in order to prevent it from going into forward bias.
Images were taken under strong illumination, and the same calculations as done with linearity measurements gave the plot in Figure 7.9.

A comparison of the stray light intensity to the signal intensity in the linearity plot can be made. Extracting the stray light value and calculating an approximate stray light contribution to the output signal gives:

Signal strength at 0.082ms (shortest exposure time recorded):
- Stray light: 799mV
- Normal signal: 805mV

Stray Light contribution at 2.46ms:
\[
\frac{799.3mV - 799mV}{1.447mV - 805mV} \times 100% = 0.062\%
\]

The relatively low stray light value together with the low light sensitivity of the image sensor suggests that we have high shutter efficiency.
Temporal noise

Temporal noise is calculated by taking the standard deviation of two images, where one is subtracted from the other. The same set of images as for the linear plot is used here as well with and additional capture for each exposure. Both captures were taken in the same environment. The standard deviation is usually calculated for the whole image. It was however chosen to use only 100x100 pixels in the middle to avoid the areas along the edge where artifacts have been found. This is done with increasing exposure time and provides the plot in Figure 7.10. As expected the temporal noise is dominant for short exposure times with low image signal, and has less effect as the image signal increases.

Fixed pattern noise

Fixed pattern noise is calculated as the standard deviation on the mean value for all columns or rows. This is done on the same image set as used in the linearity plot providing a plot of FPN versus exposure time. Figure 7.11 shows the resulting plot. The graph shows more FPN in the vertical direction which agrees with the images. This noise is probably due to different offsets in the pMOS source followers in the CDS circuit.
7.4 Future work

This section begins with mentioning some improvements that can be made to the completed project. Following are some suggestions to future projects.

7.4.1 Improvement

Skimming technique and HDR improvements

In order to optimize the skimming method, further testing of the skimming levels can be made.

- Plot linearity vs. TX pulse height. Useful information for adjusting skimming levels.

- Make changes to the PCB to allow for higher voltages (up to 4V) to the TX pulse in order to see if charge transfer can be further increased.

- Decreasing the input sense range of the ADC to fit the output signal from the image sensor can increase the dynamic range drastically.

Further characterization of image sensor

Create a platform that allows for full test of image sensor including conversion gain, quantum efficiency and image lag.

Timing improvements and additions, VHDL

- Implement integrate while read timing in VHDL to allow higher frame rates.
• Implement rolling shutter timing on the same image sensor for direct comparison.

7.4.2 Future projects

• Design a global shutter image sensor with 6T pixels allowing true correlated double sampling.

• Design a pixel with a custom PD together with an optimized FD node to potentially increase the full well capacity and increase fill factor.

• Implement on chip ADC for faster readout

• Implement on chip amplifier to increase signal output range to ADC.
Chapter 8

Conclusion

A CMOS image sensor has been designed using AMSopto $0.35\mu m$ process with global shutter mechanism and HDR capability. The image sensor consists of a 128x128 pixel array, readout circuitry and special components for global shutter and skimming operations.

A PCB and lens mount has been designed and fabricated to mount the image sensor and the camera lens. Software has been developed to control timing and reading out data while converting it to a visible image. All of this has together been used to test and verify the image sensors functionality.

The global shutter functionality has been verified through comparison with a rolling shutter camera under the same conditions. The parasitic light sensitivity have been measured and the conclusion has been that the image sensor has a high shutter efficiency.

Furthermore the HDR skimming technique has been implemented and verified through comparison with a normal capture operation. Linearity measurements have been performed to further verify the increased dynamic range.

One of the main problems with the image sensors is it’s low light sensitivity, assumed to be caused by to the metal 4 shielding. This caused problems when capturing a fast moving object to verify the global shutter mechanism. In this case a short exposure time is required for a sharp image and a strong light source was needed to compensate for this. Additionally, even though the HDR scheme has been verified it only makes a noticeable difference for ”normal” exposure times. This has leads to the conclusion that the combined benefits of having a global shutter and HDR has not been achieved. The shielding has, on the other hand, protected the FD node from being exposed leading to the high shutter efficiency.
Bibliography


Appendix A

Top layout of CMOS image sensor
Appendix B

Pixel layout and pixel Array layout
Appendix C

Full Image sensor under microscope
Appendix D

Pixel array under microscope